

Laporan Praktikum

Algoritma dan Pemrograman

PyQt5 : Widget & Layout



Asisten :

Pahril Dwi Saputra	11221056
Zaky Dio Akbar Pangestu	11221050

Disusun Oleh :

Patra Ananda	10241061
Vanessa Marie Tandiarru	11241086
Jeshua Austin Daceka	10241037
Vera Friska	10241073
Aliya Labibah	10241007
Siti Fatimah	10241067

12 November 2024

Dasar Teori

1. PyQt5

PyQt adalah lintas platform GUI perangkat Qt yang mengikat pada python yang diimplementasikan sebagai plug-in. PyQt adalah perangkat lunak gratis yang dikembangkan oleh perusahaan Inggris bernama Riverbank Computing. PyQt tersedia yang sama untuk Qt versi yang lebih tua dari 4,5 ; di bawah persyaratan lisensi GNU General Public License (GPL) dan lisensi komersial, tetapi tidak GNU Lesser General Public License dan MacOS (atau Darwin OS)

PyQt mengimplementasikan sekitar 440 kelas dan lebih dari 6.000 fungsi dan metode termasuk :

- seperangkat widget GUI yang substansial
- kelas-kelas untuk mengakses SQL database (ODBC, MySQL, PostgreSQL, Oracle, SQLite)
- QScintilla, widget editor teks kaya berbasis Scintilla
- widget data aware yang diisi secara otomatis dari database
- aplikasi XML parser dukungan SVG
- kelas-kelas untuk menyematkan kontrol ActiveX pada Windows (hanya dalam versi komersial)

PyQt5 adalah kumpulan lengkap binding Python untuk Qt v5. Ita mengimplementasikan sebagai lebih dari 35 modul ekstensi dan memungkinkan python untuk digunakan sebagai bahasa pengembangan aplikasi alternatif C++ pada semua platform yang didukung termasuk iOS dan android.

Widget PyQt adalah elemen grafis yang digunakan untuk membuat antarmuka pengguna dalam aplikasi PyQt. Widget ini digunakan untuk berbagai tujuan, seperti menampilkan informasi, menerima input pengguna, atau menyediakan kontrol untuk interaksi. PyQt menawarkan berbagai widget bawaan yang dapat digunakan pengembang untuk membangun aplikasi desktop secara efisien.

Untuk mengatur widget pada window atau form di PyQt dapat menggunakan teknik berikut:

- Gunakan dan pada widget Anda untuk memberikan ukuran dan posisi absolut..resize().move()
- Terapkan kembali dan hitung ukuran dan posisi widget Anda secara dinamis..resizeEvent()
- Gunakan pengelola tata letak dan biarkan mereka melakukan semua perhitungan dan kerja keras untuk Anda.

Layout memiliki arti sebagai sebuah susunan ataupun rancangan. Lay out dapat kita artikan sebagai sebuah susunan atau tata letak tempat dimana kita akan meletakkan sebuah widget seperti teks, tombol, gambar dan yang lainnya.

- QHBoxLayout adalah salah satu dari dua tata letak kotak yang tersedia di PyQt, dimana untuk mengatur widget secara horizontal, satu disamping yang lain. Widget ditambahkan ke tata letak dari kiri ke kanan. Ini berarti bahwa widget yang Anda tambahkan pertama kali dalam kode Anda akan menjadi widget paling kiri dalam tata letak. Untuk menambahkan widget ke objek, anda memanggil objek tata letak. Metode ini mengambil satu argumen wajib dan dua argumen opsional : `QHBoxLayout.addWidget(widget, stretch, alignment)`
- QVBoxLayout, mengatur widget secara vertikal, satu dibawah yang lain. Untuk membuat tata letak vertikal dan mengatur widget dari atas ke bawah. Metodenya bekerja sama seperti di

```
QVBoxLayout.addWidget() QHBoxLayout
```

- QGridLayout, untuk mengatur widget dalam kisi baris dan kolom. Setiap widget akan memiliki posisi relatif di grid. Untuk menentukan posisi widget, atau sel dalam kisi, menggunakan sepasang koordinat formulir. Koordinat ini harus berupa bilangan bulat berbasis nol. `(row, column)`

Source Code

LP12.py

No	Implementasi Lp12.py
1	import sys from PyQt5.QtWidgets import * from PyQt5.QtGui import * from PyQt5.QtCore import *
2	
3	
4	
5	
6	class BarangApp(QWidget):
7	def __init__(self):
8	super().__init__()
9	self.setWindowTitle("Apps List Data")
10	self.setGeometry(1300, 130, 610, 450)
11	
12	self.judul = QLabel("Tambah Data Barang", self)
13	self.judul.setFont(QFont("Open sans", 18, QFont.Bold))
14	self.judul.setAlignment(Qt.AlignCenter)
15	self.judul.setGeometry(50, 20, 500, 40)
16	
17	self.input_nama = QLineEdit(self)
18	self.input_nama.setPlaceholderText("Nama Barang")
19	self.input_nama.setFont(QFont("Open sans", 10))
20	self.input_nama.setStyleSheet("padding: 10px; border: 1px solid #5865F2; border-radius: 4px;")
21	self.input_nama.setGeometry(10, 100, 250, 40)
22	
23	self.input_harga = QLineEdit(self)
24	self.input_harga.setPlaceholderText("Harga Barang")
25	self.input_harga.setFont(QFont("Open sans", 10))
26	self.input_harga.setStyleSheet("padding: 10px; border: 1px solid #5865F2; border-radius: 4px;")
27	self.input_harga.setGeometry(300, 100, 250, 40)
28	
29	self.button_tambah = QPushButton("Tambah Data", self)
30	self.button_tambah.setFont(QFont("Open sans", 12))
31	self.button_tambah.setStyleSheet("""
32	background-color: #5865F2;
33	color: white;
	padding: 10px;

```

34             border: none;
35             border-radius: 5px;
36         """)

37         self.button_tambah.setGeometry(10, 200, 200, 40)
38         self.button_tambah.clicked.connect(self.tambah_data)

39
40         self.table = QTableWidget(self)
41         self.table.setColumnCount(2)
42         self.table.setHorizontalHeaderLabels(["Nama Barang",
43 "Harga Barang"])
44         self.table.setFont(QFont("Arial", 12))
45         self.table.setGeometry(10, 250, 580, 150)
46         self.table.setStyleSheet("""
47             QTableWidget {
48                 border: 1px solid #5865F2;
49                 border-radius: 5px;
50                 background-color: #f9f9f9;
51             }
52             QHeaderView::section {
53                 background-color: #5865F2;
54                 color: white;
55                 padding: 10px;
56                 font-size: 14px;
57             }
58         """)

59     def tambah_data(self):
60         nama_barang = self.input_nama.text().strip()
61         harga_barang = self.input_harga.text().strip()

62
63         if not nama_barang or not harga_barang:
64             QMessageBox.warning(self, "Peringatan", "Tidak ada
65 list barang.")
66             return

67         try:
68             harga = float(harga_barang)
69             if harga < 0:
70                 QMessageBox.warning(self, "Error", "Harga tidak
71 boleh negatif.")
72             return
73         except ValueError:
74             QMessageBox.warning(self, "Error", "Harga harus
berupa angka.")

```

	<pre> return 75 76 row_position = self.table.rowCount() 77 self.table.insertRow(row_position) 78 self.table.setItem(row_position, 0, 79 QTableWidgetItem(nama_barang)) 80 self.table.setItem(row_position, 1, 80 QTableWidgetItem(f"Rp {harga:.0f}")) 81 self.input_nama.clear() 82 self.input_harga.clear() 83 84 if __name__ == "__main__": 85 app = QApplication(sys.argv) 86 window = BarangApp() 87 window.show() 88 sys.exit(app.exec_()) 89 </pre>
--	---

LP13.py

No.	Implementasi LP13.py
1	<pre> import sys from PyQt5.QtWidgets import * from PyQt5.QtGui import * from PyQt5.QtCore import * 6 class BarangApp(QWidget): 7 def __init__(self): 8 super().__init__() 9 self.setWindowTitle("Apps Manajemen Barang") 10 self.resize(700, 500) 12 main_layout = QVBoxLayout(self) 13 14 self.judul = QLabel("Isi Data Barang", self) 15 self.judul.setFont(QFont("Open sans", 18, QFont.Bold)) 16 self.judul.setAlignment(Qt.AlignCenter) 17 main_layout.addWidget(self.judul) 18 form_layout = QGridLayout() 19 </pre>

```

20         self.input_nama = QLineEdit(self)
21         self.input_nama.setPlaceholderText("Nama Barang")
22         self.input_nama.setFont(QFont("Open sans", 10))
23         self.input_nama.setStyleSheet("padding: 10px; border:
24 1px solid #5865F2; border-radius: 4px;")
25         form_layout.addWidget(QLabel("Nama Barang:"), 0, 0)
26         form_layout.addWidget(self.input_nama, 0, 1)
27
28         self.input_harga = QLineEdit(self)
29         self.input_harga.setPlaceholderText("Harga Barang")
30         self.input_harga.setFont(QFont("Open sans", 10))
31         self.input_harga.setStyleSheet("padding: 10px; border:
32 1px solid #5865F2; border-radius: 4px;")
33         form_layout.addWidget(QLabel("Harga Barang:"), 1, 0)
34         form_layout.addWidget(self.input_harga, 1, 1)
35
36         main_layout.addLayout(form_layout)
37
38         button_layout = QHBoxLayout()
39
40         self.button_tambah = QPushButton("Tambah Data", self)
41         self.button_tambah.setFont(QFont("Open sans", 12))
42         self.button_tambah.setStyleSheet("""
43             background-color: #5865F2;
44             color: white;
45             padding: 10px;
46             border: none;
47             border-radius: 5px;
48             """)
49         self.button_tambah.clicked.connect(self.tambah_data)
50         button_layout.addWidget(self.button_tambah)
51
52         self.button_update = QPushButton("Update Data", self)
53         self.button_update.setFont(QFont("Open sans", 12))
54         self.button_update.setStyleSheet("""
55             background-color: #5865F2;
56             color: white;
57             padding: 10px;
58             border: none;
59             border-radius: 5px;
60             """)
61         self.button_update.clicked.connect(self.update_data)
62         button_layout.addWidget(self.button_update)

```

```

62         self.button_hapus = QPushButton("Hapus Data", self)
63         self.button_hapus.setFont(QFont("Open sans", 12))
64         self.button_hapus.setStyleSheet("""
65             background-color: #5865F2;
66             color: white;
67             padding: 10px;
68             border: none;
69             border-radius: 5px;
70             """)
71         self.button_hapus.clicked.connect(self.hapus_data)
72         button_layout.addWidget(self.button_hapus)
73
74         self.button_reset = QPushButton("Reset Data", self)
75         self.button_reset.setFont(QFont("Open sans", 12))
76         self.button_reset.setStyleSheet("""
77             background-color: #5865F2;
78             color: white;
79             padding: 10px;
80             border: none;
81             border-radius: 5px;
82             """)
83         self.button_reset.clicked.connect(self.reset_data)
84         button_layout.addWidget(self.button_reset)
85
86         main_layout.addLayout(button_layout)
87
88         self.table = QTableWidget(self)
89         self.table.setColumnCount(3)
90         self.table.setHorizontalHeaderLabels(["No.", "Nama
91 Barang", "Harga Barang"])
92         self.table.setFont(QFont("Open sans", 12))
93         self.table.setStyleSheet("""
94             QTableWidget {
95                 border: 1px solid #5865F2;
96                 border-radius: 5px;
97                 background-color: #D3D3D3;
98             }
99             QHeaderView::section {
100                 background-color: #5865F2;
101                 color: white;
102                 padding: 10px;
103                 font-size: 14px;
104             }
105             """)

```

```

105     self.table.horizontalHeader().setSectionResizeMode(QHeaderView.Stretch)
106
107     self.table.setVerticalScrollMode(QAbstractItemView.ScrollPerPixel)
108
109     self.table.setHorizontalScrollMode(QAbstractItemView.ScrollPerPixel)
110
111     self.table.setSelectionBehavior(QTableWidget.SelectRows)
112
113     def tambah_data(self):
114         nama_barang = self.input_nama.text().strip()
115         harga_barang = self.input_harga.text().strip()
116
117         if not nama_barang or not harga_barang:
118             QMessageBox.warning(self, "Peringatan", "Nama dan")
119             harga barang harus diisi.")
120             return
121
122         try:
123             harga = float(harga_barang)
124             if harga < 0:
125                 QMessageBox.warning(self, "Error", "Harga kalo")
126                 negatif ya aneh lah .")
127                 return
128
129             except ValueError:
130                 QMessageBox.warning(self, "Error", "Harga harus")
131                 berupa angka.")
132
133             formatted_harga = f"Rp {int(harga):,}").replace(",",
134             ".")
135             row_position = self.table.rowCount()
136             self.table.insertRow(row_position)
137             self.table.setItem(row_position, 0,
138             QTableWidgetItem(str(row_position + 1)))
139             self.table.setItem(row_position, 1,
140             QTableWidgetItem(nama_barang))
141             self.table.setItem(row_position, 2,
142             QTableWidgetItem(formatted_harga))
143
144             self.input_nama.clear()
145             self.input_harga.clear()

```

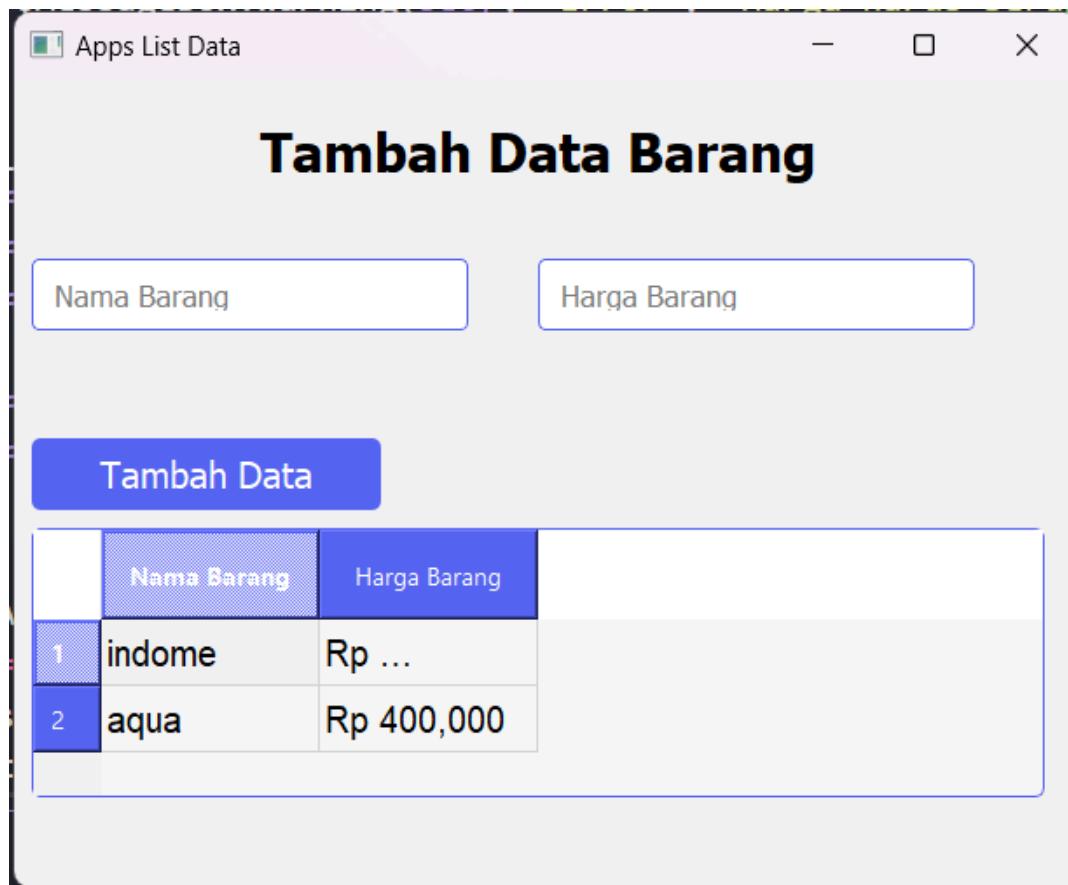
```

139         self.update_row_numbers()
140
141     def update_data(self):
142         selected_row = self.table.currentRow()
143         if selected_row == -1:
144             QMessageBox.warning(self, "Peringatan", "Pilih
145 baris yang ingin diupdate.")
146             return
147
148         nama_barang = self.input_nama.text().strip()
149         harga_barang = self.input_harga.text().strip()
150
151         if not nama_barang or not harga_barang:
152             QMessageBox.warning(self, "Peringatan", "Nama dan
153 harga barang harus diisi.")
154             return
155
156         try:
157             harga = float(harga_barang)
158             if harga < 0:
159                 QMessageBox.warning(self, "Error", "Harga tidak
160 boleh negatif.")
161                 return
162
163             formatted_harga = f"Rp {int(harga)}:,.replace(",",
164 ".")
165             self.table.setItem(selected_row, 1,
166 QTableWidgetItem(nama_barang))
167             self.table.setItem(selected_row, 2,
168 QTableWidgetItem(formatted_harga))
169
170         def hapus_data(self):
171             selected_row = self.table.currentRow()
172             if selected_row == -1:
173                 QMessageBox.warning(self, "Peringatan", "Pilih
174 baris yang ingin dihapus.")
175             return
176
177             self.table.removeRow(selected_row)
178             self.update_row_numbers()

```


Screenshot

ImplementasiLP12.py



Tampilan aplikasi, disini aplikasi tidak menggunakan layout hanya menggunakan Geometry (manual).

ImplementasiLP13.py

The screenshot shows a Windows application window titled "IsI Data Barang". The window has a title bar with standard minimize, maximize, and close buttons. Below the title bar is a header section with the title "IsI Data Barang" in bold black font. Underneath the header are two input fields: "Nama Barang:" and "Harga Barang:", each with a placeholder text "Nama Barang" and "Harga Barang" respectively. Below these fields are four blue buttons labeled "Tambah Data", "Update Data", "Hapus Data", and "Reset Data". The main area of the window contains a table with data. The table has a blue header row with columns labeled "No.", "Nama Barang", and "Harga Barang". The body of the table contains three rows of data, each with a blue background. The data is as follows:

No.	Nama Barang	Harga Barang
1	indome	Rp 30.000
2	aqua	Rp 100.000
3	sendok	Rp 50.000

Tampilan Aplikasi, ini tampilan UI/UX untuk menambah data pada tabel data barang.

Apps Manajemen Barang

IsI Data Barang

Nama Barang: indome

Harga Barang: 10000

Tambah Data Update Data Hapus Data Reset Data

	No.	Nama Barang	Harga Barang
1	1	indome	Rp 6.000

Apps Manajemen Barang

IsI Data Barang

Nama Barang: indome

Harga Barang: 10000

Tambah Data Update Data Hapus Data Reset Data

	No.	Nama Barang	Harga Barang
1	1	indome	Rp 10.000

Tampilan Pertama, disini user dapat meng-update data yang dimasukkan.

Apps Manajemen Barang

IsI Data Barang

Nama Barang:

Harga Barang:

Tambah Data **Update Data** **Hapus Data** **Reset Data**

	No.	Nama Barang	Harga Barang
1	1	indome	Rp 10.000
2	2	sendok	Rp 10.000

Tampilan Kedua, ini menghapus data dengan meng-klik data yang ingin dihapus pada tabel. Reset berfungsi untuk menghapus semua data yang ada pada tabel tersebut.

Pembahasan

ImplementasiLP12.py

No.	Pembahasan
1-4	Pada bagian ini, modul <code>sys</code> dan modul PyQt5 diimpor. <code>sys</code> digunakan untuk mengontrol sistem, sementara modul dari PyQt5 digunakan untuk membuat antarmuka pengguna seperti <code>widget</code> , <code>gaya</code> , dan pengaturan <code>event</code> .
6-10	Bagian ini mendefinisikan kelas <code>BarangApp</code> , turunan dari <code>QWidget</code> , yang menjadi kerangka utama aplikasi. <code>Window</code> aplikasi diberi judul "Apps List Data" dan ukuran serta posisinya diatur dengan <code>setGeometry</code> .
12-14	Pada bagian ini, label <code>QLabel</code> dibuat untuk menampilkan teks "Tambah Data Barang" sebagai judul aplikasi. <i>Font</i> teks diperbesar dan digarisbawahi dan posisinya diatur agar berada di tengah atas <code>window</code> .
16-21	Kemudian di bagian ini sebuah <i>input</i> teks dibuat menggunakan <code>QLineEdit</code> untuk menerima nama barang. <i>Placeholder</i> teks "Nama Barang" ditambahkan. Ukuran dan posisi input juga diatur agar terlihat di sebelah kiri atas <code>window</code> .
23-28	Dan pada bagian ini, <i>input</i> teks lain dibuat untuk menerima harga barang. Sama seperti <i>input</i> nama barang, <i>placeholder</i> teks "Harga Barang" ditambahkan. <i>Input</i> ini ditempatkan di sebelah kanan <i>input</i> nama barang dengan ukuran dan gaya yang sama.
30-31	Pada bagian ini membuat tombol menggunakan <code>QPushButton</code> dengan label "Tambah Data". Tombol ini akan digunakan untuk menambahkan data barang ke tabel. <i>Font</i> pada tombol diatur menggunakan jenis huruf "Open Sans" dengan ukuran 12.
32-38	Pada bagian ini, tombol diberi gaya visual melalui <code>setStyleSheet</code> . <code>background-color</code> digunakan untuk mengatur warna latar tombol menjadi biru (#5865F2), sedangkan <code>color</code> digunakan untuk mengatur warna teks menjadi putih. <code>padding</code> ditambahkan untuk memberikan ruang di dalam tombol, dan <code>border-radius</code> digunakan untuk membuat sudut tombol melengkung dengan radius 5px.
39	Tombol ditempatkan di posisi (10, 200) pada <code>window</code> , dengan lebar 200 piksel dan tinggi 40 piksel.
40	Pada bagian ini, tombol dihubungkan ke fungsi <code>tambah_data</code> menggunakan <code>clicked.connect</code> . Ketika tombol ditekan, fungsi tersebut akan dijalankan untuk memproses data yang dimasukkan pengguna.

42	Kode ini membuat objek tabel <code>QTableWidget</code> dimana tabel ini berfungsi untuk menampilkan data yang dimasukkan oleh pengguna, seperti nama barang dan harga barang.
43	Pada baris ini, jumlah kolom tabel diatur menjadi 2, yang masing-masing akan berisi "Nama Barang" dan "Harga Barang".
44	Di baris ini, nama kolom di tabel diatur dengan label "Nama Barang" dan "Harga Barang" menggunakan <code>setHorizontalHeaderLabels</code> . Ini akan muncul di bagian atas tabel sebagai <i>header</i> kolom.
45	Disini <i>font</i> tabel diatur menjadi "Arial" dengan ukuran 12, sehingga teks di dalam tabel lebih mudah dibaca.
46	Pada bagian ini posisi tabel diatur agar dimulai pada koordinat (10, 250) dan ukurannya diatur menjadi lebar 580 piksel dan tinggi 150 piksel. Ini menentukan di mana tabel akan ditempatkan di <i>window</i> aplikasi.
47-52	Di bagian ini gaya visual tabel diatur melalui <code>setStyleSheet</code> . Properti <code>border</code> digunakan untuk menambahkan garis tepi berwarna biru dengan ketebalan 1 piksel di sekitar tabel. <code>border-radius</code> membuat sudut tabel melengkung dengan radius 5 piksel, memberikan tampilan yang lebih halus. Sedangkan <code>background-color</code> memberikan warna latar belakang tabel dengan warna abu-abu terang.
53-59	Gaya <i>visual header</i> tabel diatur dengan <code>QHeaderView::section</code> , dengan latar belakang biru (#5865F2) dan teks putih untuk kontras. <i>Padding</i> 10 piksel agar tampilan lebih rapi, dan ukuran font diatur menjadi 14 piksel.
61-63	Bagian ini mengambil teks yang dimasukkan oleh pengguna di dua input (<code>input_nama</code> dan <code>input_harga</code>). <code>strip()</code> digunakan untuk menghapus spasi yang tidak perlu di awal atau akhir teks.
65-67	Fungsi ini memeriksa apakah <i>input</i> nama barang atau harga barang kosong. Jika ada yang kosong, sebuah pesan peringatan ditampilkan, dan fungsi dihentikan dengan <code>return</code> .
69-78	Di bagian ini, program mencoba mengonversi harga barang menjadi angka desimal. Jika konversi gagal, akan muncul pesan kesalahan. jika harga barang kurang dari 0, pesan kesalahan juga akan ditampilkan.
80-85	Bagian ini menambahkan data yang valid ke tabel. <code>rowCount()</code> digunakan untuk menghitung jumlah baris di tabel saat ini. Data barang baru dimasukkan ke baris baru yang ditambahkan dengan <code>insertRow()</code> . Nama barang dan harga dimasukkan ke dalam kolom yang sesuai menggunakan <code>setItem()</code> .
87-88	Setelah data ditambahkan ke tabel, <i>input</i> nama barang dan harga barang dikosongkan menggunakan <code>clear()</code> , sehingga dapat dimasukkan data baru.

90-94	Pada bagian ini, aplikasi dimulai. Objek <code>QApplication</code> dibuat untuk mengelola <i>event loop</i> aplikasi. Kemudian <code>window</code> dari kelas <code>BarangApp</code> ditampilkan menggunakan <code>show()</code> , dan aplikasi dijalankan hingga pengguna menutupnya.
-------	--

ImplementasiLP13.py

No.	Pembahasan
1-4	<code>Import.sys</code> ini digunakan untuk mengimpor dan mengakses data yang diperlukan oleh PyQt5. PyQt5 Adalah pustaka untuk aplikasi GUI, dengan <code>QtWidgets</code> merupakan fitur pada GUI yang berisikan komponen dasar untuk membuat aplikasi terdiri dari jendela utama, tombol, dan layout. <code>QtGui</code> fitur GUI untuk menyediakan kelas dan fungsi yang terkait dengan grafis serta tampilan visual. <code>QtCore</code> fitur pada GUI yang digunakan untuk fungsional dasar untuk aplikasi berbasis <i>event-driven</i> , dengan beberapa kelas yang digunakan.
6-8	Pada baris ini digunakan untuk mendefinisikan <code>BarangApp</code> merupakan sebuah kelas yang dibuat dan merupakan turunan dari fitur GUI <code>QWidget</code> yang akan menampilkan jendela utama (<code>window</code>). <code>def __init__(self)</code> digunakan untuk metode analisis memanggil kelas <code>BarangApp</code> dan untuk menyiapkan tempat ketika objek pertama kali dimasukkan. <code>super().__init__()</code> digunakan untuk memanggil kelas utama yaitu <code>QWidget</code> agar program berjalan dan dapat dianalisis dengan benar.
9-10	<code>self.setWindowTitle("Apps Manajemen Barang")</code> merupakan metode yang digunakan untuk mengatur judul dari jendela aplikasi, dengan lebar jendela aplikasi yang akan ditampilkan dengan lebar 700 piksel dan tinggi 500 piksel.
12.	<code>main_layout</code> merupakan sebuah variabel yang menyimpan <code>QVBoxLayout(self)</code> dimana ini adalah sebuah <i>layout</i> manager yang berfungsi untuk menyusun <i>widget</i> secara vertikal.
14-17	<code>self.judul</code> merupakan sebuah variabel yang menyimpan <code>QLabel("Tambah Data Barang", self)</code> dimana ini adalah sebuah kelas pada PyQt5 yang digunakan untuk menampilkan teks antarmuka pengguna di GUI. Dengan argumen “Tambah Data Barang” yang akan ditampilkan pada aplikasi. <code>setFont(QFont("Open sans", 18, QFont.Bold))</code> berfungsi untuk mengatur font dan gaya teks yang ingin digunakan pada widget <code>QLabel</code> dimana “Open sans” merupakan nama <i>font</i> yang ingin digunakan, 18 merupakan ukuran <i>font</i> yang akan ditampilkan, <code>QFont.Bold</code> untuk menampilkan teks dengan gaya <i>bold</i> . <code>setAlignment(Qt.AlignCenter)</code> digunakan untuk mengatur perataan dalam teks <i>widget</i> <code>QLabel</code> agar teks terletak tepat di tengah-tengah. <code>addWidget(self.judul)</code> metode yang digunakan untuk menambahkan <i>widget</i> ke dalam <i>layout</i> agar berada di dalam jendela aplikasi.

19	<code>Form_layout</code> merupakan sebuah variabel yang menyimpan <code>QGridLayout</code> yang merupakan jenis <i>layout manager</i> digunakan untuk mengatur tata letak <i>widget</i> dalam <i>grid</i> (tabel).
21-23	<code>Self.input_nama</code> merupakan sebuah variabel yang menyimpan <code>QLineEdit(self)</code> dimana ini adalah sebuah widget yang digunakan untuk menginput teks dalam bentuk satu baris lalu <code>setPlaceholderText("Nama Barang")</code> digunakan sebagai <i>placeholder</i> yang akan memberi petunjuk kepada pengguna dengan argumen "Nama Barang" untuk memberi tahu apa yang harus dimasukkan, lalu diatur jenis <i>font</i> "open sans" menggunakan <code>setFont</code> dengan ukuran 10.
24	Lalu menggunakan metode <code>setStyleSheet()</code> untuk menambahkan gaya ke <code>QLineEdit</code> , dimana gaya yang ditambahkan terdapat padding: 10px; untuk menambahkan ruang pada tiap sisi kolom, border: 1px solid #5865F2; untuk mengatur border (garis tepi) dengan warna biru, lalu border-radius: 4px; untuk memberikan bentuk sedikit melengkung pada border dengan radius 4px.
25	Pada baris ini merupakan sebuah <i>widget</i> <code>QLabel</code> yang ditambahkan ke dalam <code>form_layout</code> . <code>QLabel</code> ini berisikan argumen "Nama Barang" yang akan ditampilkan dalam teks pada posisi baris ke-0 dan Kolom ke-0 di dalam <i>grid</i> .
26	Pada baris ini merupakan sebuah <i>widget</i> <code>self.input_nama</code> yang ditambahkan ke dalam <code>form_layout</code> , dimana <i>widget</i> ini merupakan sebuah tempat untuk pengguna memasukkan nama barang yang akan ditampilkan pada baris pertama dan kolom kedua pada <i>grid layout</i> .
28-30	<code>Self.input_harga</code> merupakan sebuah variabel menyimpan <code>QLineEdit(self)</code> dimana ini adalah sebuah widget yang digunakan untuk menginput teks dalam bentuk satu baris, lalu <code>setPlaceholderText("Harga Barang")</code> digunakan sebagai <i>placeholder</i> yang akan memberi petunjuk kepada pengguna dengan argumen "Harga Barang" untuk menentukan berapa harga barang yang harus dimasukkan (<i>input</i>), lalu di atur jenis <i>font</i> "open sans" menggunakan <code>setFont</code> dengan ukuran 10.
31	Pada baris ini merupakan sebuah penggunaan metode <code>setStyleSheet()</code> untuk menambahkan gaya ke <code>QLineEdit</code> , dimana gaya yang ditambahkan terdapat padding: 10px; untuk menambahkan ruang pada tiap sisi kolom, border: 1px solid #5865F2; untuk mengatur border (garis tepi) dengan warna biru, lalu border-radius: 4px; untuk memberikan bentuk sedikit melengkung pada border dengan radius 4px.
32	Pada baris ini merupakan sebuah <i>widget</i> <code>QLabel</code> yang digunakan untuk menampilkan teks " Harga Barang :" pada kolom input memasukkan harga barang yang akan ditampilkan pada baris kedua dan kolom pertama.

33	Pada baris ini merupakan sebuah <code>widget self.input_harga</code> tempat menginput harga barang yang ditambahkan ke dalam <code>layout</code> , lalu ditampilkan pada baris kedua dan kolom kedua pada <code>grid layout</code> .
37	Pada baris ini digunakan untuk menambahkan <code>layout</code> yang skalanya lebih kecil ke <code>layout</code> dengan skala yang lebih besar agar tata letak lebih terstruktur.
39-40	Menggunakan <code>widget QPushButton</code> dari PyQt untuk membuat tampilan tombol yang bisa di klik oleh pengguna, dengan argumen “Tambah Data” yang digunakan untuk memberi petunjuk kepada pengguna untuk menambahkan data. lalu di atur jenis font “open sans” menggunakan <code>setFont</code> dengan ukuran 12.
41-47	<code>SetStyleSheet("...")</code> digunakan untuk mengatur <i>styling</i> di aplikasi GUI untuk pengaturan latar belakang yang berwarna biru muda, dengan warna teks di dalam tombol berwarna putih, jarak untuk tiap sisi teks di dalam tombolnya sendiri adalah 10px, terdapat border tanpa garis batas di sekitar tombol dengan sudut-sudut dari sekitar tombol ini 5px
48-49	Pada baris ini berfungsi untuk menampilkan tombol antarmuka pengguna, lalu memancarkan sinyal ketika tombol akan ditekan. Ketika tombol ditekan maka akan dihubungkan antara sinyal tombol dengan fungsi <code>tambah_data</code> akan berjalan. <code>addWidget(self.button_tambah)</code> digunakan untuk menambah tombol ke dalam layout untuk mengatur posisi dan penataan tombol antarmuka
51-52	Pada bagian ini, tombol “Update Data” dibuat menggunakan <code>QPushButton</code> dengan label “Update Data”. Font tombol diatur menggunakan <code>setFont</code> untuk menggunakan font "Open Sans" dengan ukuran 12, sehingga tombol terlihat lebih jelas dan modern.
53-59	<code>SetStyleSheet("...")</code> digunakan untuk mengatur <i>styling</i> di aplikasi GUI untuk pengaturan latar belakang yang berwarna biru muda, dengan warna teks di dalam tombol berwarna putih, jarak untuk tiap sisi teks di dalam tombolnya sendiri adalah 10px, terdapat border tanpa garis batas di sekitar tombol dengan sudut-sudut dari sekitar tombol ini 5px
60-61	Pada baris ini berfungsi untuk menampilkan tombol antarmuka pengguna, lalu memancarkan sinyal ketika tombol akan ditekan. Ketika tombol ditekan maka akan dihubungkan antara sinyal tombol dengan fungsi <code>update_data</code> akan berjalan. <code>addWidget(self.button_tambah)</code> digunakan untuk menambah tombol ke dalam layout untuk mengatur posisi dan penataan tombol antarmuka
63-64	Pada bagian ini, tombol “Hapus Data” dibuat menggunakan <code>QPushButton</code> dengan label “Hapus Data”. Font tombol diatur menggunakan <code>setFont</code> untuk menggunakan font "Open Sans" dengan ukuran 12, sehingga tombol terlihat lebih jelas dan modern.
65-71	<code>SetStyleSheet("...")</code> digunakan untuk mengatur <i>styling</i> di

	aplikasi GUI untuk pengaturan latar belakang yang berwarna biru muda, dengan warna teks di dalam tombol berwarna putih, jarak untuk tiap sisi teks di dalam tombolnya sendiri adalah 10px, terdapat border tanpa garis batas di sekitar tombol dengan sudut-sudut dari sekitar tombol ini 5px
72-73	Pada baris ini berfungsi untuk menampilkan tombol antarmuka pengguna, lalu memancarkan sinyal ketika tombol akan ditekan. Ketika tombol ditekan maka akan dihubungkan antara sinyal tombol dengan fungsi <code>Hapus_data</code> akan berjalan. <code>addWidget(self.button_tambah)</code> digunakan untuk menambah tombol ke dalam layout untuk mengatur posisi dan penataan tombol antarmuka
75-76	Pada bagian ini, tombol “Reset Data” dibuat menggunakan <code>QPushButton</code> dengan label “Reset Data”. Font tombol diatur menggunakan <code>setFont</code> untuk menggunakan font "Open Sans" dengan ukuran 12, sehingga tombol terlihat lebih jelas dan modern
77-83	<code>setStyleSheet("...")</code> digunakan untuk mengatur <i>styling</i> di aplikasi GUI untuk pengaturan latar belakang yang berwarna biru muda, dengan warna teks di dalam tombol berwarna putih, jarak untuk tiap sisi teks di dalam tombolnya sendiri adalah 10px, terdapat border tanpa garis batas di sekitar tombol dengan sudut-sudut dari sekitar tombol ini 5px
84-85	Pada baris ini berfungsi untuk menampilkan tombol antarmuka pengguna, lalu memancarkan sinyal ketika tombol akan ditekan. Ketika tombol ditekan maka akan dihubungkan antara sinyal tombol dengan fungsi <code>Reset_data</code> akan berjalan. <code>addWidget(self.button_tambah)</code> digunakan untuk menambah tombol ke dalam layout untuk mengatur posisi dan penataan tombol antarmuka
87	Pada baris ini digunakan untuk menambahkan layout <code>button_layout</code> ke dalam <i>layout</i> utama <code>main_layout</code> .
89-90	<code>self.table</code> merupakan sebuah variabel yang berisi <code>widget QTableWidget</code> untuk menampilkan dan mengelola data yang ditampilkan dalam bentuk tabel. Dan <code>setColumnCount(3)</code> merupakan metode untuk mengatur jumlah kolom di dalam <code>widget QTableWidget</code> agar terstruktur
91-92	Baris ini digunakan untuk <i>header</i> kolom tabel secara horizontal. “ No ” kolom pertama untuk nomor urut. “ Nama Barang ” kolom kedua, untuk nama barang. “ Harga Barang ” kolom ketiga, untuk harga barang. Mengatur <i>font</i> pada tabel, termasuk isinya. Jenis hurufnya “Open Sans” dan ukuran <i>font</i> 12
93-98	Kode ini menggunakan <i>stylesheet</i> berbasis CSS untuk mempercantik tampilan tabel. Gaya diterapkan pada tabel <code>QTableWidget</code> . Memberikan garis tepi dengan ketebalan 1 piksel, warna biru (#5865F2). Membuat sudut-sudut tabel melengkung dengan radius 5 piksel. <code>background-color: (#D3D3D3)</code> untuk mengatur warna latar belakang tabel menjadi abu-abu muda.

99-105	Kode ini digunakan untuk mengatur tampilan tabel <code>self.table</code> . Tabel diberikan border tipis berwarna biru (#5865F2) dan sudut melengkung 5 piksel, sehingga tampilannya lebih modern dan bersih. Latar belakang tabel juga diberi warna abu-abu terang (#D3D3D3).
106-109	Kode ini mengatur tampilan tabel agar kolomnya dapat otomatis menyesuaikan lebar, dapat digulir pixel per pixel, dan memilih seluruh baris saat di-klik.
111-112	Kode tersebut menambahkan <code>widget</code> tabel ke dalam <code>layout</code> utama dan mengatur layout tersebut sebagai <code>layout</code> dari <code>widget</code> saat ini.
114-116	Fungsi <code>tambah_data</code> mengambil <code>input</code> nama dan harga barang dari pengguna, lalu menghapus spasi di awal dan akhir <code>string</code> tersebut.
118-120	Kode tersebut memeriksa apakah nama dan harga barang diisi, dan jika tidak, menampilkan pesan peringatan kepada pengguna.
122-129	Kode tersebut menangani <code>input</code> harga barang, memastikan bahwa nilai yang dimasukkan adalah angka positif, dan menampilkan pesan kesalahan jika input tidak valid.
131-136	Kode tersebut menambahkan baris baru ke dalam tabel dengan nomor urut, nama barang, dan harga dalam Rupiah.
138-140	Kode tersebut menghapus isi dari dua input, yaitu <code>input_nama</code> dan <code>input_harga</code> , serta memperbarui nomor baris yang ditampilkan.
142-146	Fungsi <code>update_data</code> akan memeriksa apakah terdapat baris yang dipilih dalam tabel, dan jika tidak, maka program akan menampilkan peringatan untuk meminta pengguna memilih baris yang ingin diperbarui.
148-149	Kode tersebut mengambil <code>input</code> pengguna untuk nama, dan harga barang, kemudian menghapus spasi di awal dan di akhir <code>input</code> tersebut dengan menggunakan metode <code>strip()</code> .
151-153	Kode memeriksa apakah variabel <code>nama_barang</code> dan <code>harga_barang</code> kosong, jika iya, maka program akan menampilkan peringatan bahwa keduanya harus diisi.
155-162	Kode berfungsi untuk memvalidasi <code>input</code> harga barang, memastikan <code>input</code> -an pengguna berupa bilangan positif. Jika tidak, maka program akan menampilkan peringatan.
164	<code>int(harga)</code> : Mengkonversi nilai harga menjadi <code>integer</code> . Ini untuk hapus desimal jika ada <code>, f"Rp {int(harga)}:, "</code> : Menggunakan f-string untuk memformat harga. Tanda <code>:</code> , di dalam f-string akan memformat angka dengan pemisah ribuan (misalnya, 10000 menjadi 1,000,000), <code>replace(", ", ".")</code> : Mengganti koma <code>(,)</code> dengan titik <code>(.)</code> untuk menyesuaikan dengan format uang negara Indonesia, sehingga hasil akhirnya menjadi "Rp 1.000.000"

165-166	<code>self.table.setItem(selected_row, 1, QTableWidgetItem(nama_barang)):</code> Menambahkan item <code>nama_barang</code> ke tabel pada baris yang dipilih (<code>selected_row</code>) dan kolom ke-1, <code>self.table.setItem(selected_row, 2, QTableWidgetItem(formatted_harga))</code> : Menambahkan item <code>formatted_harga</code> ke tabel pada baris yang dipilih dan kolom ke-2. <code>QTableWidgetItem</code> adalah objek yang digunakan untuk menampilkan data dalam sel tabel.
168-175	<code>selected_row = self.table.currentRow()</code> : Mengambil indeks baris yang saat ini dipilih di tabel. <code>if selected_row == -1:</code> : Memeriksa apakah tidak ada baris yang dipilih. Jika tidak ada, akan muncul kotak peringatan, <code>QMessageBox.warning(...)</code> : Menampilkan kotak dialog peringatan dengan pesan untuk memilih baris yang ingin dihapus, <code>self.table.removeRow(selected_row)</code> : Menghapus baris yang dipilih dari tabel. <code>self.update_row_numbers()</code> : Memanggil metode <code>update_row_numbers</code> untuk memperbarui nomor baris setelah penghapusan.
180-182	<code>for row in range(self.table.rowCount()):</code> : Melakukan iterasi melalui semua baris yang ada di tabel. <code>self.table.setItem(row, 0, QTableWidgetItem(str(row + 1)))</code> : Mengatur nomor baris di kolom ke-0. Nomor baris dimulai dari 1, sehingga <code>row + 1</code> digunakan untuk menampilkan nomor yang benar.
184-188	<code>if __name__ == "__main__":</code> : Memastikan bahwa kode di dalam blok ini hanya dieksekusi jika file ini dijalankan sebagai program Utama <code>app = QApplication(sys.argv)</code> : Membuat instance aplikasi <code>Qt.window = BarangApp()</code> : Membuat <i>instance</i> dari kelas <code>BarangApp</code> , yang merupakan jendela utama aplikasi <code>window.show()</code> : Menampilkan jendela aplikasi.

Kesimpulan

Kesimpulan dari penggunaan PyQt5 dari praktikum kali ini, bahwa penggunaan pyqt membuat tampilan lebih kreatif sehingga lebih modern dari UI/UX tetapi PyQt5 lumayan sulit juga bagi kami untuk membuat *layout* dan dari segi tampilan PyQt jauh lebih unggul dibandingkan Tkinter. Untuk *Layout* itu sendiri memungkinkan program untuk menyesuaikan tampilan dengan ukuran *window* aplikasi. Ketika pengguna mengubah ukurannya, elemen-elemen dalam aplikasi itu akan tetap tersusun dengan rapi tanpa adanya terhambur pada isi *layout* itu jadi mempermudah dalam mengatur tampilan *interface*-nya.

Saran

Asisten laboratorium sudah baik dan lebih jelas dalam menjelaskan materi saat praktikum, dan adapun juga pada saat memberikan tutorial cara menggunakan PyQt5 sudah lumayan baik sehingga kami dapat memahami cara penggunaannya, walau belum sepenuhnya.

Daftar Pustaka

RiverbankComputing Limited. (19 juli 2024). *PyQt5*. :

<https://pypi.org/project/PyQt5/>

Komputer, U. S. & T. (2018). Ensiklopedia Dunia: PyQt. Program Kelas Karyawan (Kuliah Online / Blended). :

<https://p2k.stekom.ac.id/ensiklopedia/PyQt>

Leodanis p.r .(n.d.). *Python and PyQt: Building Functional GUI Layouts*. :

[Tata Letak PyQt: Buat Aplikasi GUI yang Terlihat Profesional – Python Nyata](#)

TutorialsPoint.n.d.PyQt Basic Widgets :

https://www.tutorialspoint.com/pyqt/pyqt_basic_widgets.htm