|  |  |
| --- | --- |
| **LAPORAN PRAKTIK PEMROGRAMAN**  **UJIAN AKHIR SEMESTER**  **(Aplikasi CRUD Jadwal Konsultasi Pengaduan)** | Mobile Programming, 2024  NIM.  (2021 53 167)  Nama  (Annisa Putri Hapsari) |



|  |
| --- |
| Aplikasi CRUD Aplikasi CRUD Jadwal Konsultasi Pengaduan |
| 1. Source Code |
| # memasukkan library flet ke aplikasi  # import flet as ft  import flet  from flet import \*  import mysql.connector  import datetime  # buat koneksi ke database SQL  koneksi\_db = mysql.connector.connect(host = "localhost", user = "root", password = "", database = "mp\_uas")  cursor = koneksi\_db.cursor()  class FormClient(UserControl):      def build(client) :          # buat variabel inputan          client.inputan\_id\_client = TextField(visible = False, expand = True)          client.inputan\_nip\_client = TextField(label = "NIP Client", hint\_text = "masukkan nip client ... ", expand = True, input\_filter = InputFilter(allow = True, regex\_string = r"[0-9]", replacement\_string = ""))          client.inputan\_nama\_client = TextField(label = "Nama Client", hint\_text = "masukkan nama client ... ", expand = True)          client.inputan\_jk\_client = Dropdown(label = "Jenis Kelamin Client", hint\_text = "pilih jenis kelamin client ... ", expand = True,              options = [                  dropdown.Option("Laki-Laki"),                  dropdown.Option("Perempuan"),              ],          )          client.inputan\_asal\_opd = TextField(label = "Asal OPD Client", hint\_text = "masukkan asal opd client ... ", expand = True)          client.inputan\_no\_telepon = TextField(label = "Nomor Telepon Client", hint\_text = "masukkan nomor telepon client ... ", expand = True, input\_filter = InputFilter(allow = True, regex\_string = r"[0-9]", replacement\_string = ""))          client.inputan\_topik\_konsultasi = TextField(label = "Topik Konsultasi", hint\_text = "masukkan topik konsultasi ... ", expand = True)          client.snack\_bar\_berhasil = SnackBar( Text("Operasi berhasil"), bgcolor = "#FDA403")            # memuat tabel data          def tampil\_data(e):              # Merefresh halaman & menampilkan notif              client.data\_client.rows.clear()              cursor.execute("SELECT \* FROM tb\_client ORDER BY id\_client ASC")              result = cursor.fetchall()              # menampilkan ulang data              columns = [column[0] for column in cursor.description]              rows = [dict(zip(columns,row)) for row in result]              for index, row in enumerate(rows, start = 1):                  client.data\_client.rows.append(                      DataRow(                          cells = [                              DataCell(Text(str(index))),                              DataCell(Text(row['id\_client'])),                              DataCell(Text(row['nip\_client'])),                              DataCell(Text(row['nama\_client'])),                              DataCell(Text(row['jk\_client'] + ', Asal OPD : ' + row['asal\_opd'] + ', Telp : ' + row['no\_telepon'] )),                              DataCell(Text(row['topik\_konsultasi'])),                              DataCell(                                  Row([                                      IconButton("delete", icon\_color = "red", data = row, on\_click = hapus\_client),                                      IconButton("create", icon\_color = "grey", data = row, on\_click = tampil\_dialog\_ubah),                                  ])                              ),                          ]                          )                      )          # memuat pencarian data          def inputsearch(e):              search\_term = client.nameinput.value.lower()              cursor.execute("SELECT \* FROM tb\_client ORDER BY id\_client ASC")              result = cursor.fetchall()              rows = [dict(zip([column[0] for column in cursor.description], row)) for row in result]                def match(row):                  return (search\_term in row['id\_client'].lower() or                          search\_term in row['nip\_client'].lower() or                          search\_term in row['nama\_client'].lower() or                          search\_term in row['jk\_client'].lower() or                          search\_term in row['asal\_opd'].lower() or                          search\_term in row['no\_telepon'].lower() or                          search\_term in row['topik\_konsultasi'].lower())                myfilter = list(filter(match, rows))              #print("you find ", myfilter)              client.data\_client.rows.clear()              if not client.nameinput.value == "":                  if len(myfilter) > 0:                      client.datanotfound.visible = False                      for x in myfilter:                          client.data\_client.rows.append(                              DataRow(                                  cells = [                                      DataCell(Text(str(rows.index(x) + 1))),                                      DataCell(Text(x['id\_client'])),                                      DataCell(Text(x['nip\_client'])),                                      DataCell(Text(x['nama\_client'])),                                      DataCell(Text(f"{x['jk\_client'] + ', Asal OPD : ' + x['asal\_opd'] + ', Telp : ' + x['no\_telepon']}")),                                      DataCell(Text(x['topik\_konsultasi'])),                                      DataCell(                                          Row([                                              IconButton("delete", icon\_color = "red", data = x, on\_click = hapus\_client),                                              IconButton("create", icon\_color = "grey", data = x, on\_click = tampil\_dialog\_ubah),                                          ])                                      ),                                  ]                              )                          )                      client.update()                  else:                      #print("Data tidak ditemukan")                      client.datanotfound.visible = True                      client.update()              else:                  tampil\_data(e)                  client.datanotfound.visible = False                  client.update()          # fungsi menampilkan dialog form entri          def tampil\_dialog(e):              client.inputan\_id\_client.value = ''              client.inputan\_nip\_client.value = ''              client.inputan\_nama\_client.value = ''              client.inputan\_jk\_client.value = ''              client.inputan\_asal\_opd.value = ''              client.inputan\_no\_telepon.value = ''              client.inputan\_topik\_konsultasi.value = ''              client.dialog.open = True              client.update()          def tampil\_dialog\_ubah(e):              client.inputan\_id\_client.value = e.control.data['id\_client']              client.inputan\_nip\_client.value = e.control.data['nip\_client']              client.inputan\_nama\_client.value = e.control.data['nama\_client']              client.inputan\_jk\_client.value = e.control.data['jk\_client']              client.inputan\_asal\_opd.value = e.control.data['asal\_opd']              client.inputan\_no\_telepon.value = e.control.data['no\_telepon']              client.inputan\_topik\_konsultasi.value = e.control.data['topik\_konsultasi']              client.dialog.open = True              client.update()          # fungsi simpan data          def simpan\_client(e):              try:                  if (client.inputan\_id\_client.value == '') :                      # Mendapatkan id\_client terakhir dari tabel client                      cursor.execute("SELECT id\_client FROM tb\_client ORDER BY id\_client DESC LIMIT 1")                      result = cursor.fetchone()                      # Memeriksa apakah id\_client terakhir sudah ada atau belum                      if result and result[0]:                          # Jika sudah ada, kita tambahkan 1 ke id\_client terakhir                          last\_id = result[0]                          id\_number = int(last\_id.split('-')[-1]) + 1                      else:                          # Jika belum ada, kita mulai dari 1                          id\_number = 1                      # Membuat angka dengan panjang 3 digit                      new\_id\_number = str(id\_number).zfill(3)                      # Membuat id\_client baru dengan format yang diinginkan                      new\_id\_client = f"Client-{new\_id\_number}"                      sql = "INSERT INTO tb\_client (id\_client, nip\_client, nama\_client, jk\_client, asal\_opd, no\_telepon, topik\_konsultasi) VALUES(%s, %s, %s, %s, %s, %s, %s)"                      val = (new\_id\_client, client.inputan\_nip\_client.value, client.inputan\_nama\_client.value, client.inputan\_jk\_client.value, client.inputan\_asal\_opd.value, client.inputan\_no\_telepon.value, client.inputan\_topik\_konsultasi.value)                  else :                      sql = "UPDATE tb\_client SET nip\_client = %s, nama\_client = %s, jk\_client = %s, asal\_opd = %s, no\_telepon = %s, topik\_konsultasi = %s WHERE id\_client = %s"                      val = (client.inputan\_nip\_client.value, client.inputan\_nama\_client.value, client.inputan\_jk\_client.value, client.inputan\_asal\_opd.value, client.inputan\_no\_telepon.value, client.inputan\_topik\_konsultasi.value, client.inputan\_id\_client.value)                    cursor.execute(sql, val)                  koneksi\_db.commit()                  print(cursor.rowcount, "Data di simpan!")                  client.nameinput.value=''                  tampil\_data(e)                  client.dialog.open = False                  client.snack\_bar\_berhasil.open = True                  client.update()              except Exception as e:                  print(e)                  print("Ada yang error!")          # fungsi hapus data          def hapus\_client(e):              try:                  sql = "DELETE FROM tb\_client WHERE id\_client = %s"                  val = (e.control.data['id\_client'],)                  cursor.execute(sql, val)                  koneksi\_db.commit()                  print(cursor.rowcount, "data di hapus!")                  client.data\_client.rows.clear()                  client.nameinput.value = ''                    tampil\_data(e)                  client.dialog.open = False                  client.snack\_bar\_berhasil.open = True                  client.update()              except Exception as e:                  print(e)                  print("Ada yang error!")          # menampilkan semua data ke dalam tabel          client.data\_client = DataTable(              columns = [                  DataColumn(Text("NO")),                  DataColumn(Text("ID Client")),                  DataColumn(Text("NIP Client")),                  DataColumn(Text("Nama Client")),                  DataColumn(Text("Informasi Client")),                  DataColumn(Text("Topik Konsultasi")),                  DataColumn(Text("Opsi")),              ],          )          cursor.execute("SELECT \* FROM tb\_client ORDER BY id\_client ASC")          result = cursor.fetchall()          columns = [column[0] for column in cursor.description]          rows = [dict(zip(columns,row)) for row in result]          for index, row in enumerate(rows, start = 1):              client.data\_client.rows.append(                  DataRow(                      cells = [                          DataCell(Text(str(index))),                          DataCell(Text(row['id\_client'])),                          DataCell(Text(row['nip\_client'])),                          DataCell(Text(row['nama\_client'])),                          DataCell(Text(row['jk\_client'] + ', Asal OPD : ' + row['asal\_opd'] + ', Telp : ' + row['no\_telepon'] )),                          DataCell(Text(row['topik\_konsultasi'])),                          DataCell(                              Row([                                  IconButton("delete", icon\_color = "red", data = row, on\_click = hapus\_client),                                  IconButton("create", icon\_color = "grey", data = row, on\_click = tampil\_dialog\_ubah),                              ])                          ),                      ]                  )              )          # buat variabel utk pencarian data          client.nameinput = TextField(              label = "Cari data",              hint\_text = "masukkan data yang ingin dicari ... ",              on\_change = inputsearch          )          client.datanotfound = Text(              "Data yang dicari tidak ada...",              weight = "bold",              size = 18          )          client.datanotfound.visible = False            # buat variabel utk layout data rekapan          client.layout\_data = Column()          # buat form dialog untuk form entri data          client.dialog = BottomSheet(              Container(                  Column(                      [                          Text("Form Entri Client", weight = FontWeight.BOLD),                          Row([ client.inputan\_id\_client ]),                          Row([ client.inputan\_nip\_client ]),                          Row([ client.inputan\_nama\_client ]),                          Row([ client.inputan\_jk\_client ]),                          Row([ client.inputan\_asal\_opd ]),                          Row([ client.inputan\_no\_telepon ]),                          Row([ client.inputan\_topik\_konsultasi ]),                          Row([                              #tombol tambah data                              ElevatedButton(                                  "Simpan Data",                                      icon = "SAVE\_AS",                                      icon\_color = "white",                                      color = "white",                                      bgcolor = "#2C4E80",                                      width = 280,                                      height = 50,                                      on\_click = simpan\_client,                                  )                          ]),                      ],                      horizontal\_alignment = CrossAxisAlignment.CENTER,                      height = 500,                      scroll = ScrollMode.ALWAYS                      #tight = True,                  ),                  padding = 40,                  width = 378,                  height = 500              ),              open = False,          )          return Column(              controls = [                  Row([ElevatedButton("Tambah Data", icon = icons.ADD, icon\_color = "white", width = 335, color = "white", bgcolor = "#2C4E80", on\_click = tampil\_dialog)], alignment = MainAxisAlignment.END),                  client.nameinput,                  Row([client.data\_client], scroll = ScrollMode.ALWAYS),                  client.datanotfound,                  client.dialog,                  client.snack\_bar\_berhasil              ],          )  class FormKonsultan(UserControl):      def build(konsultan) :          # buat variabel inputan          konsultan.inputan\_id\_konsultan = TextField(visible = False, expand = True)          konsultan.inputan\_nip\_konsultan = TextField(label = "NIP Konsultan", hint\_text = "masukkan nip konsultan ... ", expand = True, input\_filter = InputFilter(allow = True, regex\_string = r"[0-9]", replacement\_string = ""))          konsultan.inputan\_nama\_konsultan = TextField(label = "Nama Konsultan", hint\_text = "masukkan nama konsultan ... ", expand = True)          konsultan.inputan\_jk\_konsultan = Dropdown(label = "Jenis Kelamin Konsultan", hint\_text = "pilih jenis kelamin konsultan ... ", expand = True,              options = [                  dropdown.Option("Laki-Laki"),                  dropdown.Option("Perempuan"),              ],          )          konsultan.snack\_bar\_berhasil = SnackBar( Text("Operasi berhasil"), bgcolor = "#FDA403")            # memuat tabel data          def tampil\_data(e):              # Merefresh halaman & menampilkan notif              konsultan.data\_konsultan.rows.clear()              cursor.execute("SELECT \* FROM tb\_konsultan ORDER BY id\_konsultan ASC")              result = cursor.fetchall()              # menampilkan ulang data              columns = [column[0] for column in cursor.description]              rows = [dict(zip(columns,row)) for row in result]              for index, row in enumerate(rows, start = 1):                  konsultan.data\_konsultan.rows.append(                      DataRow(                          cells = [                              DataCell(Text(str(index))),                              DataCell(Text(row['id\_konsultan'])),                              DataCell(Text(row['nip\_konsultan'])),                              DataCell(Text(row['nama\_konsultan'])),                              DataCell(Text(row['jk\_konsultan'])),                              DataCell(                                  Row([                                      IconButton("delete", icon\_color = "red", data = row, on\_click = hapus\_konsultan),                                      IconButton("create", icon\_color = "grey", data = row, on\_click = tampil\_dialog\_ubah),                                  ])                              ),                          ]                          )                      )            # memuat pencarian data          def inputsearch(e):              search\_term = konsultan.nameinput.value.lower()              cursor.execute("SELECT \* FROM tb\_konsultan ORDER BY id\_konsultan ASC")              result = cursor.fetchall()              rows = [dict(zip([column[0] for column in cursor.description], row)) for row in result]                def match(row):                  return (search\_term in row['id\_konsultan'].lower() or                          search\_term in row['nip\_konsultan'].lower() or                          search\_term in row['nama\_konsultan'].lower() or                          search\_term in row['jk\_konsultan'].lower())                myfilter = list(filter(match, rows))              #print("you find ", myfilter)              konsultan.data\_konsultan.rows.clear()              if not konsultan.nameinput.value == "":                  if len(myfilter) > 0:                      konsultan.datanotfound.visible = False                      for x in myfilter:                          konsultan.data\_konsultan.rows.append(                              DataRow(                                  cells = [                                      DataCell(Text(str(rows.index(x) + 1))),                                      DataCell(Text(x['id\_konsultan'])),                                      DataCell(Text(x['nip\_konsultan'])),                                      DataCell(Text(x['nama\_konsultan'])),                                      DataCell(Text(x['jk\_konsultan'])),                                      DataCell(                                          Row([                                              IconButton("delete", icon\_color = "red", data = x, on\_click = hapus\_konsultan),                                              IconButton("create", icon\_color = "grey", data = x, on\_click = tampil\_dialog\_ubah),                                          ])                                      ),                                  ]                              )                          )                      konsultan.update()                  else:                      #print("Data tidak ditemukan")                      konsultan.datanotfound.visible = True                      konsultan.update()              else:                  tampil\_data(e)                  konsultan.datanotfound.visible = False                  konsultan.update()          # fungsi menampilkan dialog form entri          def tampil\_dialog(e):              konsultan.inputan\_id\_konsultan.value = ''              konsultan.inputan\_nip\_konsultan.value = ''              konsultan.inputan\_nama\_konsultan.value = ''              konsultan.inputan\_jk\_konsultan.value = ''              konsultan.dialog.open = True              konsultan.update()          def tampil\_dialog\_ubah(e):              konsultan.inputan\_id\_konsultan.value = e.control.data['id\_konsultan']              konsultan.inputan\_nip\_konsultan.value = e.control.data['nip\_konsultan']              konsultan.inputan\_nama\_konsultan.value = e.control.data['nama\_konsultan']              konsultan.inputan\_jk\_konsultan.value = e.control.data['jk\_konsultan']              konsultan.dialog.open = True              konsultan.update()          # fungsi simpan data          def simpan\_konsultan(e):              try:                  if (konsultan.inputan\_id\_konsultan.value == '') :                      # Mendapatkan id\_konsultan terakhir dari tabel konsultan                      cursor.execute("SELECT id\_konsultan FROM tb\_konsultan ORDER BY id\_konsultan DESC LIMIT 1")                      result = cursor.fetchone()                      # Memeriksa apakah id\_konsultan terakhir sudah ada atau belum                      if result and result[0]:                          # Jika sudah ada, kita tambahkan 1 ke id\_konsultan terakhir                          last\_id = result[0]                          id\_number = int(last\_id.split('-')[-1]) + 1                      else:                          # Jika belum ada, kita mulai dari 1                          id\_number = 1                      # Membuat angka dengan panjang 3 digit                      new\_id\_number = str(id\_number).zfill(3)                      # Membuat id\_client baru dengan format yang diinginkan                      new\_id\_konsultan = f"Konsultan-{new\_id\_number}"                      sql = "INSERT INTO tb\_konsultan (id\_konsultan, nip\_konsultan, nama\_konsultan, jk\_konsultan) VALUES(%s, %s, %s, %s)"                      val = (new\_id\_konsultan, konsultan.inputan\_nip\_konsultan.value, konsultan.inputan\_nama\_konsultan.value, konsultan.inputan\_jk\_konsultan.value)                  else :                      sql = "UPDATE tb\_konsultan SET nip\_konsultan = %s, nama\_konsultan = %s, jk\_konsultan = %s WHERE id\_konsultan = %s"                      val = (konsultan.inputan\_nip\_konsultan.value, konsultan.inputan\_nama\_konsultan.value, konsultan.inputan\_jk\_konsultan.value, konsultan.inputan\_id\_konsultan.value)                    cursor.execute(sql, val)                  koneksi\_db.commit()                  print(cursor.rowcount, "Data di simpan!")                  konsultan.nameinput.value = ''                  tampil\_data(e)                  konsultan.dialog.open = False                  konsultan.snack\_bar\_berhasil.open = True                  konsultan.update()              except Exception as e:                  print(e)                  print("Ada yang error!")          # fungsi hapus data          def hapus\_konsultan(e):              try:                  sql = "DELETE FROM tb\_konsultan WHERE id\_konsultan = %s"                  val = (e.control.data['id\_konsultan'],)                  cursor.execute(sql, val)                  koneksi\_db.commit()                  print(cursor.rowcount, "data di hapus!")                  konsultan.data\_konsultan.rows.clear()                  konsultan.nameinput.value = ''                    tampil\_data(e)                  konsultan.dialog.open = False                  konsultan.snack\_bar\_berhasil.open = True                  konsultan.update()              except Exception as e:                  print(e)                  print("Ada yang error!")          # menampilkan semua data ke dalam tabel          konsultan.data\_konsultan = DataTable(              columns = [                  DataColumn(Text("NO")),                  DataColumn(Text("ID Konsultan")),                  DataColumn(Text("NIP Konsultan")),                  DataColumn(Text("Nama Konsultan")),                  DataColumn(Text("Jenis Kelamin Konsultan")),                  DataColumn(Text("Opsi")),              ],          )          cursor.execute("SELECT \* FROM tb\_konsultan ORDER BY id\_konsultan ASC")          result = cursor.fetchall()          columns = [column[0] for column in cursor.description]          rows = [dict(zip(columns,row)) for row in result]          for index, row in enumerate(rows, start = 1):              konsultan.data\_konsultan.rows.append(                  DataRow(                      cells = [                          DataCell(Text(str(index))),                          DataCell(Text(row['id\_konsultan'])),                          DataCell(Text(row['nip\_konsultan'])),                          DataCell(Text(row['nama\_konsultan'])),                          DataCell(Text(row['jk\_konsultan'])),                          DataCell(                              Row([                                  IconButton("delete", icon\_color = "red", data = row, on\_click = hapus\_konsultan),                                  IconButton("create", icon\_color = "grey", data = row, on\_click = tampil\_dialog\_ubah),                              ])                          ),                      ]                  )              )            # buat variabel utk pencarian data          konsultan.nameinput = TextField(              label = "Cari data",              hint\_text = "masukkan data yang ingin dicari ... ",              on\_change = inputsearch          )          konsultan.datanotfound = Text(              "Data yang dicari tidak ada...",              weight = "bold",              size = 18          )          konsultan.datanotfound.visible = False          # buat variabel utk layout data rekapan          konsultan.layout\_data = Column()          # buat form dialog untuk form entri data          konsultan.dialog = BottomSheet(              Container(                  Column(                      [                          Text("Form Entri Konsultan", weight = FontWeight.BOLD),                          Row([ konsultan.inputan\_id\_konsultan ]),                          Row([ konsultan.inputan\_nip\_konsultan ]),                          Row([ konsultan.inputan\_nama\_konsultan ]),                          Row([ konsultan.inputan\_jk\_konsultan ]),                          Row([                              #tombol tambah data                              ElevatedButton(                                  "Simpan Data",                                      icon = "SAVE\_AS",                                      icon\_color = "white",                                      color = "white",                                      bgcolor = "#2C4E80",                                      width = 280,                                      height = 50,                                      on\_click = simpan\_konsultan,                                  )                          ]),                      ],                      horizontal\_alignment = CrossAxisAlignment.CENTER,                      height = 500,                      scroll = ScrollMode.ALWAYS                      #tight = True,                  ),                  padding = 40,                  width = 378,                  height = 500              ),              open = False,          )            return Column(              controls = [                  Row([ElevatedButton("Tambah Data", icon = icons.ADD, icon\_color = "white", width = 335, color = "white", bgcolor = "#2C4E80", on\_click = tampil\_dialog)], alignment = MainAxisAlignment.END),                  konsultan.nameinput,                  Row([konsultan.data\_konsultan], scroll = ScrollMode.ALWAYS),                  konsultan.datanotfound,                  konsultan.dialog,                  konsultan.snack\_bar\_berhasil              ],          )    class FormJadwalKonsultasi(UserControl):      def build(jadwalkonsultasi) :          # buat variabel inputan          jadwalkonsultasi.inputan\_id\_jadwalkonsultasi = TextField(visible = False, expand = True)          jadwalkonsultasi.inputan\_tgl\_konsultasi = TextField(label = "Tanggal Konsultasi", hint\_text = "masukkan tgl. konsultasi ... ", expand = True, read\_only = True)          #buat fungsi utk inputan tanggal          def ubah\_tanggal(e):              jadwalkonsultasi.inputan\_tgl\_konsultasi.value = jadwalkonsultasi.opsi\_tanggal.value.date()              jadwalkonsultasi.update()          def opsi\_tanggal\_dismissed(e):              jadwalkonsultasi.update()          jadwalkonsultasi.opsi\_tanggal = DatePicker(              on\_change = ubah\_tanggal,              on\_dismiss = opsi\_tanggal\_dismissed,              first\_date = datetime.datetime(1945, 1, 1),              last\_date = datetime.datetime(2045, 1, 1),          )            # TimePicker untuk jam mulai          def ubah\_jam\_mulai(e):              jam\_mulai = str(time\_picker\_jam\_mulai.value)              jadwalkonsultasi.inputan\_jam\_mulai\_konsultasi.value = jam\_mulai[:5]              jadwalkonsultasi.update()          def dismissed\_jam\_mulai(e):              jadwalkonsultasi.update()          time\_picker\_jam\_mulai = TimePicker(              confirm\_text = "Konfirmasi",              cancel\_text = "Batal",              error\_invalid\_text = "Waktu di luar jangkauan",              help\_text = "Pilih waktu",              on\_change = ubah\_jam\_mulai,              on\_dismiss = dismissed\_jam\_mulai,          )          # TimePicker untuk jam selesai          def ubah\_jam\_selesai(e):              jam\_selesai = str(time\_picker\_jam\_selesai.value)              jadwalkonsultasi.inputan\_jam\_selesai\_konsultasi.value = jam\_selesai[:5]              jadwalkonsultasi.update()          def dismissed\_jam\_selesai(e):              jadwalkonsultasi.update()          time\_picker\_jam\_selesai = TimePicker(              confirm\_text = "Konfirmasi",              error\_invalid\_text = "Waktu di luar jangkauan",              help\_text = "Pilih waktu",              on\_change = ubah\_jam\_selesai,              on\_dismiss = dismissed\_jam\_selesai,          )          jadwalkonsultasi.inputan\_jam\_mulai\_konsultasi = TextField(label = "Jam Mulai", hint\_text = "masukkan jam mulai ... ", expand = True, on\_change = lambda \_: time\_picker\_jam\_mulai.pick\_time())          jadwalkonsultasi.inputan\_jam\_selesai\_konsultasi = TextField(label = "Jam Selesai", hint\_text = "masukkan jam selesai ... ", expand = True, on\_change = lambda \_: time\_picker\_jam\_selesai.pick\_time())          jadwalkonsultasi.inputan\_ruang\_konsultasi = TextField(label = "Ruang Konsultasi", hint\_text = "masukkan ruang konsultasi ... ", expand = True)          cursor.execute("SELECT \* FROM tb\_client ORDER BY id\_client DESC")          jadwalkonsultasi.inputan\_client = Dropdown(label = "Nama Client", hint\_text = "pilih client ... ", expand = True,              options = [              dropdown.Option(row[0], row[1] + ' - ' + row[2]) for row in cursor.fetchall()],          )          cursor.execute("SELECT \* FROM tb\_konsultan ORDER BY id\_konsultan DESC")          jadwalkonsultasi.inputan\_konsultan = Dropdown(label = "Nama Konsultan", hint\_text = "pilih konsultan ... ", expand = True,              options = [              dropdown.Option(row[0], row[1] + ' - ' + row[2]) for row in cursor.fetchall()],          )          jadwalkonsultasi.snack\_bar\_berhasil = SnackBar( Text("Operasi berhasil"), bgcolor = "#FDA403")          # memuat tabel data          def tampil\_data(e):              # Merefresh halaman & menampilkan notif              jadwalkonsultasi.data\_jadwalkonsultasi.rows.clear()              cursor.execute("SELECT \* FROM tb\_jadwal\_konsultasi, tb\_konsultan, tb\_client WHERE tb\_jadwal\_konsultasi.id\_konsultan = tb\_konsultan.id\_konsultan AND tb\_jadwal\_konsultasi.id\_client = tb\_client.id\_client ORDER BY id\_jadwalkonsultasi ASC")              result = cursor.fetchall()              columns = [column[0] for column in cursor.description]              rows = [dict(zip(columns,row)) for row in result]              for index, row in enumerate(rows, start = 1):                  # Ambil nilai jam mulai dan jam selesai dari baris saat ini                  jam\_mulai = row['jam\_mulai']                  jam\_selesai = row['jam\_selesai']                    # Konversi jam mulai dan jam selesai ke objek datetime                  fmt = '%H:%M'                  dt\_mulai = datetime.datetime.strptime(jam\_mulai, fmt)                  dt\_selesai = datetime.datetime.strptime(jam\_selesai, fmt)                    # Hitung selisih waktu dalam menit                  selisih\_menit = (dt\_selesai - dt\_mulai).seconds // 60                  jadwalkonsultasi.data\_jadwalkonsultasi.rows.append(                      DataRow(                          cells = [                              DataCell(Text(str(index))),                              DataCell(Text(row['id\_jadwalkonsultasi'])),                              DataCell(Text(str(row['tgl\_konsultasi'].strftime("%d-%B-%Y")))),                              DataCell(Text(row['jam\_mulai'] + ' sampai ' + row['jam\_selesai'])),                              DataCell(Text(f"({selisih\_menit // 60} jam {selisih\_menit % 60} menit)")),                              DataCell(Text(row['ruang\_konsultasi'])),                              DataCell(Text(row['nama\_client'])),                              DataCell(Text(row['nama\_konsultan'])),                              DataCell(                                  Row([                                      IconButton("delete", icon\_color = "red", data = row, on\_click = hapus\_jadwalkonsultasi),                                      IconButton("create", icon\_color = "grey", data = row, on\_click = tampil\_dialog\_ubah),                                  ])                              ),                          ]                      )                  )          # memuat pencarian data          def inputsearch(e):              search\_term = jadwalkonsultasi.nameinput.value.lower()              cursor.execute("SELECT \* FROM tb\_jadwal\_konsultasi, tb\_konsultan, tb\_client WHERE tb\_jadwal\_konsultasi.id\_konsultan = tb\_konsultan.id\_konsultan AND tb\_jadwal\_konsultasi.id\_client = tb\_client.id\_client ORDER BY id\_jadwalkonsultasi ASC")              result = cursor.fetchall()              rows = [dict(zip([column[0] for column in cursor.description], row)) for row in result]                def match(row):                  # Ambil nilai jam mulai dan jam selesai dari baris saat ini                  jam\_mulai = row['jam\_mulai']                  jam\_selesai = row['jam\_selesai']                    # Konversi jam mulai dan jam selesai ke objek datetime                  fmt = '%H:%M'                  dt\_mulai = datetime.datetime.strptime(jam\_mulai, fmt)                  dt\_selesai = datetime.datetime.strptime(jam\_selesai, fmt)                    # Hitung selisih waktu dalam menit                  selisih\_menit = (dt\_selesai - dt\_mulai).seconds // 60                  return (search\_term in row['id\_jadwalkonsultasi'].lower() or                          search\_term in datetime.datetime.strptime(str(row['tgl\_konsultasi']),"%Y-%m-%d").strftime("%d-%m-%Y").lower() or                          search\_term in row['jam\_mulai'].lower() or                          search\_term in row['jam\_selesai'].lower() or                          search\_term in row['jam\_mulai'] + ' sampai ' + row['jam\_selesai'].lower() or                          search\_term in f"({selisih\_menit // 60} jam {selisih\_menit % 60} menit)".lower() or                          search\_term in row['ruang\_konsultasi'].lower() or                          search\_term in row['nama\_client'].lower() or                          search\_term in row['nama\_konsultan'].lower())                myfilter = list(filter(match, rows))              #print("you find ", myfilter)              jadwalkonsultasi.data\_jadwalkonsultasi.rows.clear()              if not jadwalkonsultasi.nameinput.value == "":                  if len(myfilter) > 0:                      jadwalkonsultasi.datanotfound.visible = False                      for x in myfilter:                          # Ambil nilai jam mulai dan jam selesai dari baris saat ini                          jam\_mulai = x['jam\_mulai']                          jam\_selesai = x['jam\_selesai']                            # Konversi jam mulai dan jam selesai ke objek datetime                          fmt = '%H:%M'                          dt\_mulai = datetime.datetime.strptime(jam\_mulai, fmt)                          dt\_selesai = datetime.datetime.strptime(jam\_selesai, fmt)                            # Hitung selisih waktu dalam menit                          selisih\_menit = (dt\_selesai - dt\_mulai).seconds // 60                          jadwalkonsultasi.data\_jadwalkonsultasi.rows.append(                              DataRow(                                  cells = [                                      DataCell(Text(str(rows.index(x) + 1))),                                      DataCell(Text(x['id\_jadwalkonsultasi'])),                                      DataCell(Text(datetime.datetime.strptime(str(x['tgl\_konsultasi']),"%Y-%m-%d").strftime("%d-%m-%Y"))),                                      DataCell(Text(f"{x['jam\_mulai']} sampai {x['jam\_selesai']}")),                                      DataCell(Text(f"({selisih\_menit // 60} jam {selisih\_menit % 60} menit)")),                                      DataCell(Text(x['ruang\_konsultasi'])),                                      DataCell(Text(x['nama\_client'])),                                      DataCell(Text(x['nama\_konsultan'])),                                      DataCell(                                          Row([                                              IconButton("delete", icon\_color = "red", data = x, on\_click = hapus\_jadwalkonsultasi),                                              IconButton("create", icon\_color = "grey", data = x, on\_click = tampil\_dialog\_ubah),                                          ])                                      ),                                  ]                              )                          )                      jadwalkonsultasi.update()                  else:                      #print("Data tidak ditemukan")                      jadwalkonsultasi.datanotfound.visible = True                      jadwalkonsultasi.update()              else:                  tampil\_data(e)                  jadwalkonsultasi.datanotfound.visible = False                  jadwalkonsultasi.update()          # fungsi menampilkan dialog form entri          def tampil\_dialog(e):              jadwalkonsultasi.inputan\_id\_jadwalkonsultasi.value = ''              jadwalkonsultasi.inputan\_tgl\_konsultasi.value = ''              jadwalkonsultasi.inputan\_jam\_mulai\_konsultasi.value = ''              jadwalkonsultasi.inputan\_jam\_selesai\_konsultasi.value = ''              jadwalkonsultasi.inputan\_ruang\_konsultasi.value = ''              jadwalkonsultasi.inputan\_client.value = ''              jadwalkonsultasi.inputan\_konsultan.value = ''              jadwalkonsultasi.dialog.open = True              jadwalkonsultasi.update()          def tampil\_dialog\_ubah(e):              jadwalkonsultasi.inputan\_id\_jadwalkonsultasi.value = e.control.data['id\_jadwalkonsultasi']              jadwalkonsultasi.inputan\_tgl\_konsultasi.value = e.control.data['tgl\_konsultasi']              jadwalkonsultasi.inputan\_jam\_mulai\_konsultasi.value = e.control.data['jam\_mulai'][:5]              jadwalkonsultasi.inputan\_jam\_selesai\_konsultasi.value = e.control.data['jam\_selesai'][-5:]              jadwalkonsultasi.inputan\_ruang\_konsultasi.value = e.control.data['ruang\_konsultasi']              jadwalkonsultasi.inputan\_client.value = e.control.data['id\_client']              jadwalkonsultasi.inputan\_konsultan.value = e.control.data['id\_konsultan']              jadwalkonsultasi.dialog.open = True              jadwalkonsultasi.update()          # fungsi simpan data          def simpan\_jadwalkonsultasi(e):              try:                  if (jadwalkonsultasi.inputan\_id\_jadwalkonsultasi.value == '') :                      # Mendapatkan id\_jadwalkonsultasi terakhir dari tabel jadwal konsultasi                      cursor.execute("SELECT id\_jadwalkonsultasi FROM tb\_jadwal\_konsultasi ORDER BY id\_jadwalkonsultasi DESC LIMIT 1")                      result = cursor.fetchone()                      # Memeriksa apakah id\_jadwalkonsultasi terakhir sudah ada atau belum                      if result and result[0]:                          # Jika sudah ada, kita tambahkan 1 ke id\_jadwalkonsultasi terakhir                          last\_id = result[0]                          id\_number = int(last\_id.split('-')[-1]) + 1                      else:                          # Jika belum ada, kita mulai dari 1                          id\_number = 1                      # Membuat angka dengan panjang 3 digit                      new\_id\_number = str(id\_number).zfill(3)                      # Membuat id\_client baru dengan format yang diinginkan                      new\_id\_jadwalkonsultasi = f"Jadwal-{new\_id\_number}"                      sql = "INSERT INTO tb\_jadwal\_konsultasi (id\_jadwalkonsultasi, tgl\_konsultasi, jam\_mulai, jam\_selesai, ruang\_konsultasi, id\_client, id\_konsultan) VALUES(%s, %s, %s, %s, %s, %s, %s)"                      val = (new\_id\_jadwalkonsultasi, str(jadwalkonsultasi.inputan\_tgl\_konsultasi.value), jadwalkonsultasi.inputan\_jam\_mulai\_konsultasi.value, jadwalkonsultasi.inputan\_jam\_selesai\_konsultasi.value, jadwalkonsultasi.inputan\_ruang\_konsultasi.value, jadwalkonsultasi.inputan\_client.value, jadwalkonsultasi.inputan\_konsultan.value)                  else :                      sql = "UPDATE tb\_jadwal\_konsultasi SET tgl\_konsultasi =  %s, jam\_mulai =  %s, jam\_selesai =  %s, ruang\_konsultasi =  %s, id\_client =  %s, id\_konsultan =  %s WHERE id\_jadwalkonsultasi = %s"                      val = (str(jadwalkonsultasi.inputan\_tgl\_konsultasi.value), jadwalkonsultasi.inputan\_jam\_mulai\_konsultasi.value, jadwalkonsultasi.inputan\_jam\_selesai\_konsultasi.value, jadwalkonsultasi.inputan\_ruang\_konsultasi.value, jadwalkonsultasi.inputan\_client.value, jadwalkonsultasi.inputan\_konsultan.value, jadwalkonsultasi.inputan\_id\_jadwalkonsultasi.value)                  cursor.execute(sql, val)                  koneksi\_db.commit()                  print(cursor.rowcount, "Data di simpan!")                  jadwalkonsultasi.nameinput.value = ''                  tampil\_data(e)                  jadwalkonsultasi.dialog.open = False                  jadwalkonsultasi.snack\_bar\_berhasil.open = True                  jadwalkonsultasi.update()              except Exception as e:                  print(e)                  print("Ada yang error!")          # fungsi hapus data          def hapus\_jadwalkonsultasi(e):              try:                  sql = "DELETE FROM tb\_jadwal\_konsultasi WHERE id\_jadwalkonsultasi = %s"                  val = (e.control.data['id\_jadwalkonsultasi'],)                  cursor.execute(sql, val)                  koneksi\_db.commit()                  print(cursor.rowcount, "data di hapus!")                  jadwalkonsultasi.data\_jadwalkonsultasi.rows.clear()                  jadwalkonsultasi.nameinput.value = ''                    tampil\_data(e)                  jadwalkonsultasi.dialog.open = False                  jadwalkonsultasi.snack\_bar\_berhasil.open = True                  jadwalkonsultasi.update()              except Exception as e:                  print(e)                  print("Ada yang error!")          jadwalkonsultasi.data\_jadwalkonsultasi = DataTable(              columns = [                  DataColumn(Text("Nomor")),                  DataColumn(Text("ID Jadwal Konsultasi")),                  DataColumn(Text("Tanggal Konsultasi")),                  DataColumn(Text("Jam Konsultasi")),                  DataColumn(Text("Durasi Konsultasi")),                  DataColumn(Text("Ruang Konsultasi")),                  DataColumn(Text("Nama Client")),                  DataColumn(Text("Nama Konsultan")),                  DataColumn(Text("Opsi")),              ],          )          cursor.execute("SELECT \* FROM tb\_jadwal\_konsultasi, tb\_konsultan, tb\_client WHERE tb\_jadwal\_konsultasi.id\_konsultan = tb\_konsultan.id\_konsultan AND tb\_jadwal\_konsultasi.id\_client = tb\_client.id\_client ORDER BY id\_jadwalkonsultasi ASC")          result = cursor.fetchall()          columns = [column[0] for column in cursor.description]          rows = [dict(zip(columns,row)) for row in result]          for index, row in enumerate(rows, start = 1):              # Ambil nilai jam mulai dan jam selesai dari baris saat ini              jam\_mulai = row['jam\_mulai']              jam\_selesai = row['jam\_selesai']                # Konversi jam mulai dan jam selesai ke objek datetime              fmt = '%H:%M'              dt\_mulai = datetime.datetime.strptime(jam\_mulai, fmt)              dt\_selesai = datetime.datetime.strptime(jam\_selesai, fmt)                # Hitung selisih waktu dalam menit              selisih\_menit = (dt\_selesai - dt\_mulai).seconds // 60                jadwalkonsultasi.data\_jadwalkonsultasi.rows.append(                  DataRow(                      cells = [                          DataCell(Text(str(index))),                          DataCell(Text(row['id\_jadwalkonsultasi'])),                          DataCell(Text(str(row['tgl\_konsultasi'].strftime("%d-%B-%Y")))),                          DataCell(Text(row['jam\_mulai'] + ' sampai ' + row['jam\_selesai'])),                          DataCell(Text(f"({selisih\_menit // 60} jam {selisih\_menit % 60} menit)")),                          DataCell(Text(row['ruang\_konsultasi'])),                          DataCell(Text(row['nama\_client'])),                          DataCell(Text(row['nama\_konsultan'])),                          DataCell(                              Row([                                  IconButton("delete", icon\_color = "red", data = row, on\_click = hapus\_jadwalkonsultasi),                                  IconButton("create", icon\_color = "grey", data = row, on\_click = tampil\_dialog\_ubah),                              ])                          ),                      ]                  )              )            # buat variabel utk pencarian data          jadwalkonsultasi.nameinput = TextField(              label = "Cari data",              hint\_text = "masukkan data yang ingin dicari ... ",              on\_change = inputsearch          )          jadwalkonsultasi.datanotfound = Text(              "Data yang dicari tidak ada...",              weight = "bold",              size = 18          )          jadwalkonsultasi.datanotfound.visible = False          # buat variabel utk layout data rekapan          jadwalkonsultasi.layout\_data = Column()          # buat form dialog untuk form entri data          jadwalkonsultasi.dialog = BottomSheet(              Container(                  Column(                      [                          Text("Form Entri Jadwal Konsultasi", weight = FontWeight.BOLD),                          Row([ jadwalkonsultasi.inputan\_id\_jadwalkonsultasi ]),                          Row([                              jadwalkonsultasi.inputan\_tgl\_konsultasi, jadwalkonsultasi.opsi\_tanggal,                              IconButton(                                  icon = icons.DATE\_RANGE\_OUTLINED,                                  tooltip = "Pilih Tanggal",                                  on\_click = lambda \_: jadwalkonsultasi.opsi\_tanggal.pick\_date(),                              ),                              ]),                          Row([                               jadwalkonsultasi.inputan\_jam\_mulai\_konsultasi, time\_picker\_jam\_mulai,                               IconButton(                                  icon=icons.MORE\_TIME\_ROUNDED,                                  on\_click=lambda \_: time\_picker\_jam\_mulai.pick\_time(),                              ),                           ]),                          Row([                               jadwalkonsultasi.inputan\_jam\_selesai\_konsultasi, time\_picker\_jam\_selesai,                               IconButton(                                  icon=icons.MORE\_TIME\_ROUNDED,                                  on\_click=lambda \_: time\_picker\_jam\_selesai.pick\_time(),                              ),                           ]),                          Row([ jadwalkonsultasi.inputan\_ruang\_konsultasi ]),                          Row([ jadwalkonsultasi.inputan\_client ]),                          Row([ jadwalkonsultasi.inputan\_konsultan ]),                            Row([                              #tombol tambah data                              ElevatedButton(                                  "Simpan Data",                                      icon = "SAVE\_AS",                                      icon\_color = "white",                                      color = "white",                                      bgcolor = "#2C4E80",                                      width = 280,                                      height = 50,                                      on\_click = simpan\_jadwalkonsultasi,                                  )                          ]),                        ],                      horizontal\_alignment = CrossAxisAlignment.CENTER,                      scroll = ScrollMode.AUTO                      #tight = True,                  ),                  padding = 40,                  width = 378,                  height = 500              ),              open = False,          )          return Column(              controls = [                  Row([ElevatedButton("Tambah Data", icon = icons.ADD, icon\_color = "white", width = 335, color = "white", bgcolor = "#2C4E80", on\_click = tampil\_dialog)], alignment = MainAxisAlignment.END),                  jadwalkonsultasi.nameinput,                  Row([jadwalkonsultasi.data\_jadwalkonsultasi], scroll = ScrollMode.ALWAYS),                  jadwalkonsultasi.datanotfound,                  jadwalkonsultasi.dialog,                  jadwalkonsultasi.snack\_bar\_berhasil              ],          )  # fungsi utama  def main (page : Page):      # mengatur halaman      page.title = "Aplikasi CRUD (Menu & SQL)"      page.window\_width = 375      page.window\_height = 612      page.window\_resizable = False      page.window\_maximizable = True      page.window\_minimizable = True      page.scroll = "adaptive"      #page.theme\_mode = "light"      page.theme\_mode = ThemeMode.LIGHT      # fungsi untuk mode halaman dark/light      def mode\_tema(e):          page.theme\_mode = "light" if page.theme\_mode == "dark" else "dark"          page.update()      # fungsi untuk routing / pembagian halaman      def route\_change(route):          page.views.clear()          page.views.append(              View("/",                  [                      AppBar(                          title = Text("Aplikasi CRUD Jadwal Konsultasi Pengaduan", size = 1, weight = FontWeight.BOLD, color = colors.WHITE),                          bgcolor = "#2C4E80",                          center\_title = True,                          actions = [                              IconButton(icons.WB\_SUNNY\_OUTLINED, icon\_color = "white", on\_click = mode\_tema),                          ],                      ),                      Column(                          [                              Image(src = "Konsultasi.jpg", width = 200, border\_radius = 10 ),                              #Icon(name = icons.CALENDAR\_MONTH\_ROUNDED, color = "#6C9BCF", size = 180),                              Column(                                  controls = [                                      ElevatedButton("Menu Client", icon = icons.PERSON\_3\_ROUNDED, on\_click = lambda \_: page.go("/client")),                                      ElevatedButton("Menu Konsultan", icon = icons.PEOPLE\_ROUNDED, on\_click = lambda \_: page.go("/konsultan")),                                      ElevatedButton("Menu Jadwal Konsultasi", icon = icons.CALENDAR\_MONTH\_SHARP, on\_click = lambda \_: page.go("/jadwalkonsultasi")),                                  ],                                  width = 375,                                  horizontal\_alignment = CrossAxisAlignment.CENTER,                              ),                          ],                          height = 400,                          width = 375,                          alignment = MainAxisAlignment.SPACE\_AROUND,                          horizontal\_alignment = CrossAxisAlignment.CENTER,                      ),                      BottomAppBar(                          bgcolor = '#2C4E80',                          height = 45,                          content = Row(                              controls = [                                  Text('Mobile Programming @2024', size = 13, color = colors.WHITE)                              ],                              alignment = MainAxisAlignment.SPACE\_AROUND,                          ),                      ),                  ],              )          )          if page.route == "/client":              page.views.append(                  View("/client",                      [                          AppBar(title = Text("Menu Client", size = 14, weight = FontWeight.BOLD), bgcolor = colors.SURFACE\_VARIANT),                          FormClient()                      ],                  )              )          elif page.route == "/konsultan":              page.views.append(                  View("/konsultan",                      [                          AppBar(title = Text("Menu Konsultan", size = 14, weight = FontWeight.BOLD), bgcolor = colors.SURFACE\_VARIANT),                          FormKonsultan()                      ],                  )              )          elif page.route == "/jadwalkonsultasi":              page.views.append(                  View("/jadwalkonsultasi",                      [                          AppBar(title = Text("Menu Jadwal Konsultasi", size = 14, weight = FontWeight.BOLD), bgcolor = colors.SURFACE\_VARIANT),                          FormJadwalKonsultasi()                      ],                  )              )            page.update()      # fungsi untuk pop up halaman      def view\_pop(view):          page.views.pop()          top\_view = page.views[-1]          page.go(top\_view.route)      page.on\_route\_change = route\_change      page.on\_view\_pop = view\_pop      page.go(page.route)  # mengatur output aplikasi  # flet.app(target = main)  flet.app(target = main, view = flet.AppView.WEB\_BROWSER) |
| 1. Screenshot Aplikasi / Tampilan / UI |
| * Table Client                * Table Konsultan            * Tabel Jadwal Konsultasi |
| 1. Screenshot Relasi Tabel |
|  |
|  |
|  |