# EXERCISE 5: NAVIGATION AND ROUTING

import flet as ft

import datetime

def main(page: ft.Page):

    page.title = "Navigation and Routing"

    def handle\_change(e):

        page.add(ft.Text(f"Date changed: {e.control.value.strftime('%Y-%m-%d')}"))

    def handle\_dismissal(e):

        page.add(ft.Text(f"DatePicker dismissed"))

    name = ft.TextField(label="Name: ")

    date = ft.DatePicker(

      value=datetime.date.today(),

      on\_change=handle\_change,

      on\_dismiss=handle\_dismissal,

    )

    sex = ft.RadioGroup(content=ft.Row([

      ft.Radio(value="Female", label="Female"),

      ft.Radio(value="Male", label="Male"),

      ft.Radio(value="Other", label="Other"),]))

    address = ft.TextField(label="Address: ")

    country = ft.Dropdown(label="Country: ",

             width=100,

             options=[

               ft.dropdown.Option("Finland"),

               ft.dropdown.Option("Sweden"),

               ft.dropdown.Option("Norway"),

               ft.dropdown.Option("USA"),

               ft.dropdown.Option("Spain"),

               ft.dropdown.Option("France"),

               ft.dropdown.Option("Other"),

             ])

    username = ft.TextField(label="Username")

    password = ft.TextField(label="Password", password=True, can\_reveal\_password=True)

    def validate\_form(e):

        has\_error = False

        # Check username

        if not username.value:

          username.error\_text="The username is required"

          has\_error=True

        else:

          username.error\_text=None

        # Check password

        if not password.value:

          password.error\_text="The password is required"

          has\_error=True

        else:

          password.error\_text=None

        username.update()

        password.update()

        # If there isn't any problem, it goes to forms

        if not has\_error:

          page.go("/form")

    def route\_change(route):

        page.views.clear()

        page.views.append(

          ft.View("/", [

            ft.AppBar(title=ft.Text("Navigation and Routing"), bgcolor=ft.colors.SURFACE\_VARIANT),

            username,

            password,

            ft.ElevatedButton("Login", on\_click=validate\_form),

          ],))

        if page.route == "/form":

          page.views.append(

            ft.View("/form", [

              ft.AppBar(title=ft.Text("Form"), bgcolor=ft.colors.SURFACE\_VARIANT),

              name,

              ft.Text("Birth date"),

              ft.ElevatedButton(

              "Date",

              icon=ft.icons.CALENDAR\_MONTH,

              on\_click=lambda e: page.open(date),),

              ft.Text("Sex: "),

              sex,

              address,

              country,

              ft.ElevatedButton("Show Details", on\_click=lambda \_: page.go("/showdetails")),

              ft.ElevatedButton("Go Home", on\_click=lambda \_: page.go("/")),

],))

    if page.route == "/showdetails":

      page.views.append(

        ft.View("/showdetails", [

        ft.AppBar(title=ft.Text("Show Details"), bgcolor=ft.colors.SURFACE\_VARIANT),

            ft.Card(content=ft.Container(

               content=ft.Column([

           ft.Text(f"Name: {name.value}"),

ft.Text(f"Birth date: {date.value.day}-{date.value.month}-{date.value.year}"),

                 ft.Text(f"Sex: {sex.value}"),

                  ft.Text(f"Address: {address.value}"),

                  ft.Text(f"Country: {country.value}"),

                  ft.Row([ft.ElevatedButton("Return to form", on\_click=lambda \_: page.go("/form"))],

                  alignment=ft.MainAxisAlignment.END,

                 ),

               ],),

              width=400,

              padding=10,

            )),

        ],))

        page.update()

    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)

ft.app(main, view=ft.AppView.WEB\_BROWSER)A screenshot of a login screen

Description automatically generated

A screenshot of a login screen

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a form

Description automatically generated