Book Marketplace Project

By:…………………………………………

Delhi Public School - Bopal, Ahmedabad

CERTIFICATE

This is to certify that ____________________ student of Class XII has successfully completed project on the topic **Book Marketplace** under the guidance of Ms. Malvika Sharma, during the Academic year 2023-2024, for partial fulfilment of Computer Science (083), Practical Examination conducted by AISSCE 2023-2024.

Subject: Computer Science (083)

Board Roll number: ___________________

Subject Teacher   Examiner

Date: ___________

Index

Acknowledgement

I extend my sincere gratitude to my dedicated Computer Science teacher, Ms. Malvika Sharma, whose guidance was pivotal in the successful completion of my Class 12 project. She not only shared valuable insights but also provided unwavering support, fostering an environment where I could explore and apply complex concepts with confidence.

Ms. Malvika Sharma exhibited a remarkable balance of encouragement and constructive critique, significantly enhancing the depth and quality of my project. The commitment to our learning journey and the accessibility to address queries played a crucial role in shaping my understanding of the subject.

Completing this project under Ms. Malvika Sharma’s mentorship has been an enriching experience, offering not only academic insights but also instilling a passion for the field. I am grateful for her dedication and the positive impact she has had on my educational trajectory.

……………………………………………

Introduction

This project presents a Book Marketplace that provides a digital platform to view and purchase books. Our project aims to showcase the application of fundamental Python and Database management principles.

the application allows the user to search for books, create an account and log in, add books to the marketplace, view book information and purchase books. The application is displayed in form a neat user interface made using the Flet UI framework in python. At the back end of the project, it consists of a database with two tables related to user information and book information. The project is also presenting a neat user interface.

System Specifications

A) Hardware:

- **CPU:** Intel® Core™ i5-10400T @2.00GHz
- **Memory:** 8.0 GB
- **GPU:** Intel® UHD Graphics 630

B) Software:

- **OS:** Windows® 11
- **Development tools:** Python, SQL, written in VS Code

Modules and Functions

A) *User Defined Functions:*

- addBook: This Function is used to add a unique book entry to the database and accepts information for fields such as book_id, book_name, book_description, book_year, etc.

- addUser: This function is used to add a user to the database and accepts information as user_name, user_id, user password and etc.

- getUser: This function is used to fetch information of the specified username from the database

- getAllBooks: This function is used to fetch information about all books.

- getBook: This function is used to fetch information about the specified book from the database.

- getGenre: This function is used to fetch all the distinct genres mentioned in the database.

- updateBook: this function is used to update the stock of the mentioned book

- delAllUsers: Function used to delete all users from the database.

- delAllBooks: Function used to delete all books from the database.

- delBook: Function used to delete book of specified id from the database.

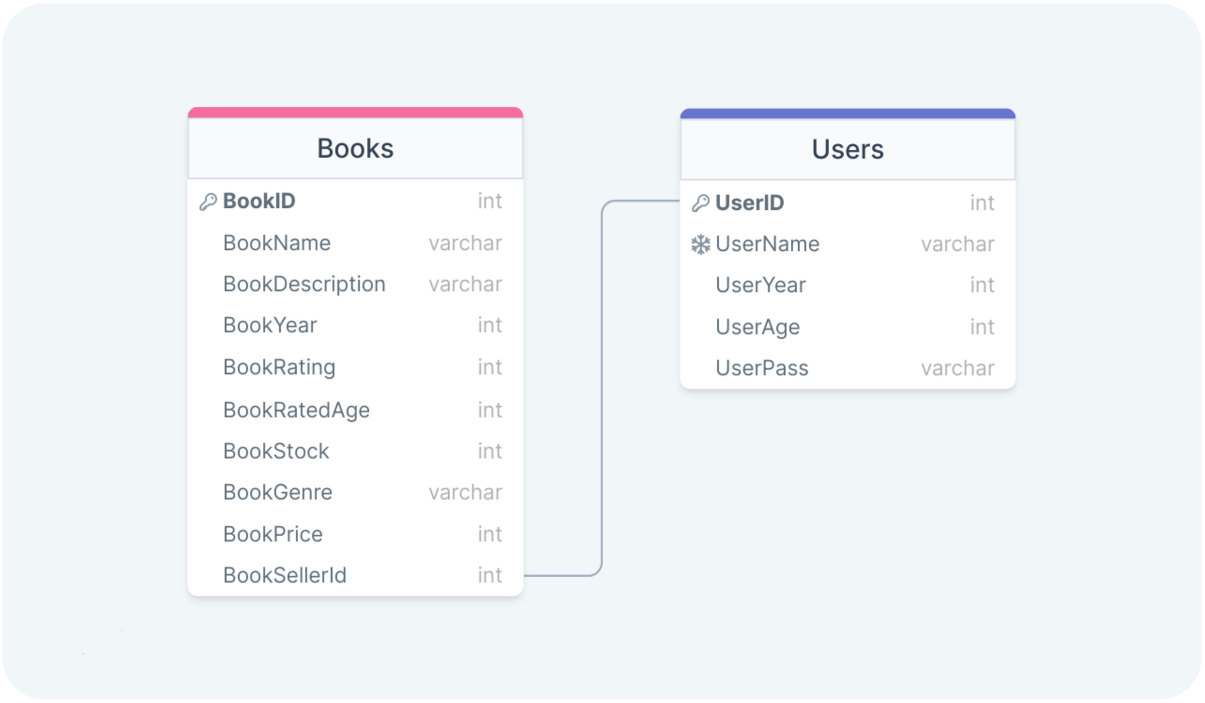
- build: Used across multiple classes, this function is used to construct the view of the page.

Database Structure

Database Design:

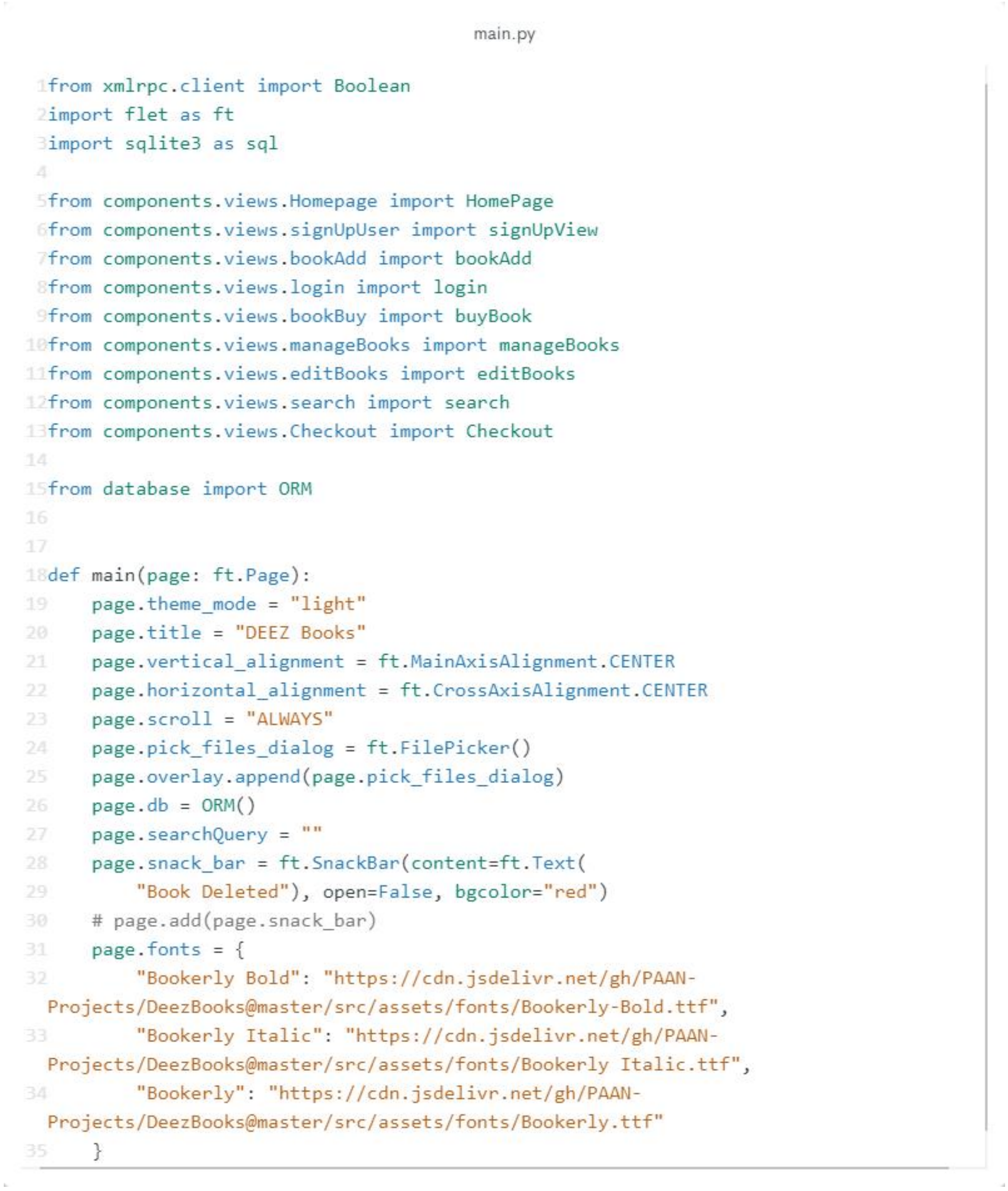
The database consists of two tables; one for User information and the other for Book information. The information about the field datatypes and keys can be viewed in the graphic shown below:

File Name: database.db

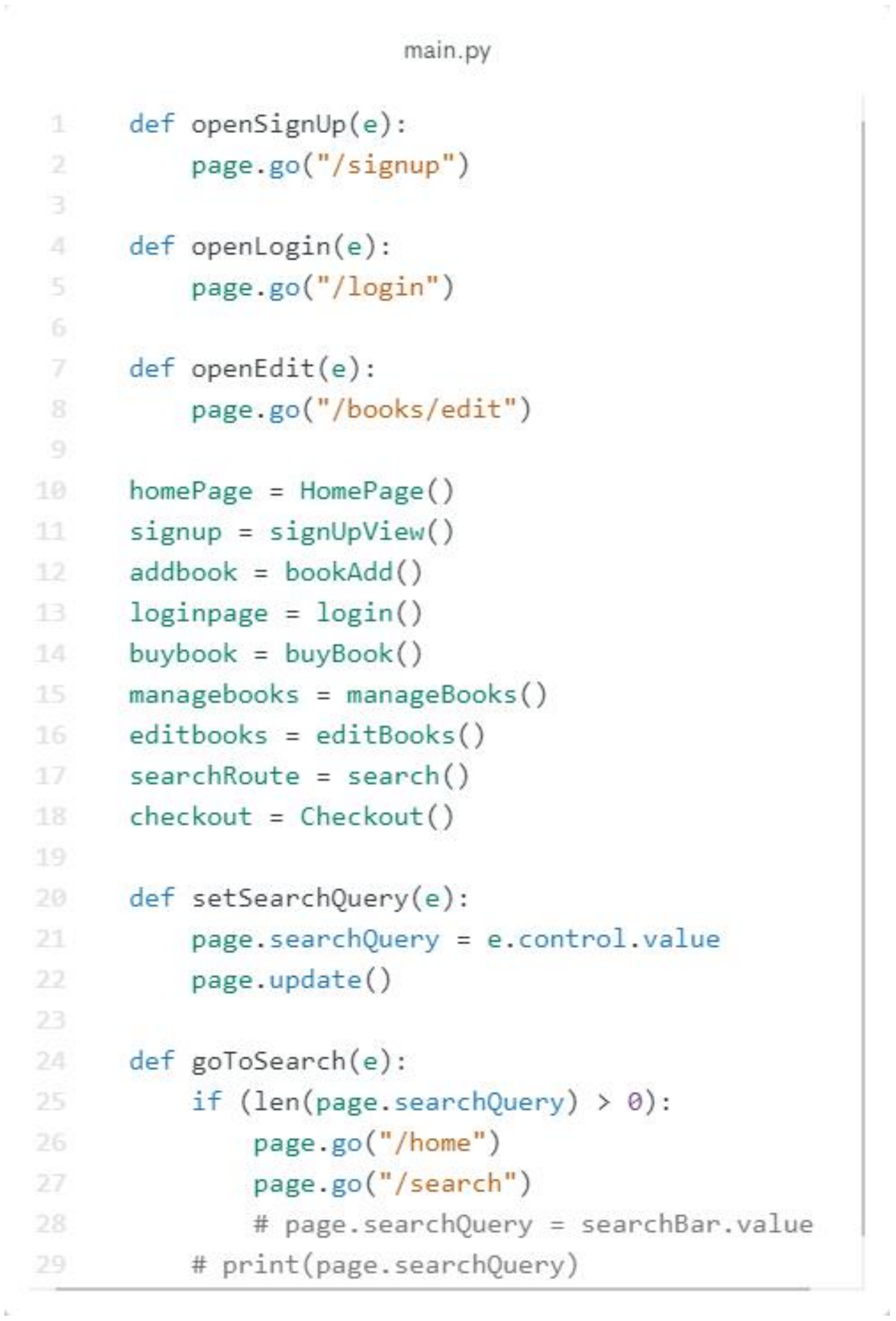


Python Code

File: main.py

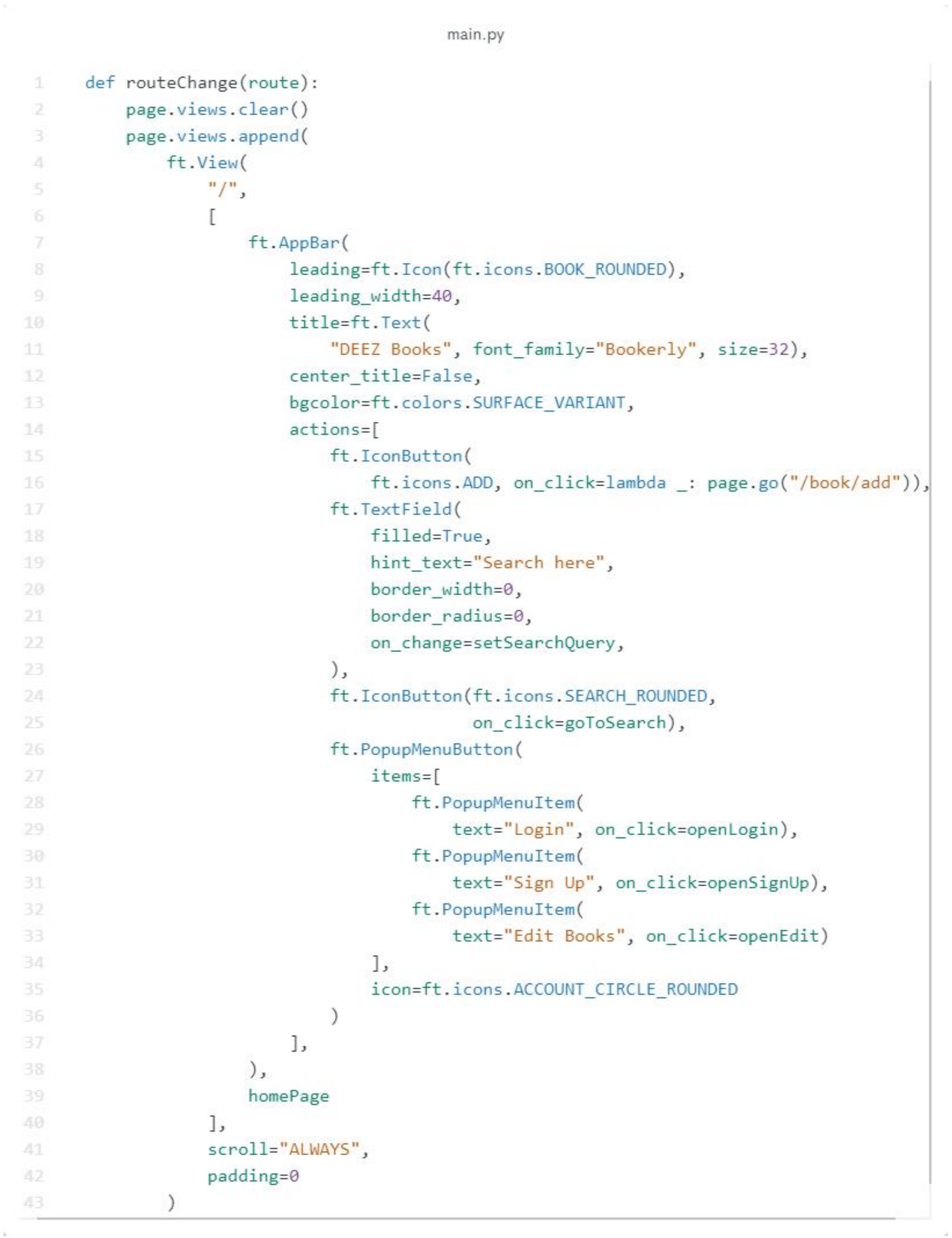

main.py

```python
from xmlrpc.client import Boolean
import flet as ft
import sqlite3 as sql

from components.views.Homepage import HomePage
from components.views.signUpUser import signUpView
from components.views.bookAdd import bookAdd
from components.views.login import login
from components.views.bookBuy import buyBook
from components.views.manageBooks import manageBooks
from components.views.editBooks import editBooks
from components.views.search import search
from components.views.Checkout import Checkout

from database import ORM


def main(page: ft.Page):
    page.theme_mode = "light"
    page.title = "DEEZ Books"
    page.vertical_alignment = ft.MainAxisAlignment.CENTER
    page.horizontal_alignment = ft.CrossAxisAlignment.CENTER
    page.scroll = "ALWAYS"
    page.pick_files_dialog = ft.FilePicker()
    page.overlay.append(page.pick_files_dialog)
    page.db = ORM()
    page.searchQuery = ""
    page.snack_bar = ft.SnackBar(content=ft.Text(
        "Book Deleted"), open=False, bgcolor="red")
    # page.add(page.snack_bar)
    page.fonts = {
        "Bookerly Bold": "https://cdn.jsdelivr.net/gh/PAAN-Projects/DeezBooks@master/src/assets/fonts/Bookerly-Bold.ttf",
        "Bookerly Italic": "https://cdn.jsdelivr.net/gh/PAAN-Projects/DeezBooks@master/src/assets/fonts/Bookerly Italic.ttf",
        "Bookerly": "https://cdn.jsdelivr.net/gh/PAAN-Projects/DeezBooks@master/src/assets/fonts/Bookerly.ttf"
    }
```

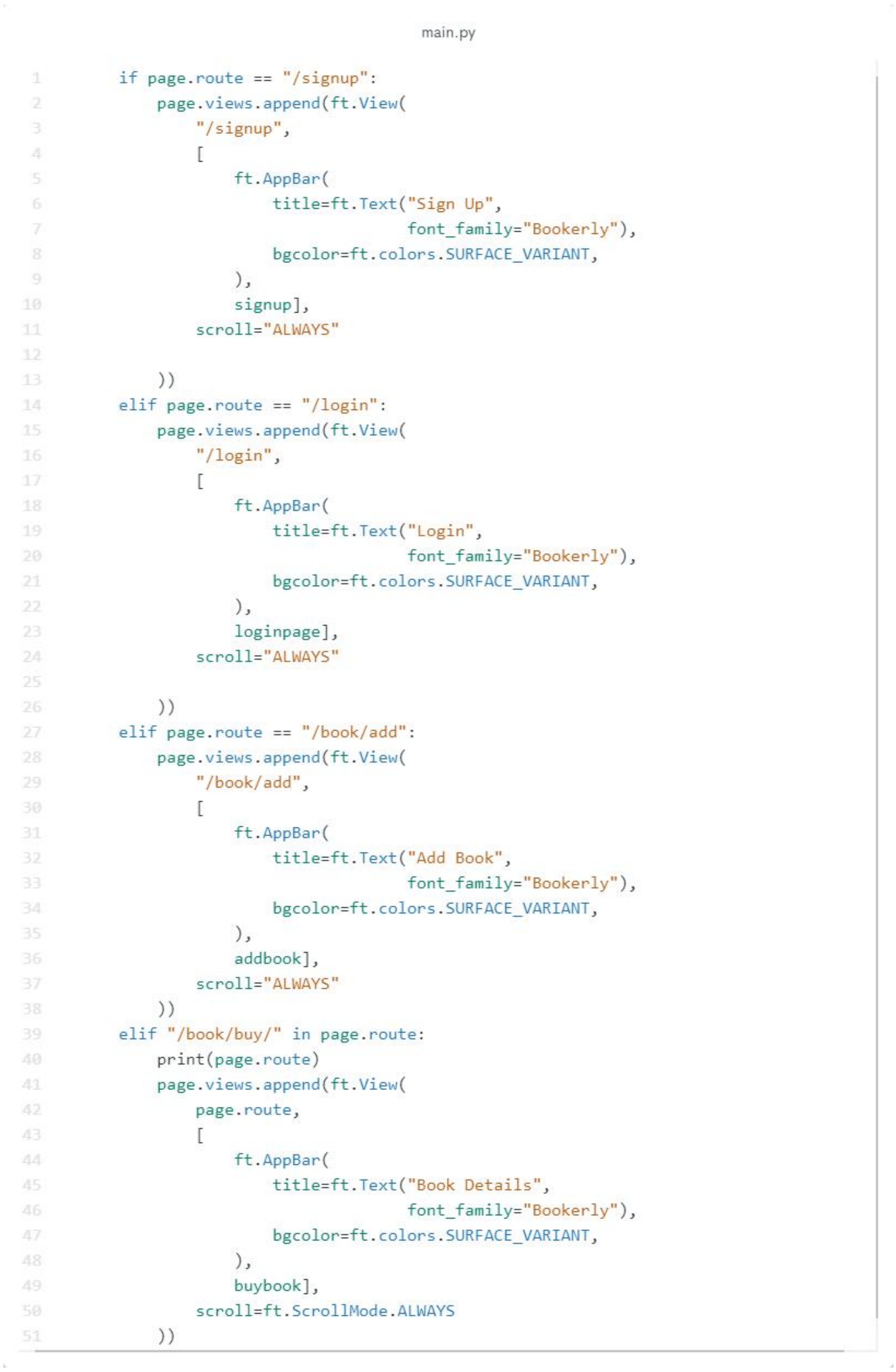


main.py

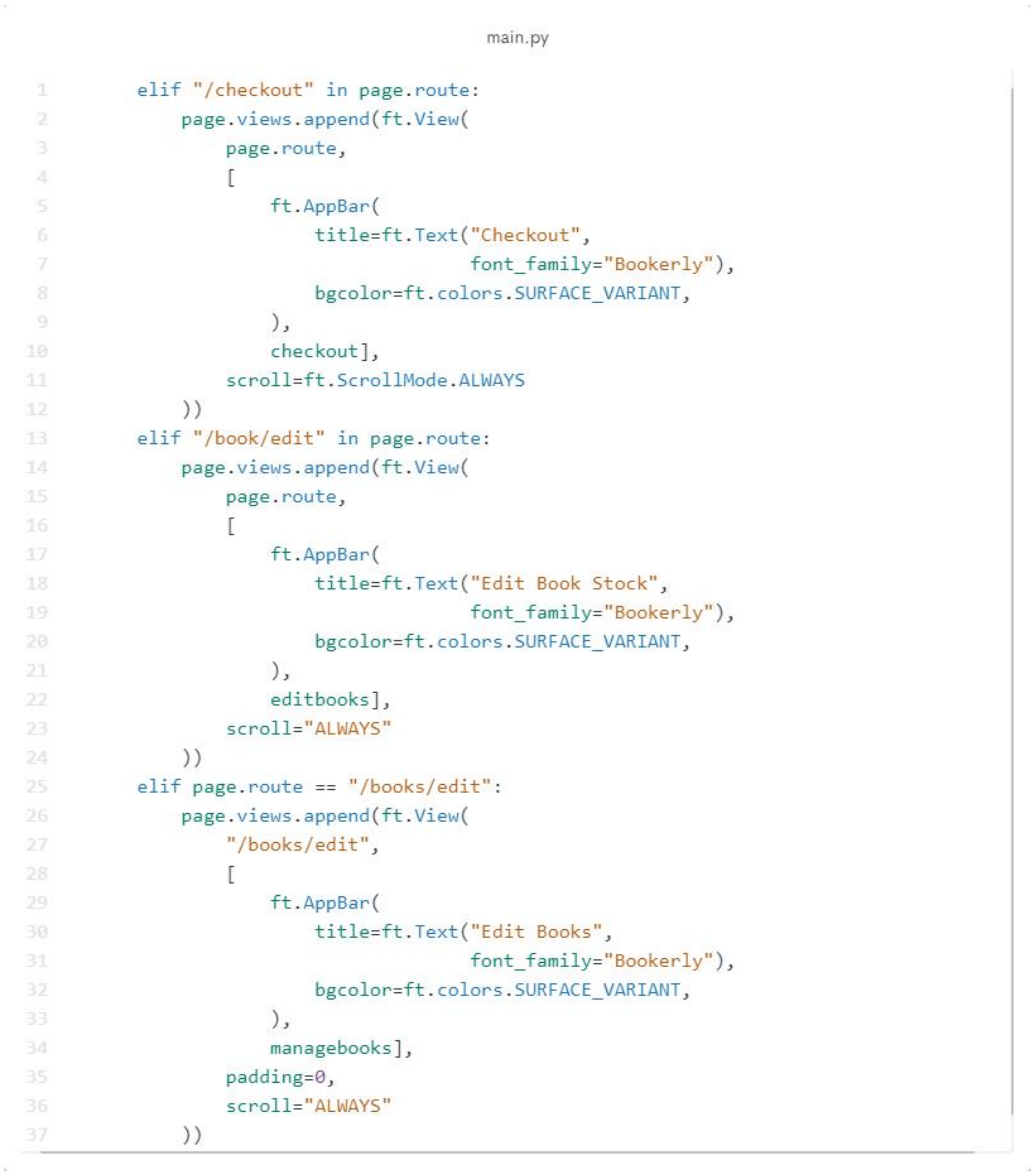
```python
def openSignUp(e):
    page.go("/signup")

def openLogin(e):
    page.go("/login")

def openEdit(e):
    page.go("/books/edit")

homePage = HomePage()
signup = signUpView()
addbook = bookAdd()
loginpage = login()
buybook = buyBook()
managebooks = manageBooks()
editbooks = editBooks()
searchRoute = search()
checkout = Checkout()

def setSearchQuery(e):
    page.searchQuery = e.control.value
    page.update()

def goToSearch(e):
    if (len(page.searchQuery) > 0):
        page.go("/home")
        page.go("/search")
        # page.searchQuery = searchBar.value
    # print(page.searchQuery)
```


main.py

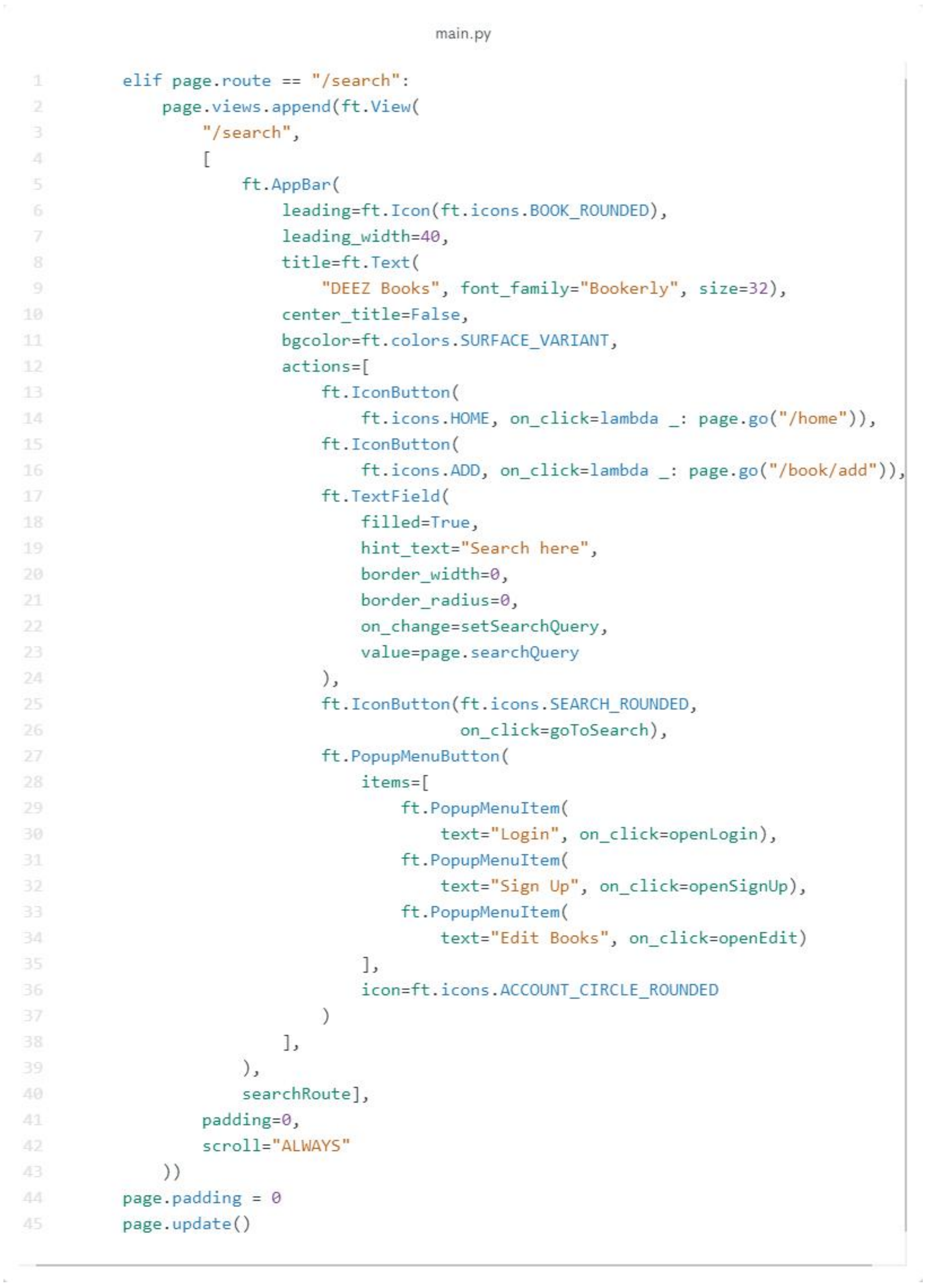
```python
def routeChange(route):
    page.views.clear()
    page.views.append(
        ft.View(
            "/",
            [
                ft.AppBar(
                    leading=ft.Icon(ft.icons.BOOK_ROUNDED),
                    leading_width=40,
                    title=ft.Text(
                        "DEEZ Books", font_family="Bookerly", size=32),
                    center_title=False,
                    bgcolor=ft.colors.SURFACE_VARIANT,
                    actions=[
                        ft.IconButton(
                            ft.icons.ADD, on_click=lambda _: page.go("/book/add")),
                        ft.TextField(
                            filled=True,
                            hint_text="Search here",
                            border_width=0,
                            border_radius=0,
                            on_change=setSearchQuery,
                        ),
                        ft.IconButton(ft.icons.SEARCH_ROUNDED,
                                      on_click=goToSearch),
                        ft.PopupMenuButton(
                            items=[
                                ft.PopupMenuItem(
                                    text="Login", on_click=openLogin),
                                ft.PopupMenuItem(
                                    text="Sign Up", on_click=openSignUp),
                                ft.PopupMenuItem(
                                    text="Edit Books", on_click=openEdit)
                            ],
                            icon=ft.icons.ACCOUNT_CIRCLE_ROUNDED
                        )
                    ],
                ),
                homePage
            ],
            scroll="ALWAYS",
            padding=0
        )
```


main.py

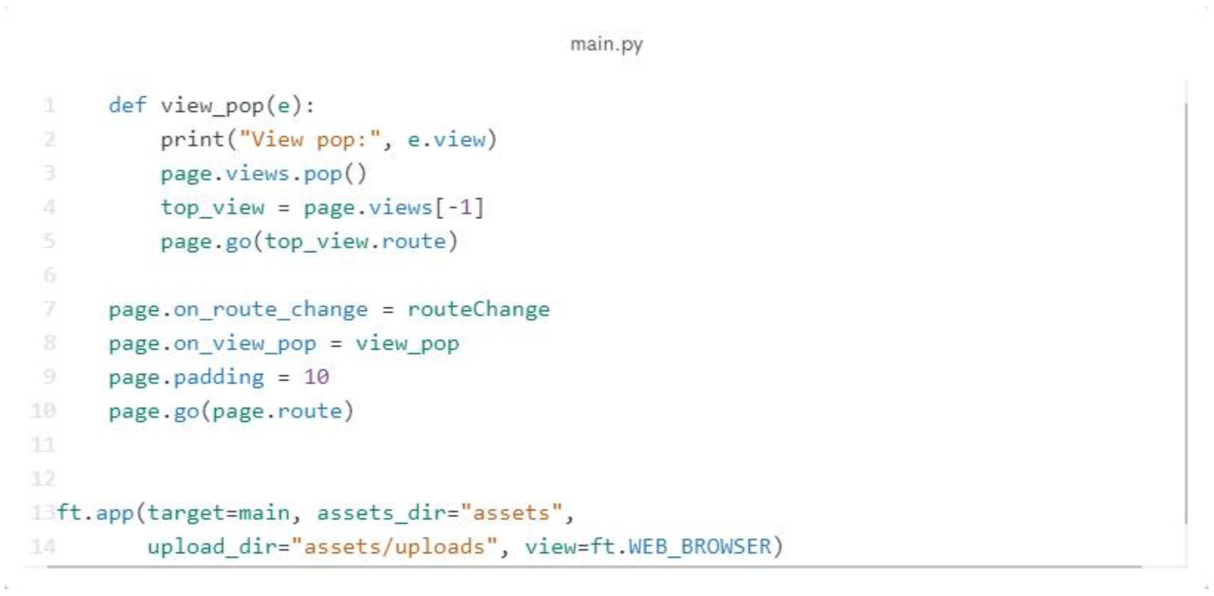
```python
        if page.route == "/signup":
            page.views.append(ft.View(
                "/signup",
                [
                    ft.AppBar(
                        title=ft.Text("Sign Up",
                                      font_family="Bookerly"),
                        bgcolor=ft.colors.SURFACE_VARIANT,
                    ),
                    signup],
                scroll="ALWAYS"

            ))
        elif page.route == "/login":
            page.views.append(ft.View(
                "/login",
                [
                    ft.AppBar(
                        title=ft.Text("Login",
                                      font_family="Bookerly"),
                        bgcolor=ft.colors.SURFACE_VARIANT,
                    ),
                    loginpage],
                scroll="ALWAYS"

            ))
        elif page.route == "/book/add":
            page.views.append(ft.View(
                "/book/add",
                [
                    ft.AppBar(
                        title=ft.Text("Add Book",
                                      font_family="Bookerly"),
                        bgcolor=ft.colors.SURFACE_VARIANT,
                    ),
                    addbook],
                scroll="ALWAYS"
            ))
        elif "/book/buy/" in page.route:
            print(page.route)
            page.views.append(ft.View(
                page.route,
                [
                    ft.AppBar(
                        title=ft.Text("Book Details",
                                      font_family="Bookerly"),
                        bgcolor=ft.colors.SURFACE_VARIANT,
                    ),
                    buybook],
                scroll=ft.ScrollMode.ALWAYS
            ))
```


main.py

```python
        elif "/checkout" in page.route:
            page.views.append(ft.View(
                page.route,
                [
                    ft.AppBar(
                        title=ft.Text("Checkout",
                                      font_family="Bookerly"),
                        bgcolor=ft.colors.SURFACE_VARIANT,
                    ),
                    checkout],
                scroll=ft.ScrollMode.ALWAYS
            ))
        elif "/book/edit" in page.route:
            page.views.append(ft.View(
                page.route,
                [
                    ft.AppBar(
                        title=ft.Text("Edit Book Stock",
                                      font_family="Bookerly"),
                        bgcolor=ft.colors.SURFACE_VARIANT,
                    ),
                    editbooks],
                scroll="ALWAYS"
            ))
        elif page.route == "/books/edit":
            page.views.append(ft.View(
                "/books/edit",
                [
                    ft.AppBar(
                        title=ft.Text("Edit Books",
                                      font_family="Bookerly"),
                        bgcolor=ft.colors.SURFACE_VARIANT,
                    ),
                    managebooks],
                padding=0,
                scroll="ALWAYS"
            ))
```

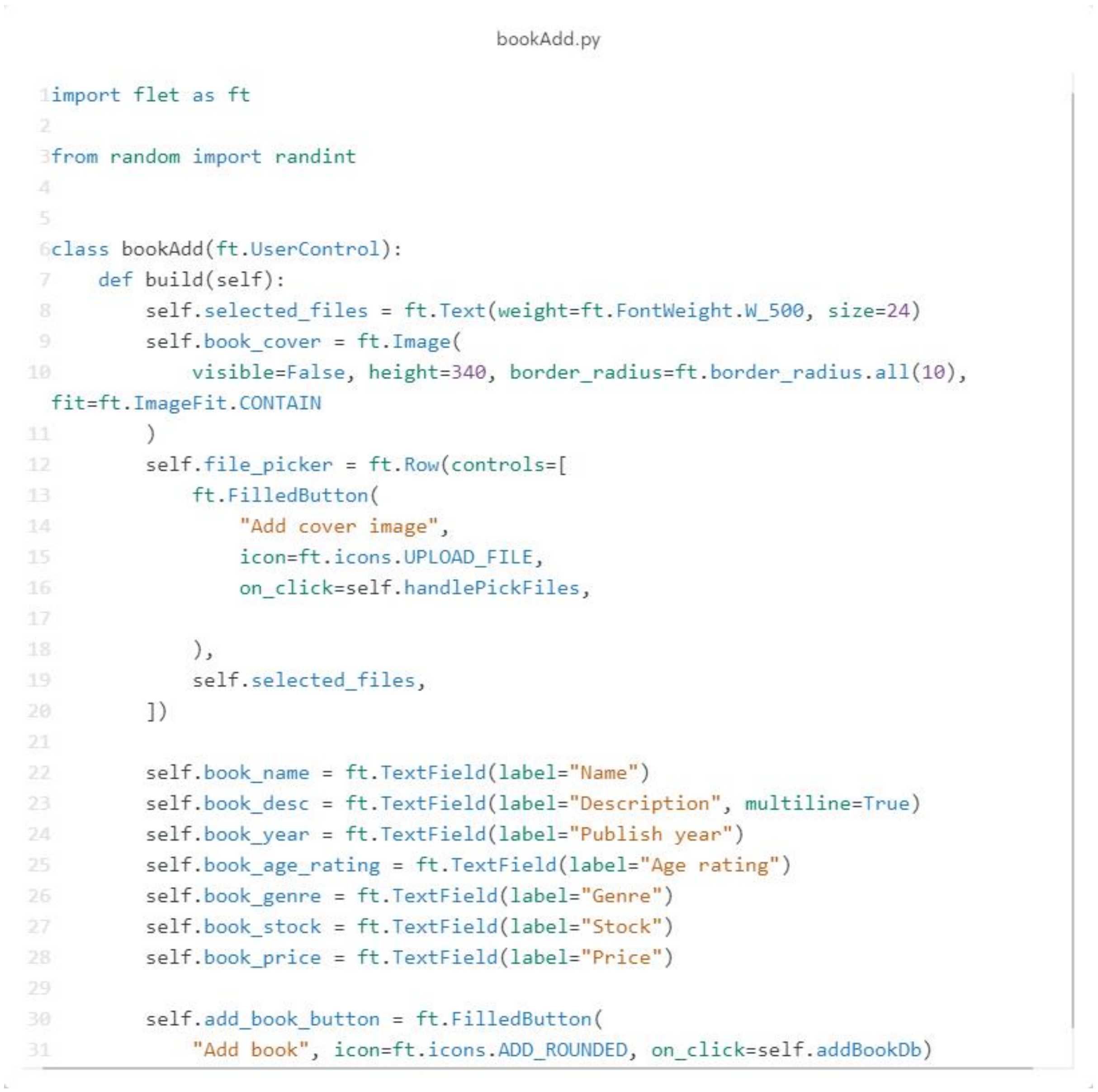

main.py

```
        elif page.route == "/search":
            page.views.append(ft.View(
                "/search",
                [
                    ft.AppBar(
                        leading=ft.Icon(ft.icons.BOOK_ROUNDED),
                        leading_width=40,
                        title=ft.Text(
                            "DEEZ Books", font_family="Bookerly", size=32),
                        center_title=False,
                        bgcolor=ft.colors.SURFACE_VARIANT,
                        actions=[
                            ft.IconButton(
                                ft.icons.HOME, on_click=lambda _: page.go("/home")),
                            ft.IconButton(
                                ft.icons.ADD, on_click=lambda _: page.go("/book/add")),
                            ft.TextField(
                                filled=True,
                                hint_text="Search here",
                                border_width=0,
                                border_radius=0,
                                on_change=setSearchQuery,
                                value=page.searchQuery
                            ),
                            ft.IconButton(ft.icons.SEARCH_ROUNDED,
                                          on_click=goToSearch),
                            ft.PopupMenuButton(
                                items=[
                                    ft.PopupMenuItem(
                                        text="Login", on_click=openLogin),
                                    ft.PopupMenuItem(
                                        text="Sign Up", on_click=openSignUp),
                                    ft.PopupMenuItem(
                                        text="Edit Books", on_click=openEdit)
                                ],
                                icon=ft.icons.ACCOUNT_CIRCLE_ROUNDED
                            )
                        ],
                    ),
                    searchRoute],
                padding=0,
                scroll="ALWAYS"
            ))
        page.padding = 0
        page.update()
```

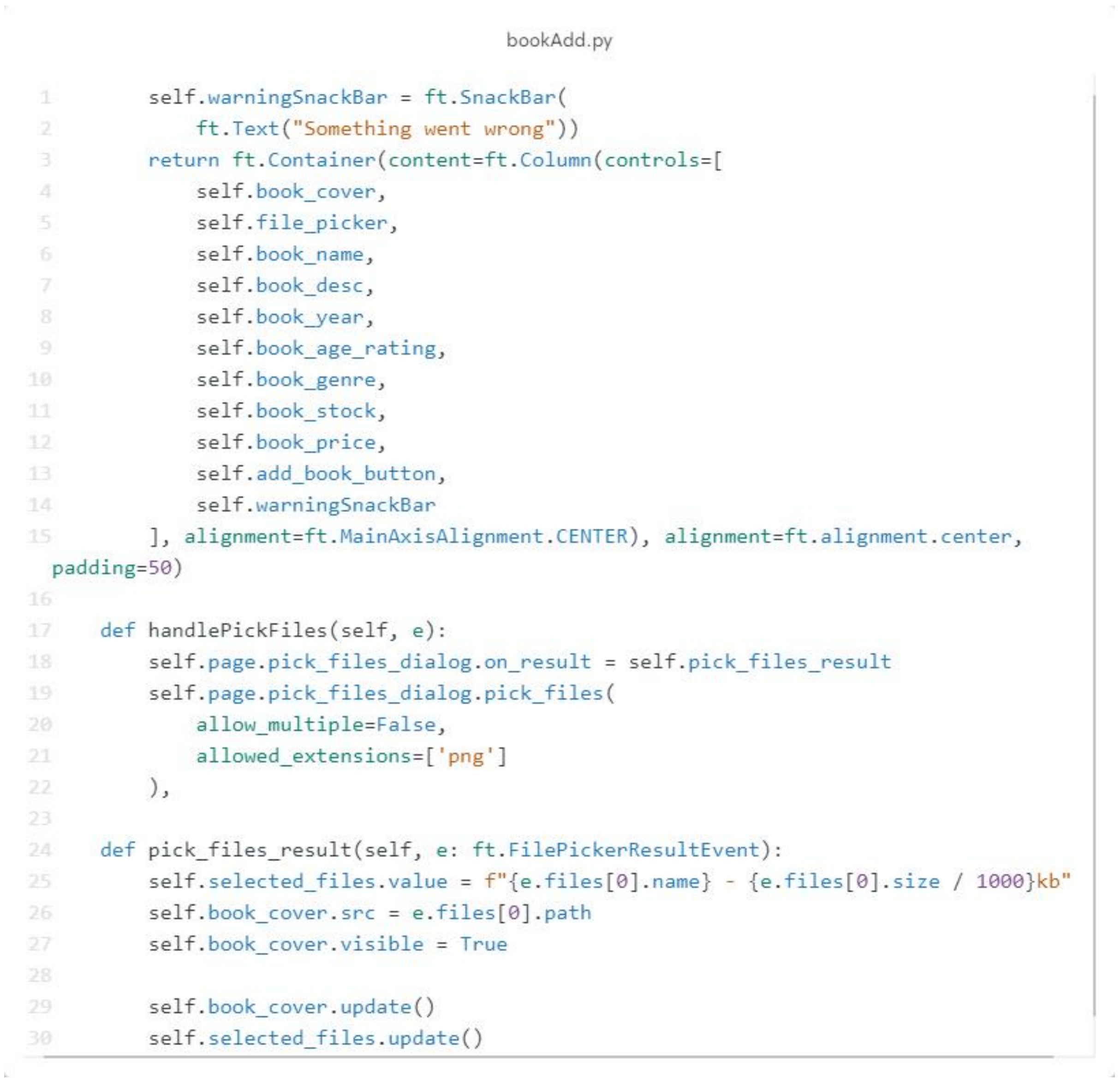

main.py

```python
    def view_pop(e):
        print("View pop:", e.view)
        page.views.pop()
        top_view = page.views[-1]
        page.go(top_view.route)

    page.on_route_change = routeChange
    page.on_view_pop = view_pop
    page.padding = 10
    page.go(page.route)


ft.app(target=main, assets_dir="assets",
       upload_dir="assets/uploads", view=ft.WEB_BROWSER)
```

File: bookAdd.py

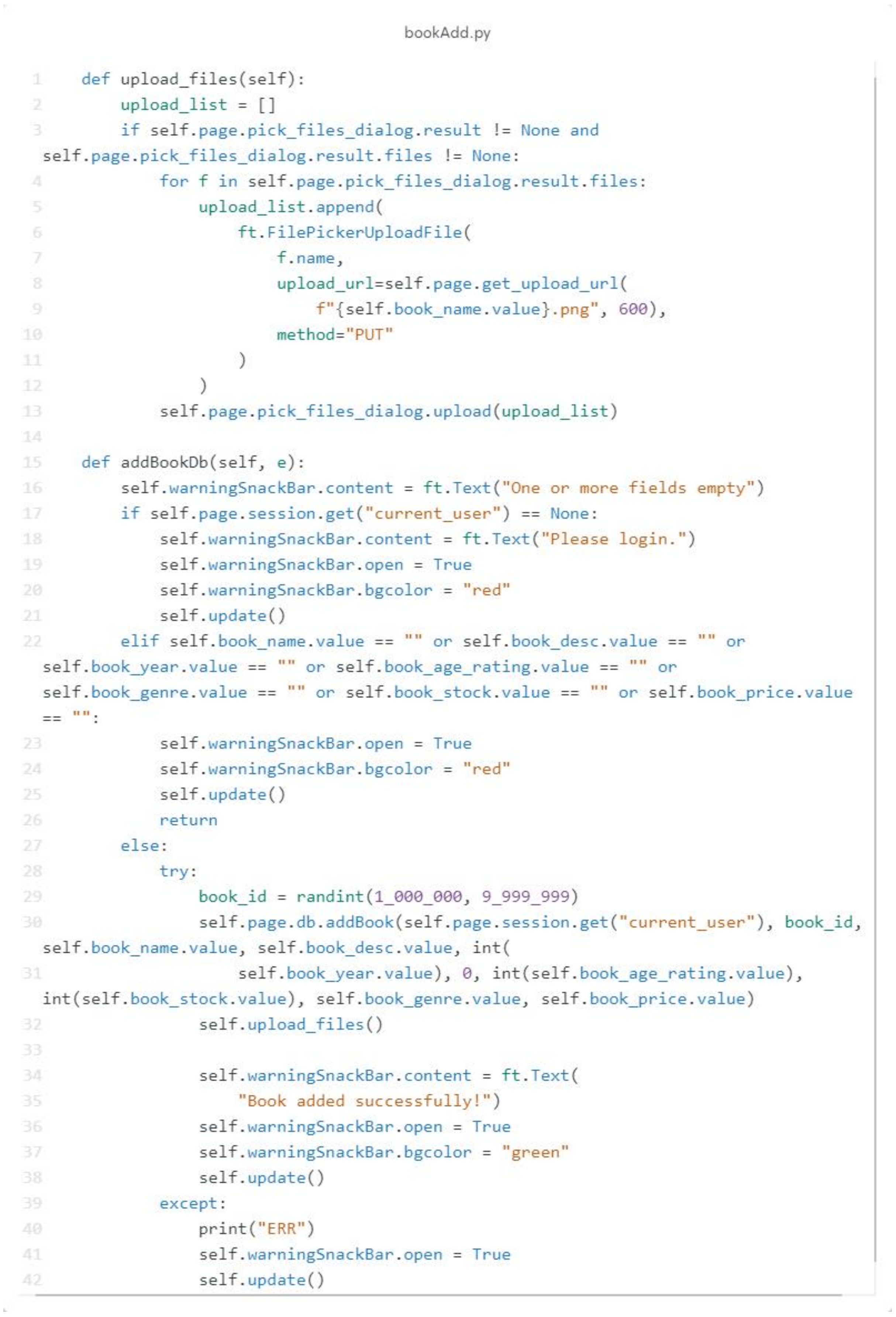

bookAdd.py

```python
import flet as ft

from random import randint


class bookAdd(ft.UserControl):
    def build(self):
        self.selected_files = ft.Text(weight=ft.FontWeight.W_500, size=24)
        self.book_cover = ft.Image(
            visible=False, height=340, border_radius=ft.border_radius.all(10),
fit=ft.ImageFit.CONTAIN
        )
        self.file_picker = ft.Row(controls=[
            ft.FilledButton(
                "Add cover image",
                icon=ft.icons.UPLOAD_FILE,
                on_click=self.handlePickFiles,

            ),
            self.selected_files,
        ])

        self.book_name = ft.TextField(label="Name")
        self.book_desc = ft.TextField(label="Description", multiline=True)
        self.book_year = ft.TextField(label="Publish year")
        self.book_age_rating = ft.TextField(label="Age rating")
        self.book_genre = ft.TextField(label="Genre")
        self.book_stock = ft.TextField(label="Stock")
        self.book_price = ft.TextField(label="Price")

        self.add_book_button = ft.FilledButton(
            "Add book", icon=ft.icons.ADD_ROUNDED, on_click=self.addBookDb)
```



bookAdd.py

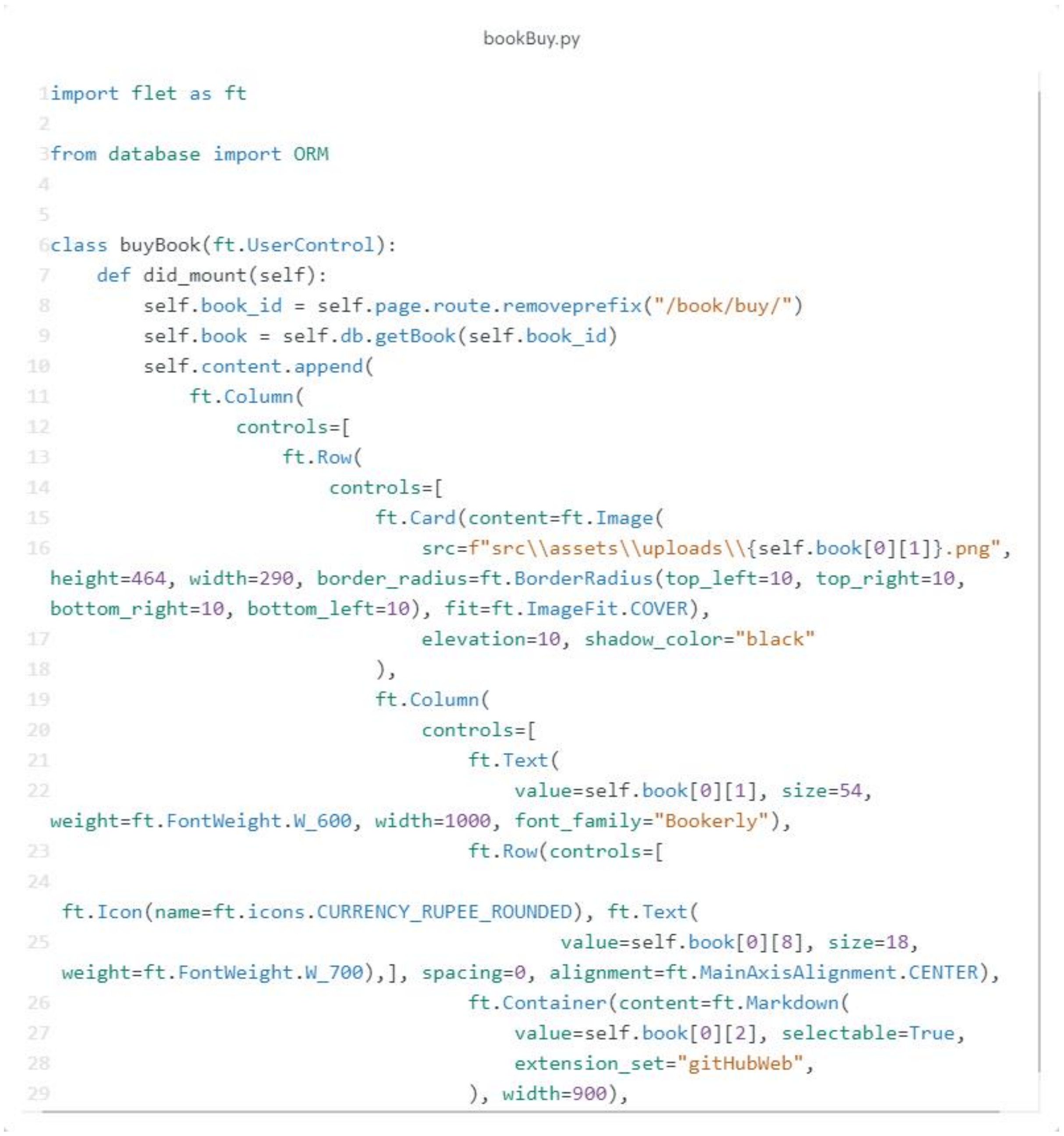
```python
        self.warningSnackBar = ft.SnackBar(
            ft.Text("Something went wrong"))
        return ft.Container(content=ft.Column(controls=[
            self.book_cover,
            self.file_picker,
            self.book_name,
            self.book_desc,
            self.book_year,
            self.book_age_rating,
            self.book_genre,
            self.book_stock,
            self.book_price,
            self.add_book_button,
            self.warningSnackBar
        ], alignment=ft.MainAxisAlignment.CENTER), alignment=ft.alignment.center, padding=50)

    def handlePickFiles(self, e):
        self.page.pick_files_dialog.on_result = self.pick_files_result
        self.page.pick_files_dialog.pick_files(
            allow_multiple=False,
            allowed_extensions=['png']
        ),

    def pick_files_result(self, e: ft.FilePickerResultEvent):
        self.selected_files.value = f"{e.files[0].name} - {e.files[0].size / 1000}kb"
        self.book_cover.src = e.files[0].path
        self.book_cover.visible = True

        self.book_cover.update()
        self.selected_files.update()
```



bookAdd.py

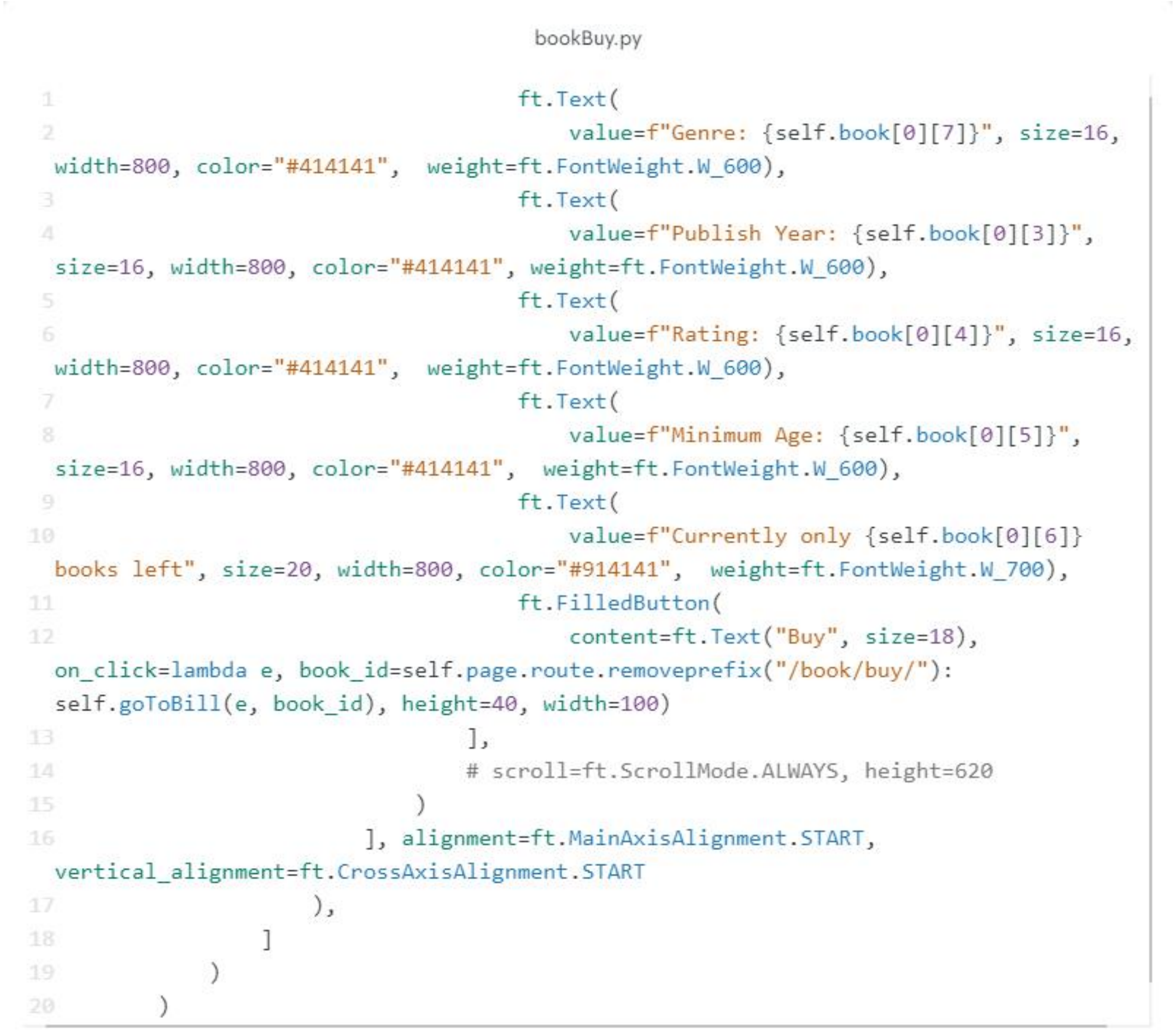
```python
    def upload_files(self):
        upload_list = []
        if self.page.pick_files_dialog.result != None and self.page.pick_files_dialog.result.files != None:
            for f in self.page.pick_files_dialog.result.files:
                upload_list.append(
                    ft.FilePickerUploadFile(
                        f.name,
                        upload_url=self.page.get_upload_url(
                            f"{self.book_name.value}.png", 600),
                        method="PUT"
                    )
                )
            self.page.pick_files_dialog.upload(upload_list)

    def addBookDb(self, e):
        self.warningSnackBar.content = ft.Text("One or more fields empty")
        if self.page.session.get("current_user") == None:
            self.warningSnackBar.content = ft.Text("Please login.")
            self.warningSnackBar.open = True
            self.warningSnackBar.bgcolor = "red"
            self.update()
        elif self.book_name.value == "" or self.book_desc.value == "" or self.book_year.value == "" or self.book_age_rating.value == "" or self.book_genre.value == "" or self.book_stock.value == "" or self.book_price.value == "":
            self.warningSnackBar.open = True
            self.warningSnackBar.bgcolor = "red"
            self.update()
            return
        else:
            try:
                book_id = randint(1_000_000, 9_999_999)
                self.page.db.addBook(self.page.session.get("current_user"), book_id, self.book_name.value, self.book_desc.value, int(
                    self.book_year.value), 0, int(self.book_age_rating.value), int(self.book_stock.value), self.book_genre.value, self.book_price.value)
                self.upload_files()

                self.warningSnackBar.content = ft.Text(
                    "Book added successfully!")
                self.warningSnackBar.open = True
                self.warningSnackBar.bgcolor = "green"
                self.update()
            except:
                print("ERR")
                self.warningSnackBar.open = True
                self.update()
```

File: bookBuy.py



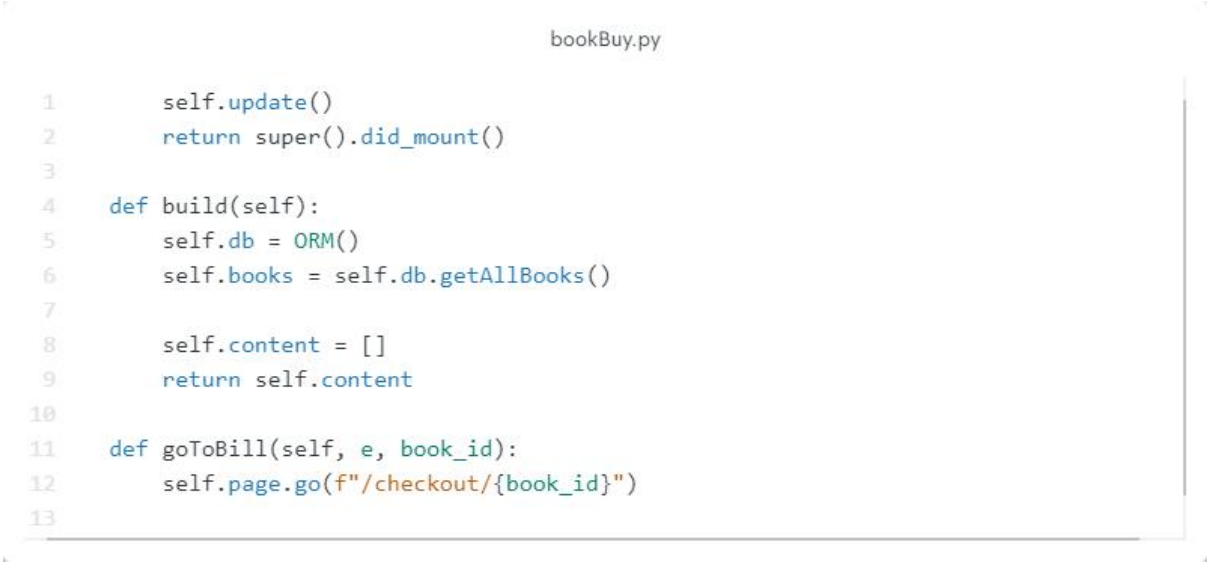
bookBuy.py

```python
import flet as ft

from database import ORM


class buyBook(ft.UserControl):
    def did_mount(self):
        self.book_id = self.page.route.removeprefix("/book/buy/")
        self.book = self.db.getBook(self.book_id)
        self.content.append(
            ft.Column(
                controls=[
                    ft.Row(
                        controls=[
                            ft.Card(content=ft.Image(
                                src=f"src\\assets\\uploads\\{self.book[0][1]}.png",
height=464, width=290, border_radius=ft.BorderRadius(top_left=10, top_right=10,
bottom_right=10, bottom_left=10), fit=ft.ImageFit.COVER),
                                elevation=10, shadow_color="black"
                            ),
                            ft.Column(
                                controls=[
                                    ft.Text(
                                        value=self.book[0][1], size=54,
weight=ft.FontWeight.W_600, width=1000, font_family="Bookerly"),
                                    ft.Row(controls=[

 ft.Icon(name=ft.icons.CURRENCY_RUPEE_ROUNDED), ft.Text(
                                            value=self.book[0][8], size=18,
 weight=ft.FontWeight.W_700),], spacing=0, alignment=ft.MainAxisAlignment.CENTER),
                                    ft.Container(content=ft.Markdown(
                                        value=self.book[0][2], selectable=True,
                                        extension_set="gitHubWeb",
                                    ), width=900),
```



bookBuy.py

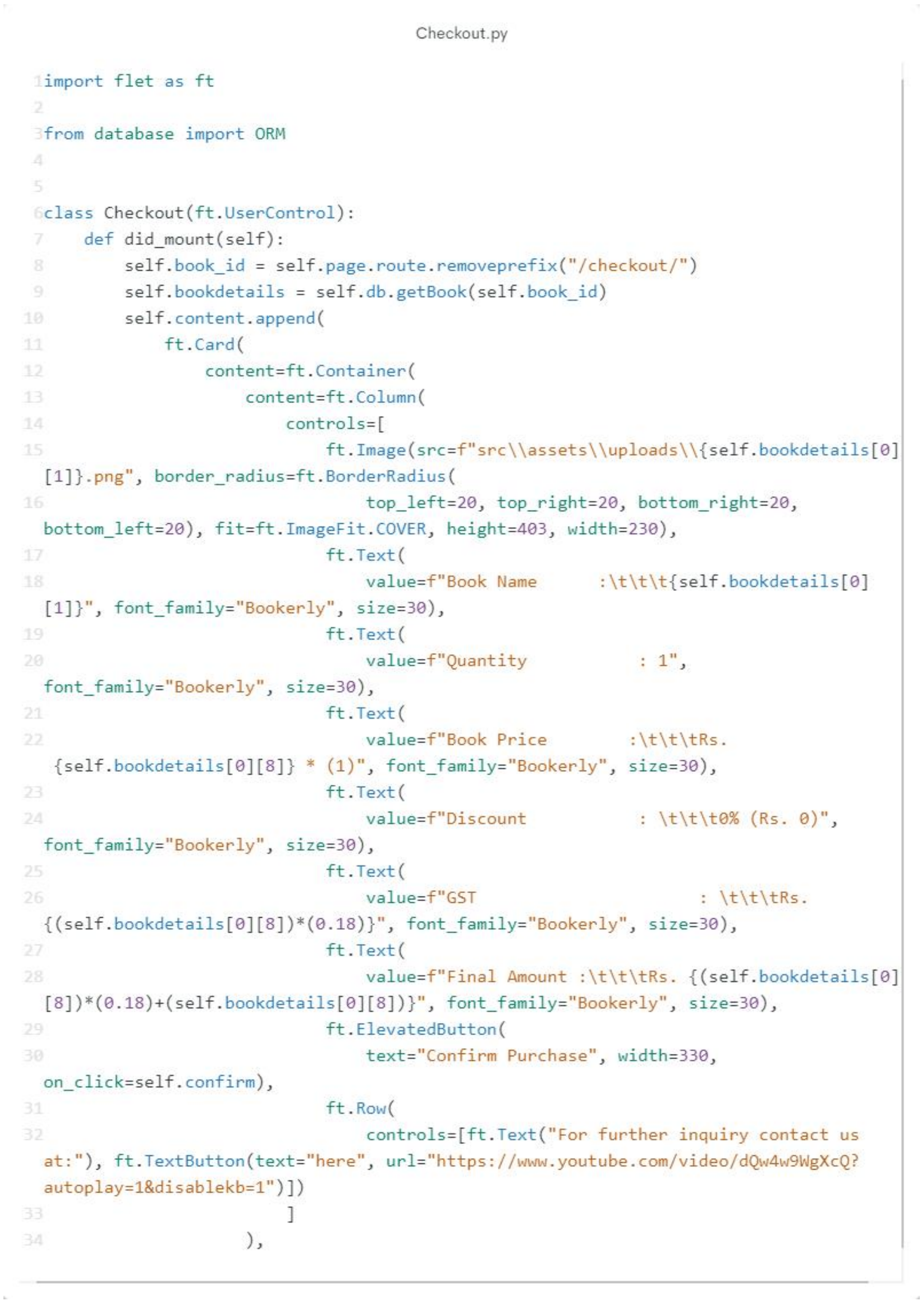
```python
                                    ft.Text(
                                        value=f"Genre: {self.book[0][7]}", size=16, width=800, color="#414141",  weight=ft.FontWeight.W_600),
                                    ft.Text(
                                        value=f"Publish Year: {self.book[0][3]}", size=16, width=800, color="#414141", weight=ft.FontWeight.W_600),
                                    ft.Text(
                                        value=f"Rating: {self.book[0][4]}", size=16, width=800, color="#414141",  weight=ft.FontWeight.W_600),
                                    ft.Text(
                                        value=f"Minimum Age: {self.book[0][5]}", size=16, width=800, color="#414141",  weight=ft.FontWeight.W_600),
                                    ft.Text(
                                        value=f"Currently only {self.book[0][6]} books left", size=20, width=800, color="#914141",  weight=ft.FontWeight.W_700),
                                    ft.FilledButton(
                                        content=ft.Text("Buy", size=18), on_click=lambda e, book_id=self.page.route.removeprefix("/book/buy/"): self.goToBill(e, book_id), height=40, width=100)
                                ],
                                # scroll=ft.ScrollMode.ALWAYS, height=620
                            )
                        ], alignment=ft.MainAxisAlignment.START, vertical_alignment=ft.CrossAxisAlignment.START
                    ),
                ]
            )
        )
```



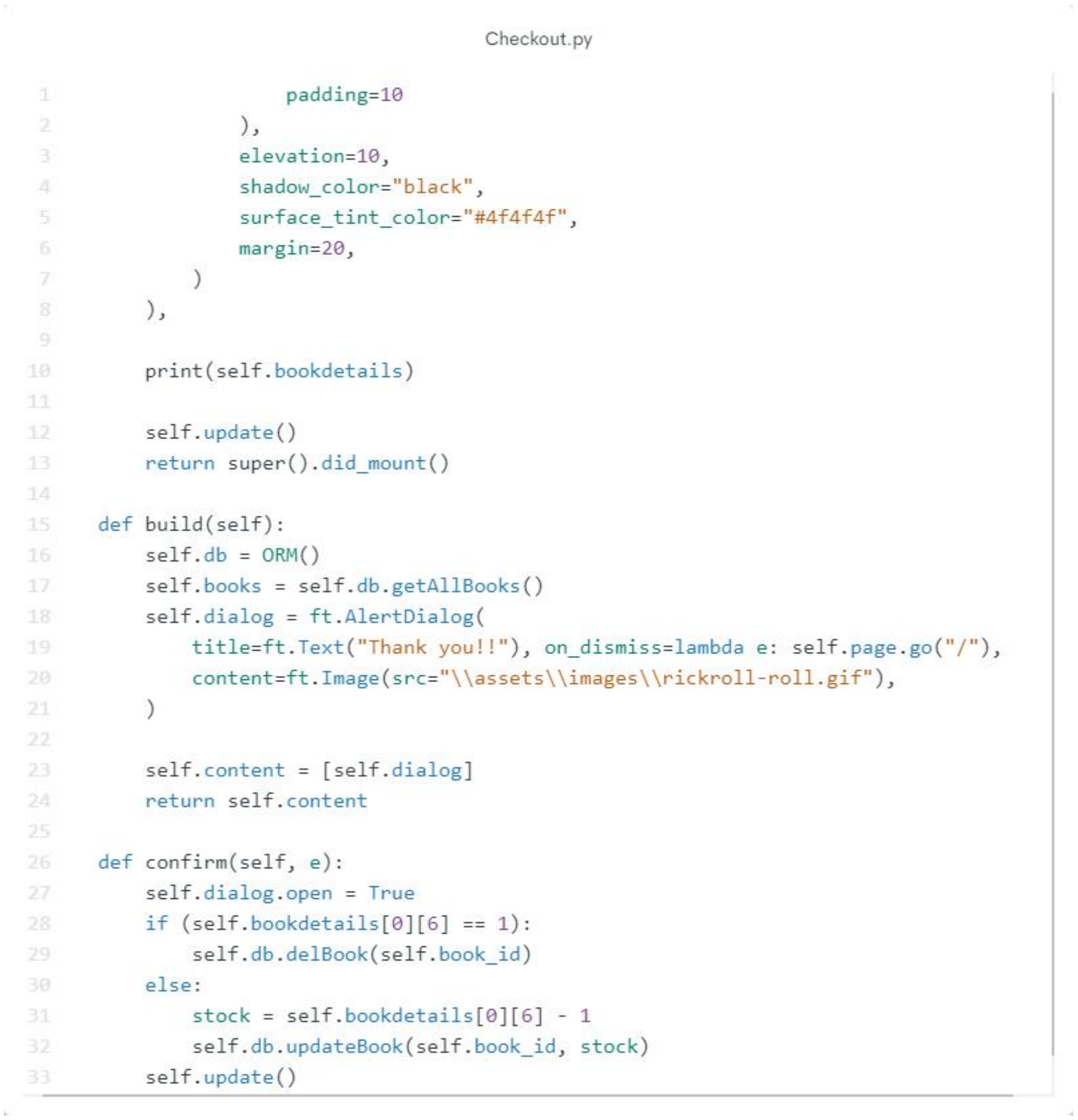
bookBuy.py

```python
        self.update()
        return super().did_mount()

    def build(self):
        self.db = ORM()
        self.books = self.db.getAllBooks()

        self.content = []
        return self.content

    def goToBill(self, e, book_id):
        self.page.go(f"/checkout/{book_id}")

```

File: Checkout.py


Checkout.py

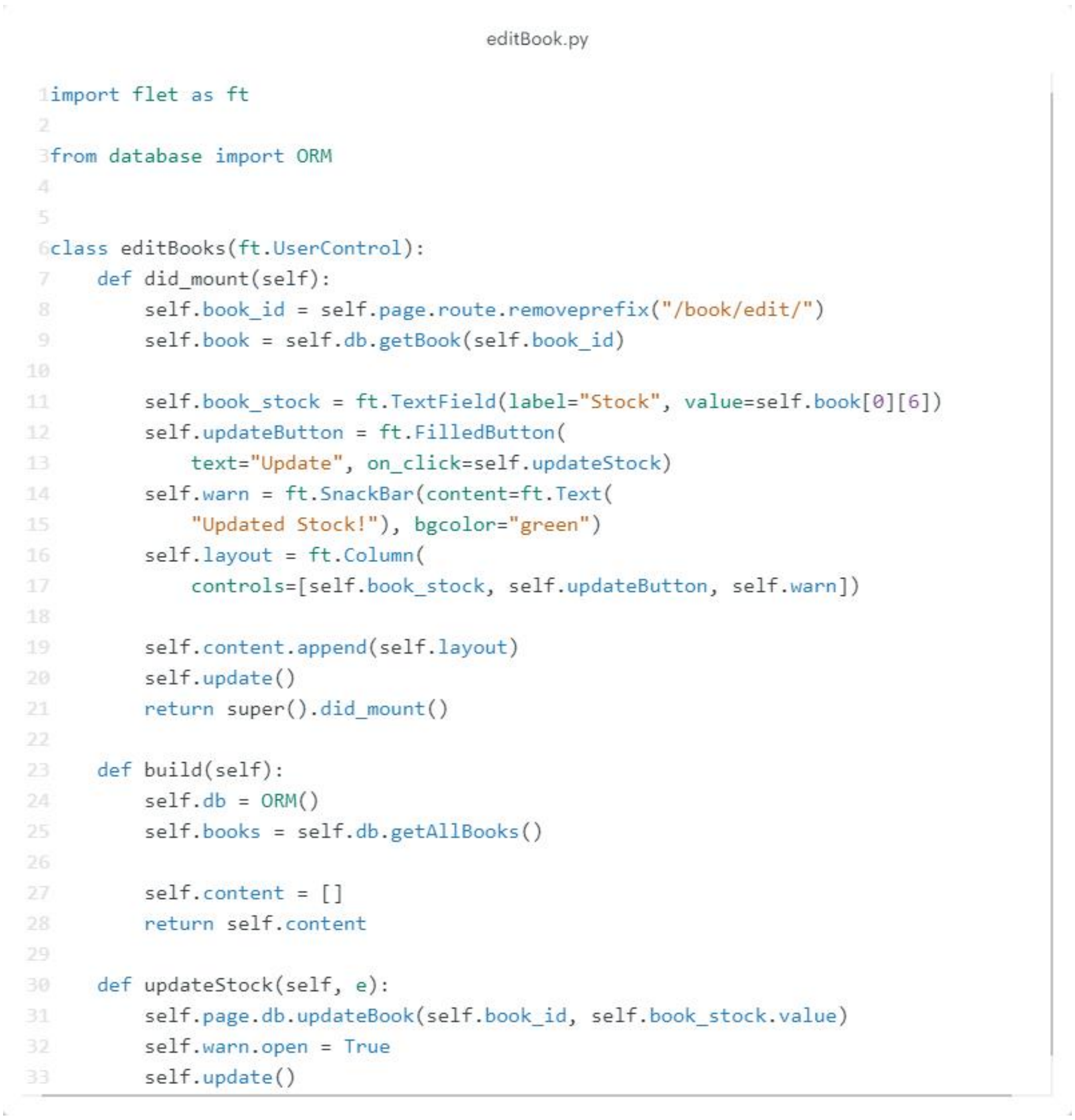
```python
import flet as ft

from database import ORM


class Checkout(ft.UserControl):
    def did_mount(self):
        self.book_id = self.page.route.removeprefix("/checkout/")
        self.bookdetails = self.db.getBook(self.book_id)
        self.content.append(
            ft.Card(
                content=ft.Container(
                    content=ft.Column(
                        controls=[
                            ft.Image(src=f"src\\assets\\uploads\\{self.bookdetails[0][1]}.png", border_radius=ft.BorderRadius(
                                top_left=20, top_right=20, bottom_right=20, bottom_left=20), fit=ft.ImageFit.COVER, height=403, width=230),
                            ft.Text(
                                value=f"Book Name      :\t\t\t{self.bookdetails[0][1]}", font_family="Bookerly", size=30),
                            ft.Text(
                                value=f"Quantity           : 1", font_family="Bookerly", size=30),
                            ft.Text(
                                value=f"Book Price        :\t\t\tRs. {self.bookdetails[0][8]} * (1)", font_family="Bookerly", size=30),
                            ft.Text(
                                value=f"Discount           : \t\t\t0% (Rs. 0)", font_family="Bookerly", size=30),
                            ft.Text(
                                value=f"GST                        : \t\t\tRs. {(self.bookdetails[0][8])*(0.18)}", font_family="Bookerly", size=30),
                            ft.Text(
                                value=f"Final Amount :\t\t\tRs. {(self.bookdetails[0][8])*(0.18)+(self.bookdetails[0][8])}", font_family="Bookerly", size=30),
                            ft.ElevatedButton(
                                text="Confirm Purchase", width=330, on_click=self.confirm),
                            ft.Row(
                                controls=[ft.Text("For further inquiry contact us at:"), ft.TextButton(text="here", url="https://www.youtube.com/video/dQw4w9WgXcQ?autoplay=1&disablekb=1")])
                        ]
                    ),
```



Checkout.py

```python
                    padding=10
                ),
                elevation=10,
                shadow_color="black",
                surface_tint_color="#4f4f4f",
                margin=20,
            )
        ),

        print(self.bookdetails)

        self.update()
        return super().did_mount()

    def build(self):
        self.db = ORM()
        self.books = self.db.getAllBooks()
        self.dialog = ft.AlertDialog(
            title=ft.Text("Thank you!!"), on_dismiss=lambda e: self.page.go("/"),
            content=ft.Image(src="\\assets\\images\\rickroll-roll.gif"),
        )

        self.content = [self.dialog]
        return self.content

    def confirm(self, e):
        self.dialog.open = True
        if (self.bookdetails[0][6] == 1):
            self.db.delBook(self.book_id)
        else:
            stock = self.bookdetails[0][6] - 1
            self.db.updateBook(self.book_id, stock)
        self.update()
```

File: editBooks.py

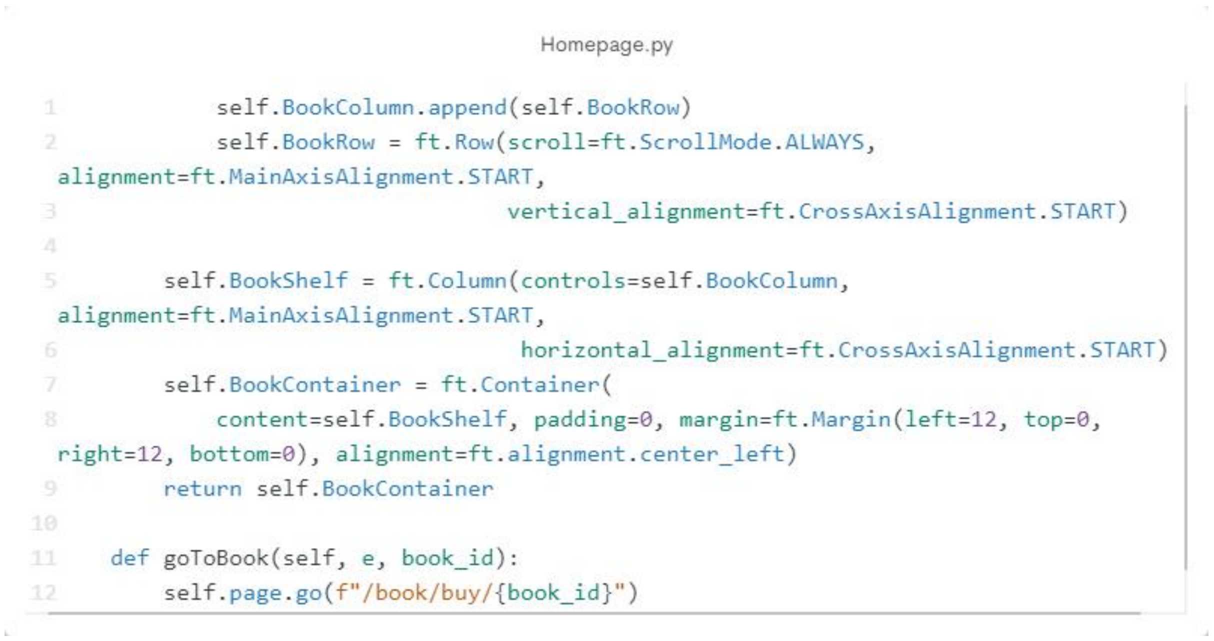

editBook.py

```python
import flet as ft

from database import ORM


class editBooks(ft.UserControl):
    def did_mount(self):
        self.book_id = self.page.route.removeprefix("/book/edit/")
        self.book = self.db.getBook(self.book_id)

        self.book_stock = ft.TextField(label="Stock", value=self.book[0][6])
        self.updateButton = ft.FilledButton(
            text="Update", on_click=self.updateStock)
        self.warn = ft.SnackBar(content=ft.Text(
            "Updated Stock!"), bgcolor="green")
        self.layout = ft.Column(
            controls=[self.book_stock, self.updateButton, self.warn])

        self.content.append(self.layout)
        self.update()
        return super().did_mount()

    def build(self):
        self.db = ORM()
        self.books = self.db.getAllBooks()

        self.content = []
        return self.content

    def updateStock(self, e):
        self.page.db.updateBook(self.book_id, self.book_stock.value)
        self.warn.open = True
        self.update()
```

File: Homepage.py

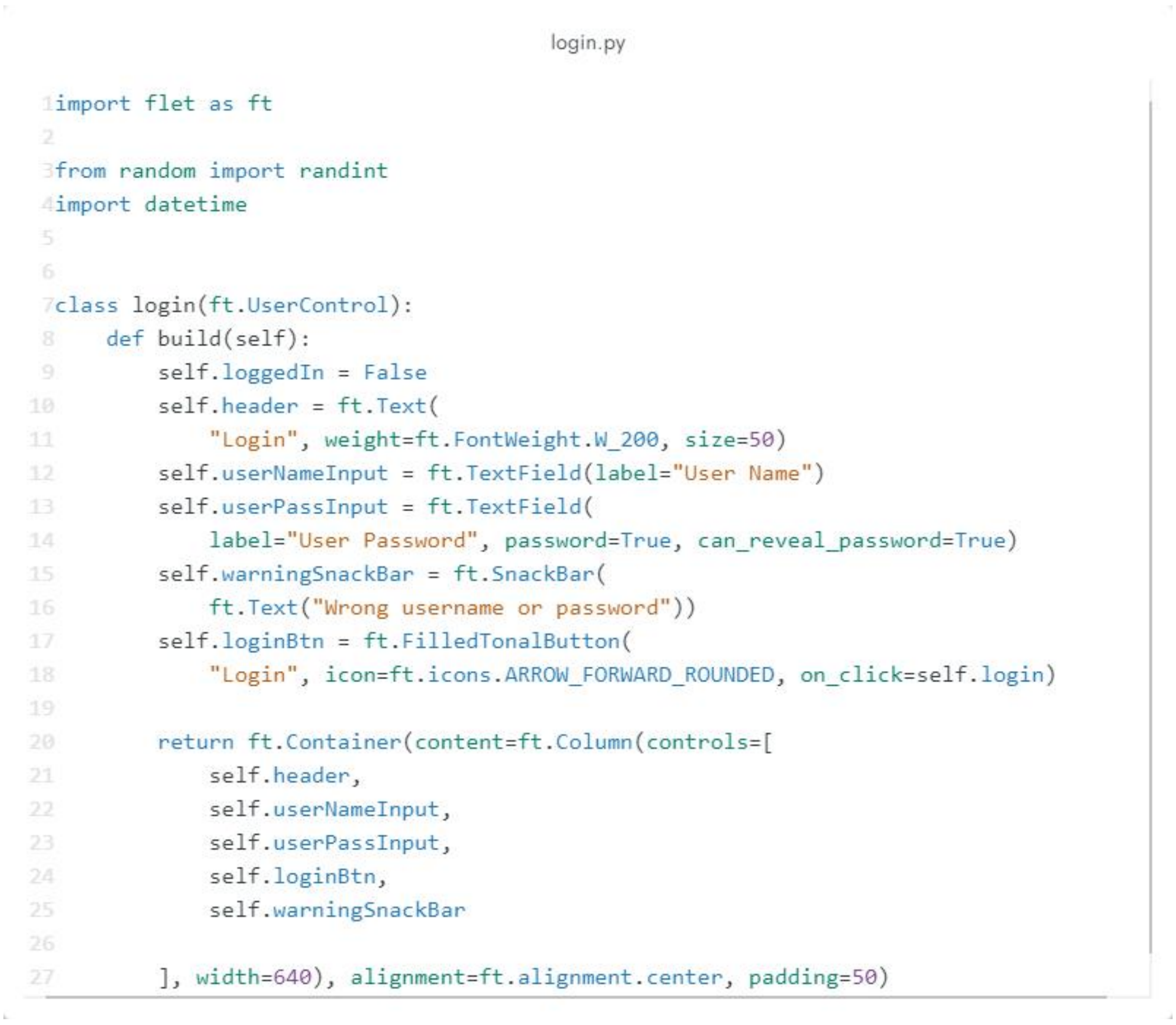
Homepage.py

```
import flet as ft

from database import ORM


class HomePage(ft.UserControl):

    def build(self):
        self.db = ORM()
        self.books = self.db.getAllBooks()
        self.genre = self.db.getGenre()
        self.BookRow = ft.Row(scroll=ft.ScrollMode.ALWAYS,
alignment=ft.MainAxisAlignment.START,
                              vertical_alignment=ft.CrossAxisAlignment.START)
        self.BookColumn = []

        for j in self.genre:
            self.BookColumn.append(ft.Text(
                value=f"| {j[0]}", weight=ft.FontWeight.W_600,
text_align=ft.TextAlign.END, size=28))
            for i in self.books:
                if i[7] == j[0]:
                    self.BookRow.controls.append(
                        ft.TextButton(
                            content=ft.Column(controls=[
                                ft.Image(src=f"\\assets\\uploads\\{i[1]}.png",
height=260,
                                         fit=ft.ImageFit.COVER,
border_radius=ft.BorderRadius(10, 10, 10, 10)),
                                ft.Row(controls=[
                                    ft.Icon(name=ft.icons.CURRENCY_RUPEE_ROUNDED),
ft.Text(
                                        value=i[8], size=18,
 weight=ft.FontWeight.W_700),], spacing=0, alignment=ft.MainAxisAlignment.START),
                                ft.Text(value=i[1], size=18,
weight=ft.FontWeight.W_500,
                                        text_align=ft.TextAlign.START),
                            ], alignment=ft.MainAxisAlignment.CENTER),
style=ft.ButtonStyle(bgcolor="#f0f0f0", color="black", shape={
                                ft.MaterialState.DEFAULT:
ft.RoundedRectangleBorder(radius=10),
                            }, padding=15), on_click=lambda e, book_id=i[0]:
self.goToBook(e, book_id), width=190)
                    )
```

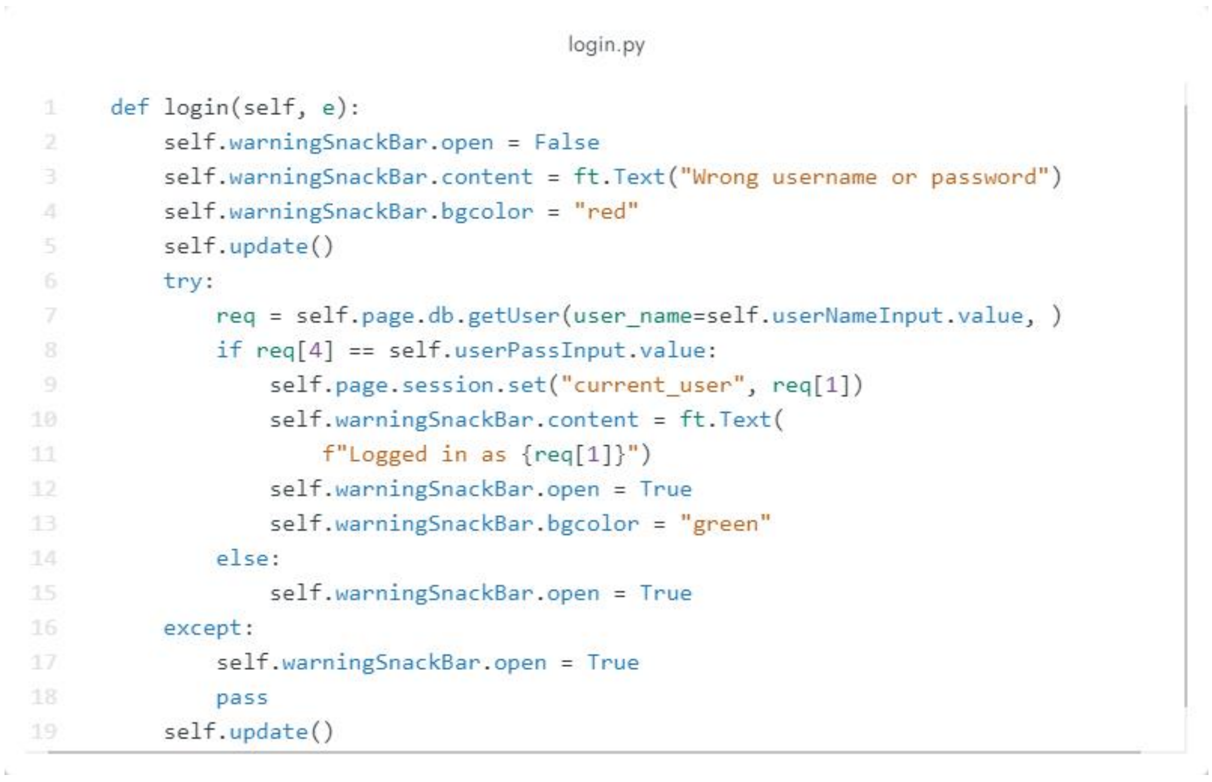

Homepage.py

```python
            self.BookColumn.append(self.BookRow)
            self.BookRow = ft.Row(scroll=ft.ScrollMode.ALWAYS, alignment=ft.MainAxisAlignment.START,
                                  vertical_alignment=ft.CrossAxisAlignment.START)

        self.BookShelf = ft.Column(controls=self.BookColumn, alignment=ft.MainAxisAlignment.START,
                                   horizontal_alignment=ft.CrossAxisAlignment.START)
        self.BookContainer = ft.Container(
            content=self.BookShelf, padding=0, margin=ft.Margin(left=12, top=0, right=12, bottom=0), alignment=ft.alignment.center_left)
        return self.BookContainer

    def goToBook(self, e, book_id):
        self.page.go(f"/book/buy/{book_id}")
```

File: login.py

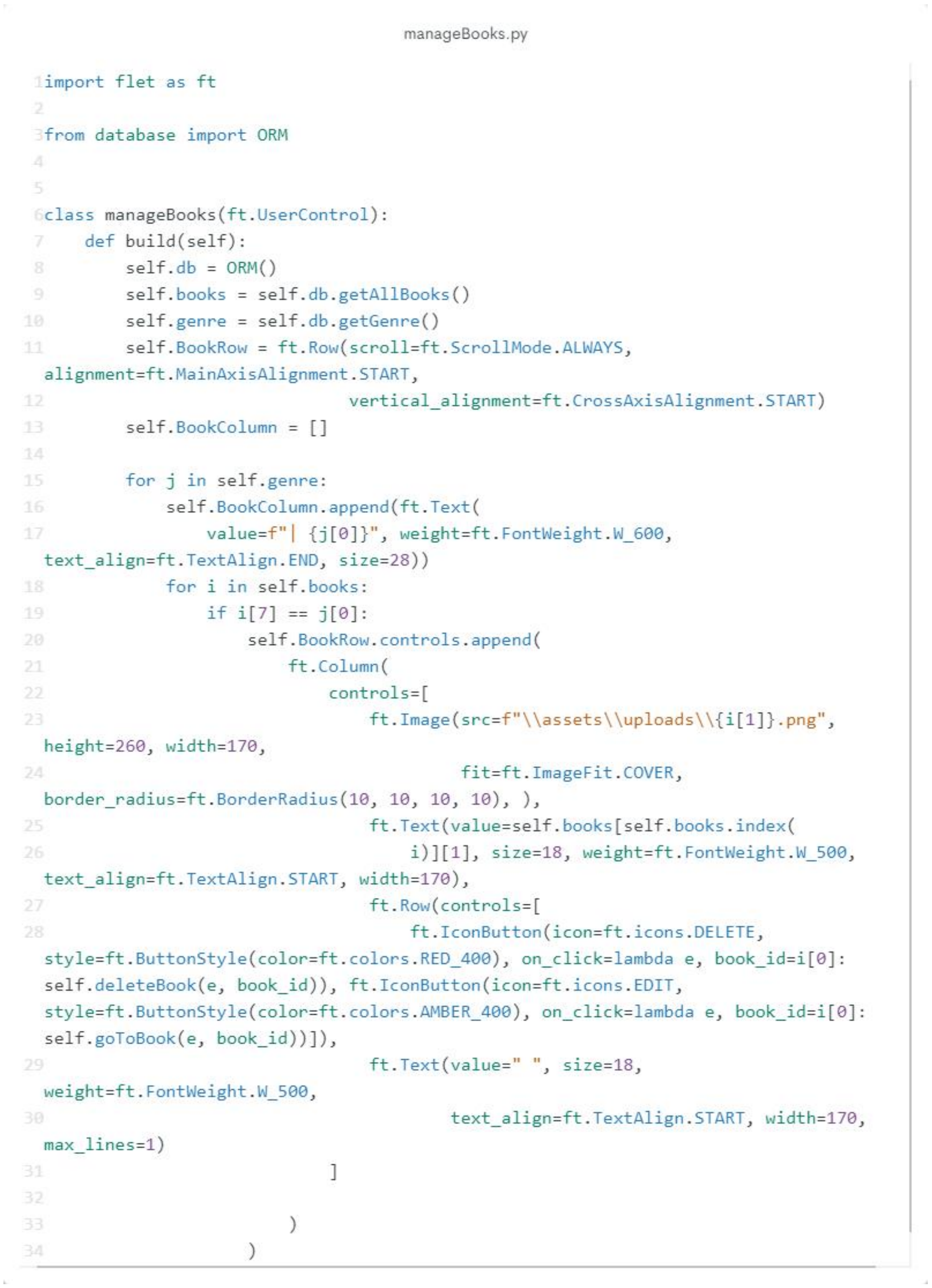

login.py

```python
import flet as ft

from random import randint
import datetime


class login(ft.UserControl):
    def build(self):
        self.loggedIn = False
        self.header = ft.Text(
            "Login", weight=ft.FontWeight.W_200, size=50)
        self.userNameInput = ft.TextField(label="User Name")
        self.userPassInput = ft.TextField(
            label="User Password", password=True, can_reveal_password=True)
        self.warningSnackBar = ft.SnackBar(
            ft.Text("Wrong username or password"))
        self.loginBtn = ft.FilledTonalButton(
            "Login", icon=ft.icons.ARROW_FORWARD_ROUNDED, on_click=self.login)

        return ft.Container(content=ft.Column(controls=[
            self.header,
            self.userNameInput,
            self.userPassInput,
            self.loginBtn,
            self.warningSnackBar

        ], width=640), alignment=ft.alignment.center, padding=50)
```

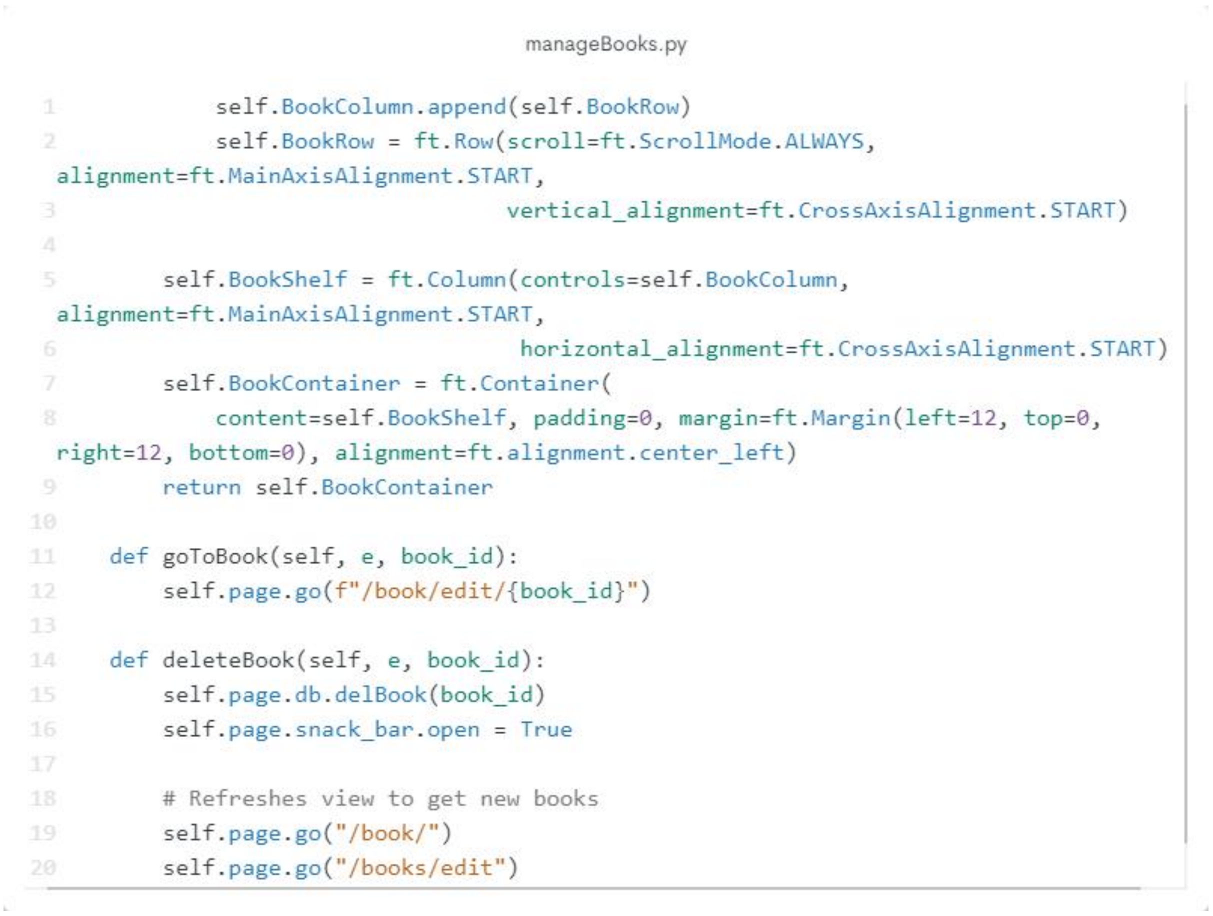

login.py

```python
def login(self, e):
    self.warningSnackBar.open = False
    self.warningSnackBar.content = ft.Text("Wrong username or password")
    self.warningSnackBar.bgcolor = "red"
    self.update()
    try:
        req = self.page.db.getUser(user_name=self.userNameInput.value, )
        if req[4] == self.userPassInput.value:
            self.page.session.set("current_user", req[1])
            self.warningSnackBar.content = ft.Text(
                f"Logged in as {req[1]}")
            self.warningSnackBar.open = True
            self.warningSnackBar.bgcolor = "green"
        else:
            self.warningSnackBar.open = True
    except:
        self.warningSnackBar.open = True
        pass
    self.update()
```

File: manageBooks.py


manageBooks.py

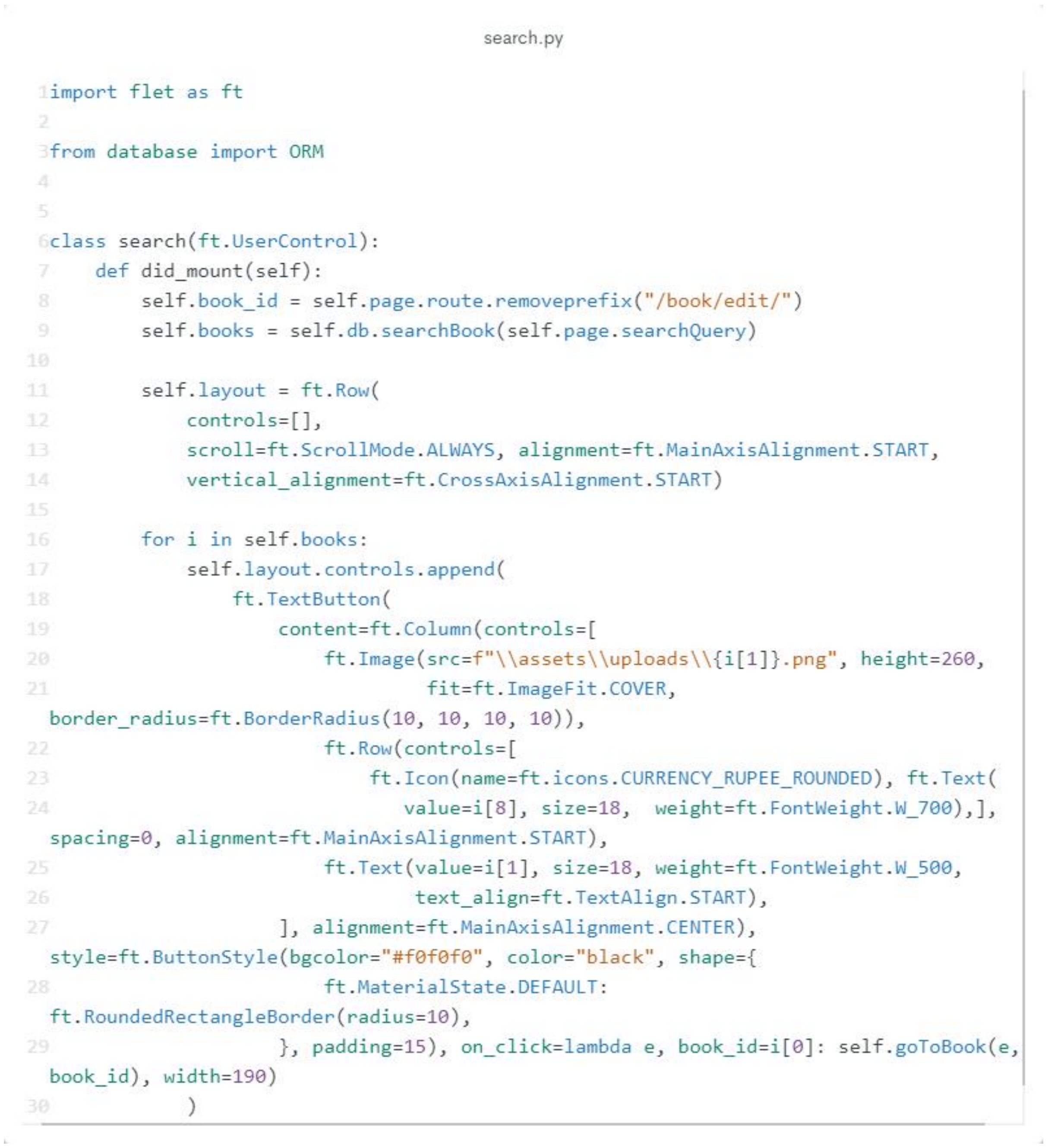
```python
import flet as ft

from database import ORM


class manageBooks(ft.UserControl):
    def build(self):
        self.db = ORM()
        self.books = self.db.getAllBooks()
        self.genre = self.db.getGenre()
        self.BookRow = ft.Row(scroll=ft.ScrollMode.ALWAYS, alignment=ft.MainAxisAlignment.START,
                              vertical_alignment=ft.CrossAxisAlignment.START)
        self.BookColumn = []

        for j in self.genre:
            self.BookColumn.append(ft.Text(
                value=f"| {j[0]}", weight=ft.FontWeight.W_600, text_align=ft.TextAlign.END, size=28))
            for i in self.books:
                if i[7] == j[0]:
                    self.BookRow.controls.append(
                        ft.Column(
                            controls=[
                                ft.Image(src=f"\\assets\\uploads\\{i[1]}.png", height=260, width=170,
                                         fit=ft.ImageFit.COVER, border_radius=ft.BorderRadius(10, 10, 10, 10), ),
                                ft.Text(value=self.books[self.books.index(
                                    i)][1], size=18, weight=ft.FontWeight.W_500, text_align=ft.TextAlign.START, width=170),
                                ft.Row(controls=[
                                    ft.IconButton(icon=ft.icons.DELETE, style=ft.ButtonStyle(color=ft.colors.RED_400), on_click=lambda e, book_id=i[0]: self.deleteBook(e, book_id)), ft.IconButton(icon=ft.icons.EDIT, style=ft.ButtonStyle(color=ft.colors.AMBER_400), on_click=lambda e, book_id=i[0]: self.goToBook(e, book_id))]),
                                ft.Text(value=" ", size=18, weight=ft.FontWeight.W_500,
                                        text_align=ft.TextAlign.START, width=170, max_lines=1)
                            ]

                        )
                    )
```



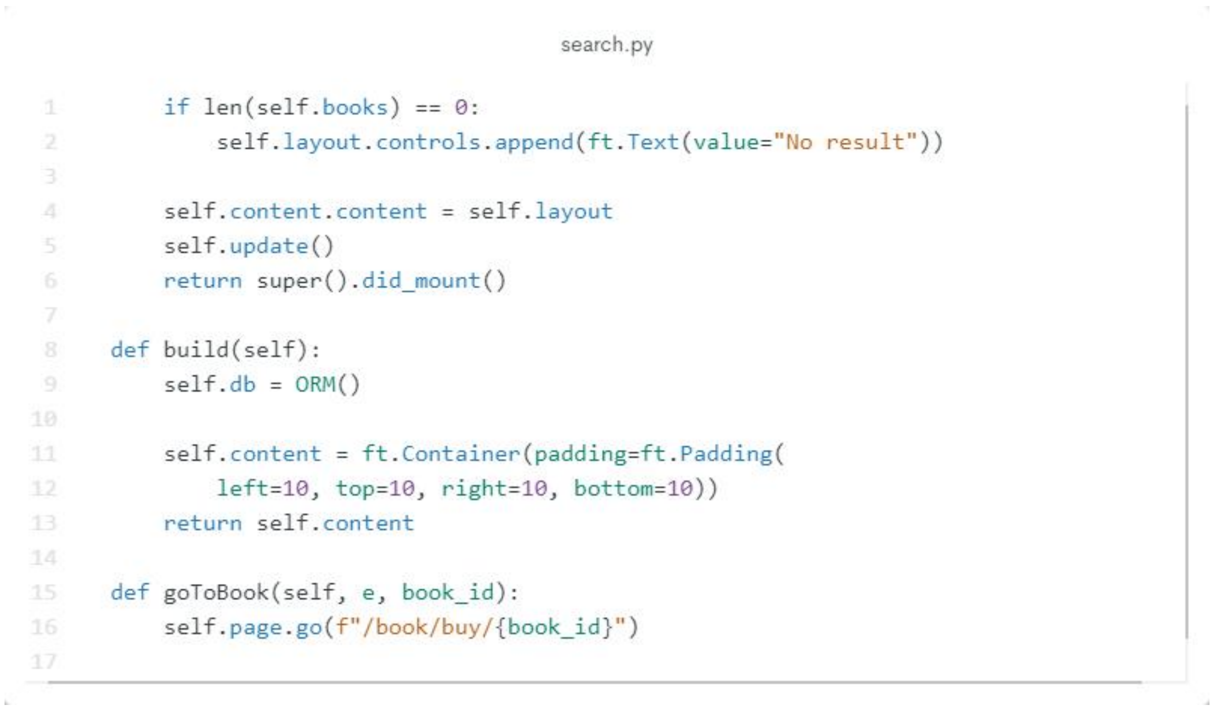
manageBooks.py

```python
            self.BookColumn.append(self.BookRow)
            self.BookRow = ft.Row(scroll=ft.ScrollMode.ALWAYS, alignment=ft.MainAxisAlignment.START,
                                  vertical_alignment=ft.CrossAxisAlignment.START)

        self.BookShelf = ft.Column(controls=self.BookColumn, alignment=ft.MainAxisAlignment.START,
                                   horizontal_alignment=ft.CrossAxisAlignment.START)
        self.BookContainer = ft.Container(
            content=self.BookShelf, padding=0, margin=ft.Margin(left=12, top=0, right=12, bottom=0), alignment=ft.alignment.center_left)
        return self.BookContainer

    def goToBook(self, e, book_id):
        self.page.go(f"/book/edit/{book_id}")

    def deleteBook(self, e, book_id):
        self.page.db.delBook(book_id)
        self.page.snack_bar.open = True

        # Refreshes view to get new books
        self.page.go("/book/")
        self.page.go("/books/edit")
```

File: search.py


search.py

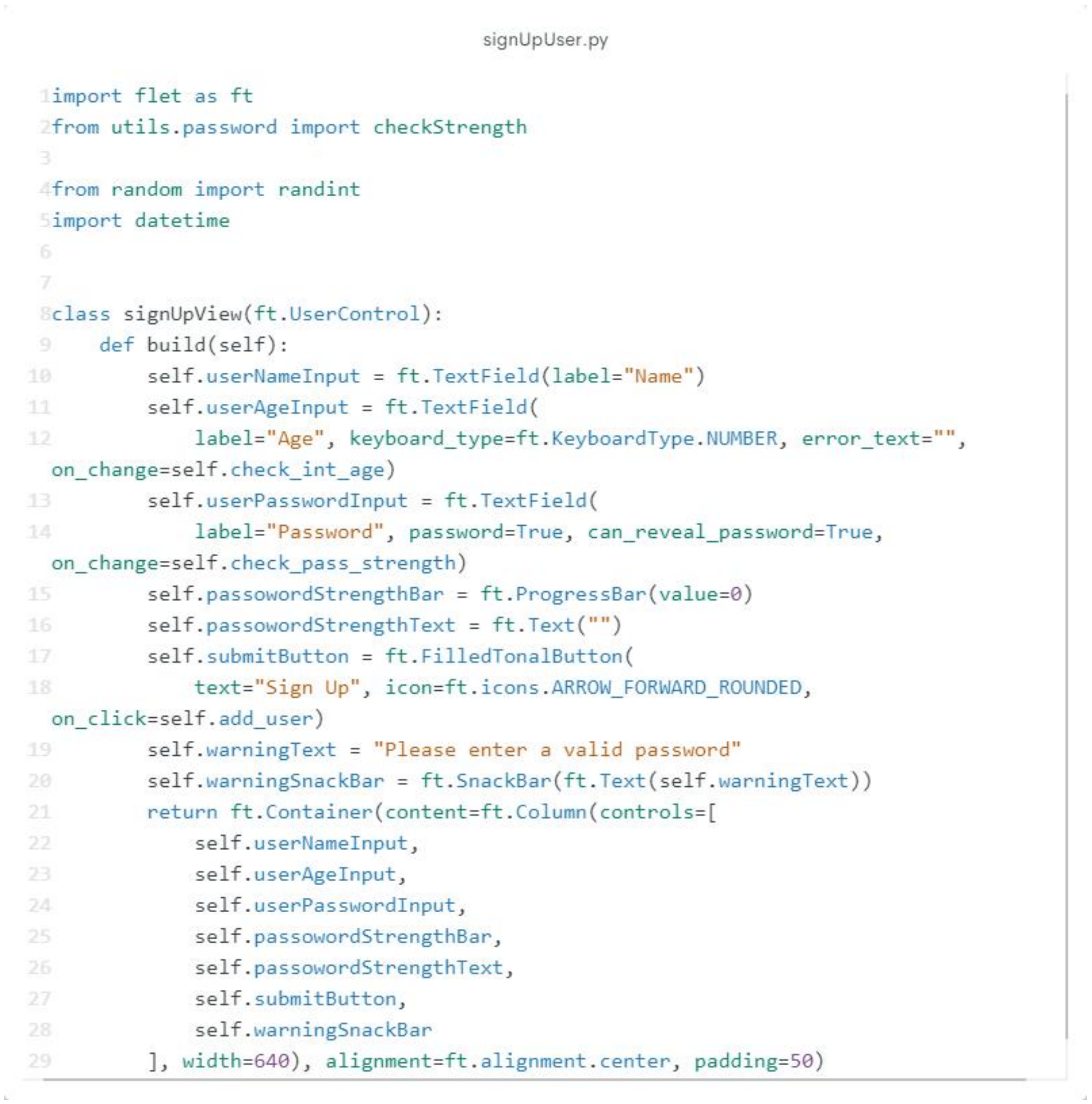
```
import flet as ft

from database import ORM


class search(ft.UserControl):
    def did_mount(self):
        self.book_id = self.page.route.removeprefix("/book/edit/")
        self.books = self.db.searchBook(self.page.searchQuery)

        self.layout = ft.Row(
            controls=[],
            scroll=ft.ScrollMode.ALWAYS, alignment=ft.MainAxisAlignment.START,
            vertical_alignment=ft.CrossAxisAlignment.START)

        for i in self.books:
            self.layout.controls.append(
                ft.TextButton(
                    content=ft.Column(controls=[
                        ft.Image(src=f"\\assets\\uploads\\{i[1]}.png", height=260,
                                 fit=ft.ImageFit.COVER,
border_radius=ft.BorderRadius(10, 10, 10, 10)),
                        ft.Row(controls=[
                            ft.Icon(name=ft.icons.CURRENCY_RUPEE_ROUNDED), ft.Text(
                               value=i[8], size=18,  weight=ft.FontWeight.W_700),],
spacing=0, alignment=ft.MainAxisAlignment.START),
                        ft.Text(value=i[1], size=18, weight=ft.FontWeight.W_500,
                                text_align=ft.TextAlign.START),
                    ], alignment=ft.MainAxisAlignment.CENTER),
style=ft.ButtonStyle(bgcolor="#f0f0f0", color="black", shape={
                        ft.MaterialState.DEFAULT:
ft.RoundedRectangleBorder(radius=10),
                    }, padding=15), on_click=lambda e, book_id=i[0]: self.goToBook(e,
book_id), width=190)
            )
```



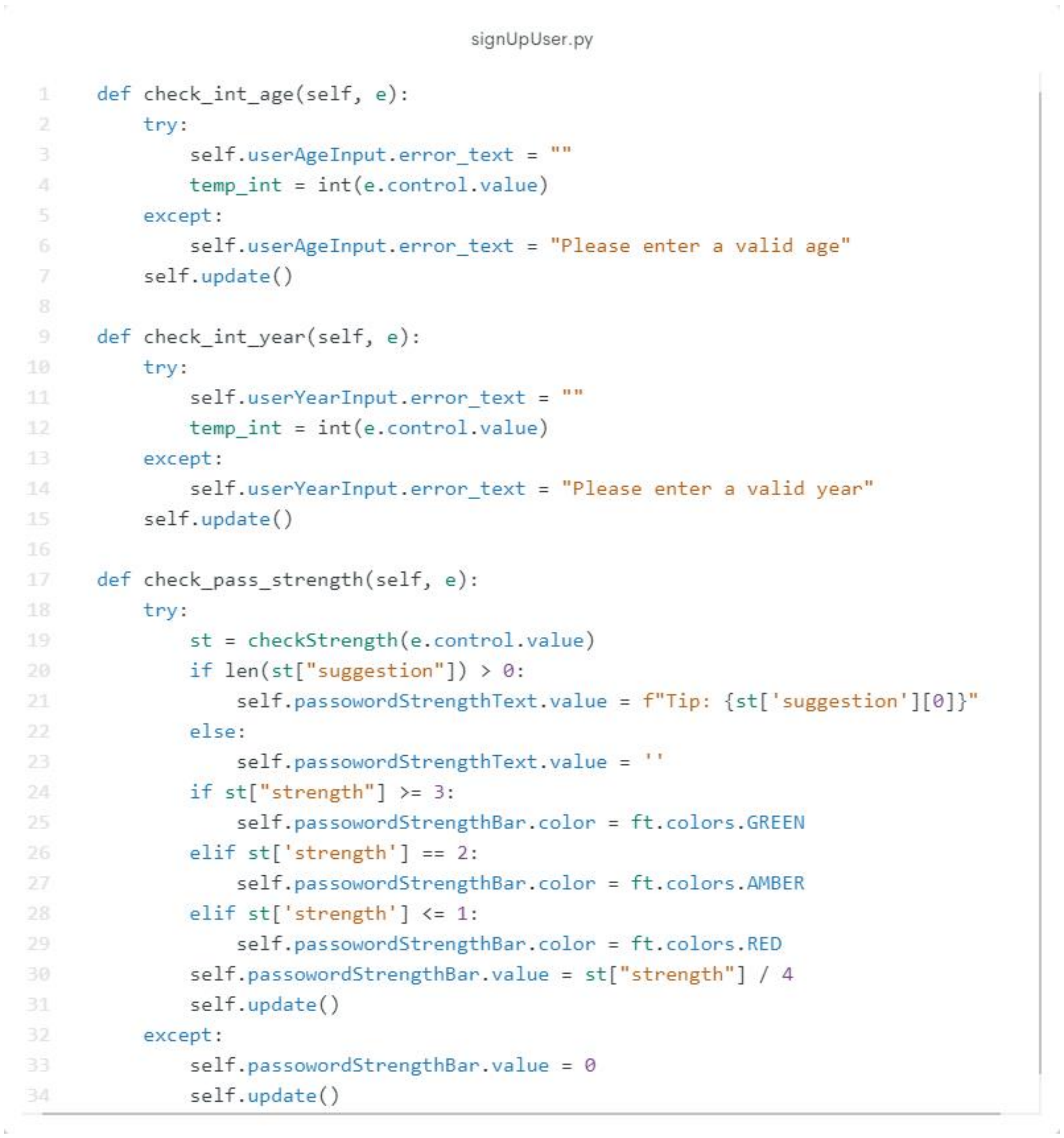
search.py

```python
        if len(self.books) == 0:
            self.layout.controls.append(ft.Text(value="No result"))

        self.content.content = self.layout
        self.update()
        return super().did_mount()

    def build(self):
        self.db = ORM()

        self.content = ft.Container(padding=ft.Padding(
            left=10, top=10, right=10, bottom=10))
        return self.content

    def goToBook(self, e, book_id):
        self.page.go(f"/book/buy/{book_id}")

```

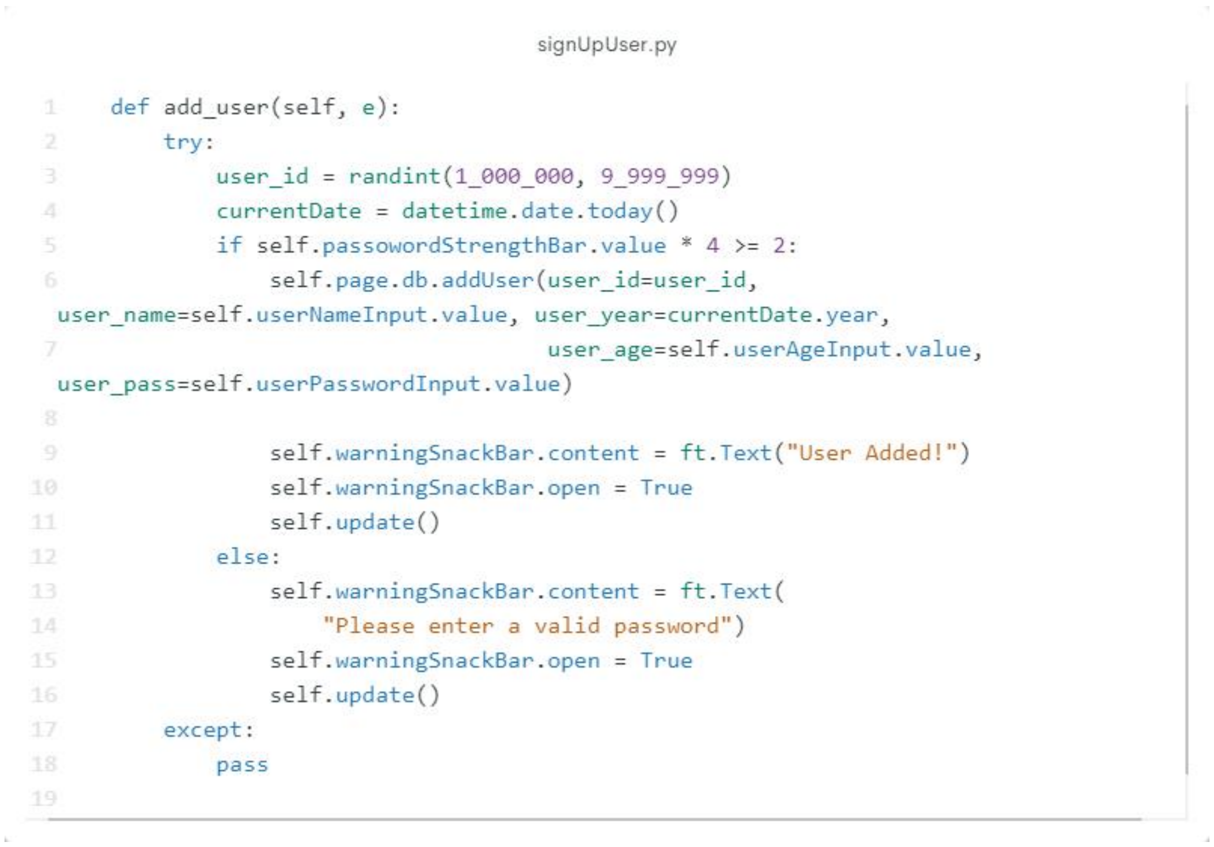
File: signUpUser.py


signUpUser.py

```python
import flet as ft
from utils.password import checkStrength

from random import randint
import datetime


class signUpView(ft.UserControl):
    def build(self):
        self.userNameInput = ft.TextField(label="Name")
        self.userAgeInput = ft.TextField(
            label="Age", keyboard_type=ft.KeyboardType.NUMBER, error_text="", on_change=self.check_int_age)
        self.userPasswordInput = ft.TextField(
            label="Password", password=True, can_reveal_password=True, on_change=self.check_pass_strength)
        self.passowordStrengthBar = ft.ProgressBar(value=0)
        self.passowordStrengthText = ft.Text("")
        self.submitButton = ft.FilledTonalButton(
            text="Sign Up", icon=ft.icons.ARROW_FORWARD_ROUNDED, on_click=self.add_user)
        self.warningText = "Please enter a valid password"
        self.warningSnackBar = ft.SnackBar(ft.Text(self.warningText))
        return ft.Container(content=ft.Column(controls=[
            self.userNameInput,
            self.userAgeInput,
            self.userPasswordInput,
            self.passowordStrengthBar,
            self.passowordStrengthText,
            self.submitButton,
            self.warningSnackBar
        ], width=640), alignment=ft.alignment.center, padding=50)
```

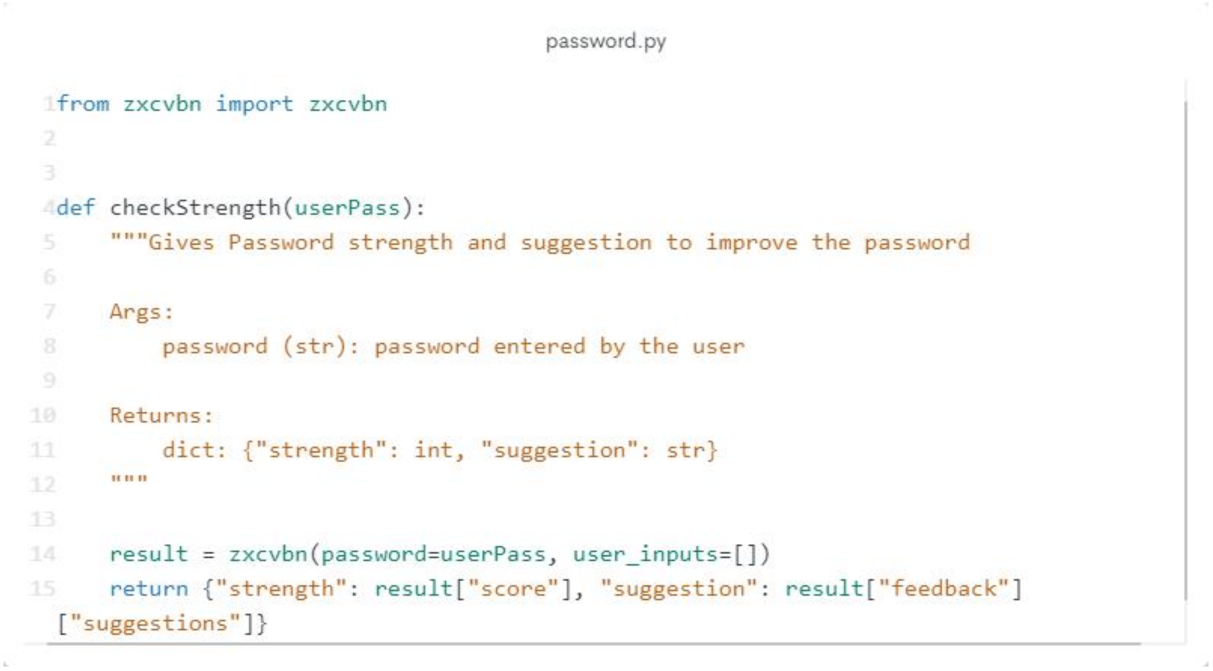

signUpUser.py

```python
def check_int_age(self, e):
    try:
        self.userAgeInput.error_text = ""
        temp_int = int(e.control.value)
    except:
        self.userAgeInput.error_text = "Please enter a valid age"
    self.update()

def check_int_year(self, e):
    try:
        self.userYearInput.error_text = ""
        temp_int = int(e.control.value)
    except:
        self.userYearInput.error_text = "Please enter a valid year"
    self.update()

def check_pass_strength(self, e):
    try:
        st = checkStrength(e.control.value)
        if len(st["suggestion"]) > 0:
            self.passowordStrengthText.value = f"Tip: {st['suggestion'][0]}"
        else:
            self.passowordStrengthText.value = ''
        if st["strength"] >= 3:
            self.passowordStrengthBar.color = ft.colors.GREEN
        elif st['strength'] == 2:
            self.passowordStrengthBar.color = ft.colors.AMBER
        elif st['strength'] <= 1:
            self.passowordStrengthBar.color = ft.colors.RED
        self.passowordStrengthBar.value = st["strength"] / 4
        self.update()
    except:
        self.passowordStrengthBar.value = 0
        self.update()
```


signUpUser.py

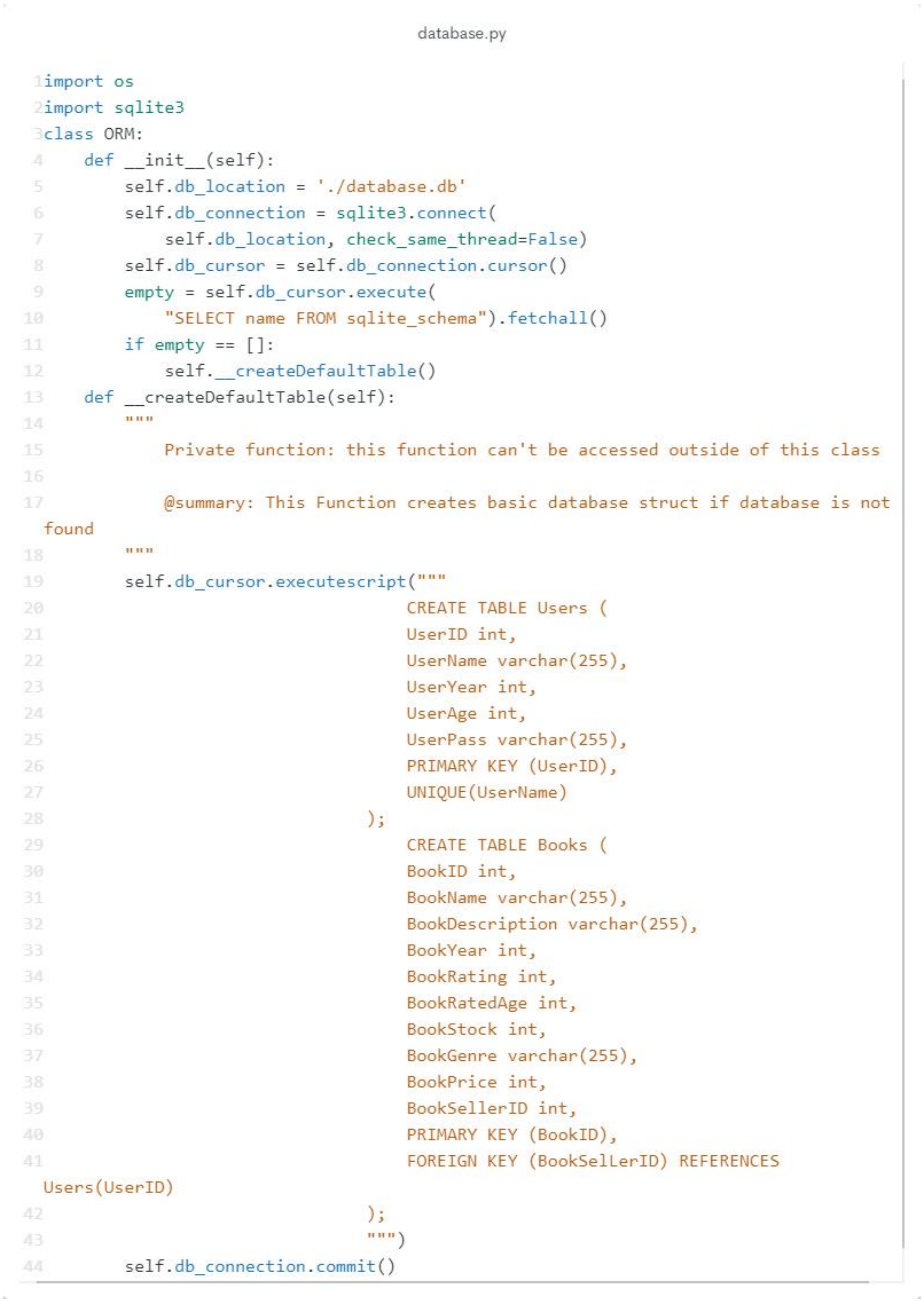
```python
def add_user(self, e):
    try:
        user_id = randint(1_000_000, 9_999_999)
        currentDate = datetime.date.today()
        if self.passowordStrengthBar.value * 4 >= 2:
            self.page.db.addUser(user_id=user_id, user_name=self.userNameInput.value, user_year=currentDate.year,
                                 user_age=self.userAgeInput.value, user_pass=self.userPasswordInput.value)

            self.warningSnackBar.content = ft.Text("User Added!")
            self.warningSnackBar.open = True
            self.update()
        else:
            self.warningSnackBar.content = ft.Text(
                "Please enter a valid password")
            self.warningSnackBar.open = True
            self.update()
    except:
        pass

```

File: password.py

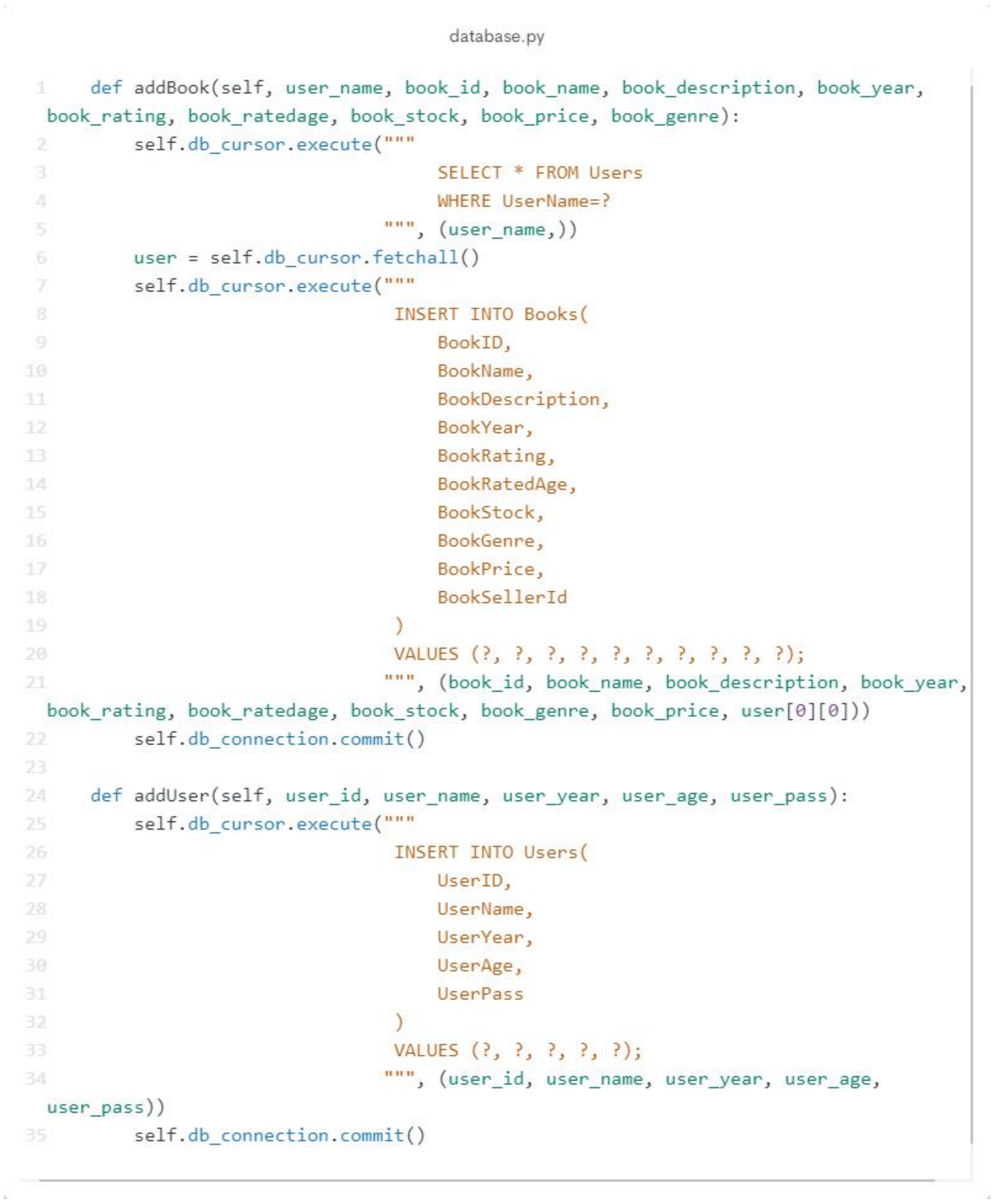

password.py

```python
from zxcvbn import zxcvbn


def checkStrength(userPass):
    """Gives Password strength and suggestion to improve the password

    Args:
        password (str): password entered by the user

    Returns:
        dict: {"strength": int, "suggestion": str}
    """

    result = zxcvbn(password=userPass, user_inputs=[])
    return {"strength": result["score"], "suggestion": result["feedback"]
["suggestions"]}
```

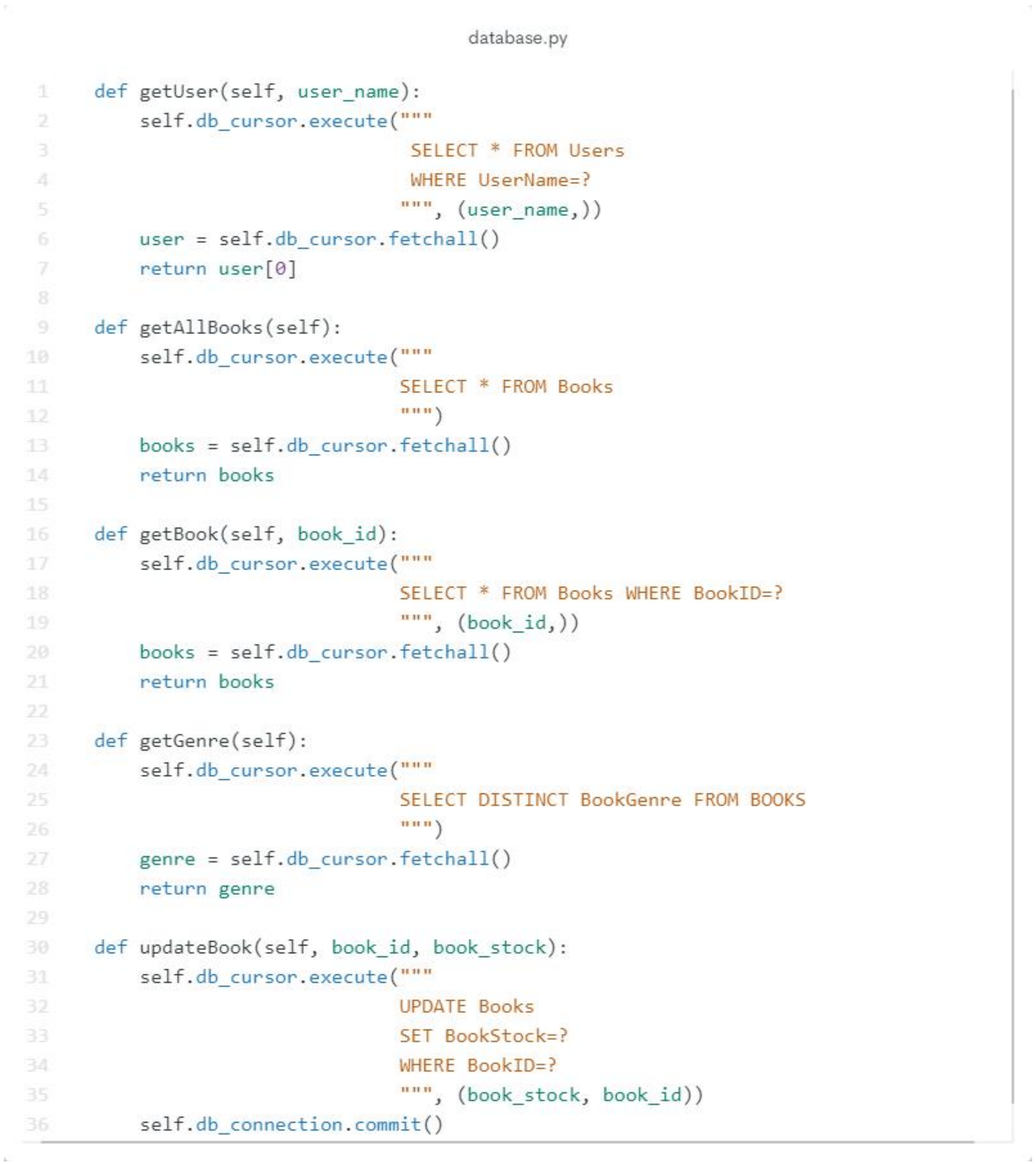
File: database.py


database.py

```python
import os
import sqlite3
class ORM:
    def __init__(self):
        self.db_location = './database.db'
        self.db_connection = sqlite3.connect(
            self.db_location, check_same_thread=False)
        self.db_cursor = self.db_connection.cursor()
        empty = self.db_cursor.execute(
            "SELECT name FROM sqlite_schema").fetchall()
        if empty == []:
            self.__createDefaultTable()
    def __createDefaultTable(self):
        """
            Private function: this function can't be accessed outside of this class

            @summary: This Function creates basic database struct if database is not found
        """
        self.db_cursor.executescript("""
                                    CREATE TABLE Users (
                                    UserID int,
                                    UserName varchar(255),
                                    UserYear int,
                                    UserAge int,
                                    UserPass varchar(255),
                                    PRIMARY KEY (UserID),
                                    UNIQUE(UserName)
                                );
                                    CREATE TABLE Books (
                                    BookID int,
                                    BookName varchar(255),
                                    BookDescription varchar(255),
                                    BookYear int,
                                    BookRating int,
                                    BookRatedAge int,
                                    BookStock int,
                                    BookGenre varchar(255),
                                    BookPrice int,
                                    BookSellerID int,
                                    PRIMARY KEY (BookID),
                                    FOREIGN KEY (BookSellerID) REFERENCES Users(UserID)
                                );
                                """)
        self.db_connection.commit()
```


database.py

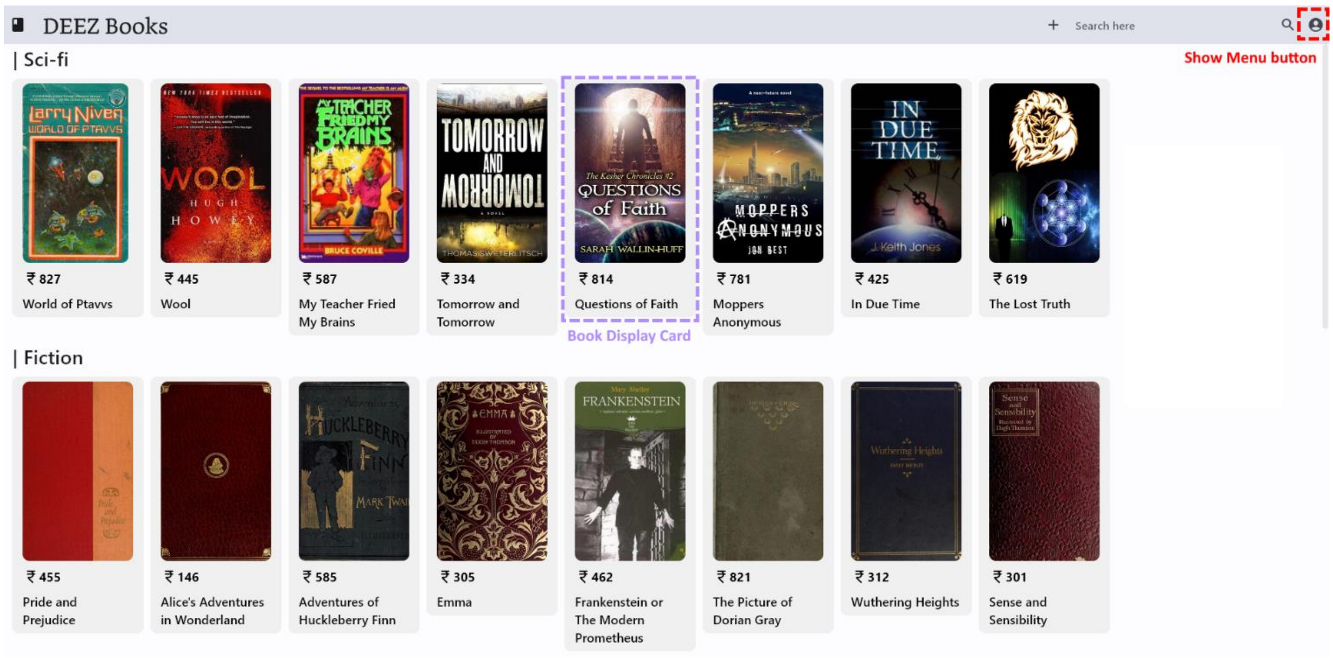
```python
    def addBook(self, user_name, book_id, book_name, book_description, book_year, book_rating, book_ratedage, book_stock, book_price, book_genre):
        self.db_cursor.execute("""
                                    SELECT * FROM Users
                                    WHERE UserName=?
                               """, (user_name,))
        user = self.db_cursor.fetchall()
        self.db_cursor.execute("""
                                INSERT INTO Books(
                                    BookID,
                                    BookName,
                                    BookDescription,
                                    BookYear,
                                    BookRating,
                                    BookRatedAge,
                                    BookStock,
                                    BookGenre,
                                    BookPrice,
                                    BookSellerId
                                )
                                VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?);
                               """, (book_id, book_name, book_description, book_year, book_rating, book_ratedage, book_stock, book_genre, book_price, user[0][0]))
        self.db_connection.commit()

    def addUser(self, user_id, user_name, user_year, user_age, user_pass):
        self.db_cursor.execute("""
                                INSERT INTO Users(
                                    UserID,
                                    UserName,
                                    UserYear,
                                    UserAge,
                                    UserPass
                                )
                                VALUES (?, ?, ?, ?, ?);
                               """, (user_id, user_name, user_year, user_age, user_pass))
        self.db_connection.commit()
```


database.py

```
def getUser(self, user_name):
    self.db_cursor.execute("""
                            SELECT * FROM Users
                            WHERE UserName=?
                           """, (user_name,))
    user = self.db_cursor.fetchall()
    return user[0]

def getAllBooks(self):
    self.db_cursor.execute("""
                           SELECT * FROM Books
                           """)
    books = self.db_cursor.fetchall()
    return books

def getBook(self, book_id):
    self.db_cursor.execute("""
                           SELECT * FROM Books WHERE BookID=?
                           """, (book_id,))
    books = self.db_cursor.fetchall()
    return books

def getGenre(self):
    self.db_cursor.execute("""
                           SELECT DISTINCT BookGenre FROM BOOKS
                           """)
    genre = self.db_cursor.fetchall()
    return genre

def updateBook(self, book_id, book_stock):
    self.db_cursor.execute("""
                           UPDATE Books
                           SET BookStock=?
                           WHERE BookID=?
                           """, (book_stock, book_id))
    self.db_connection.commit()
```

Output

Homepage



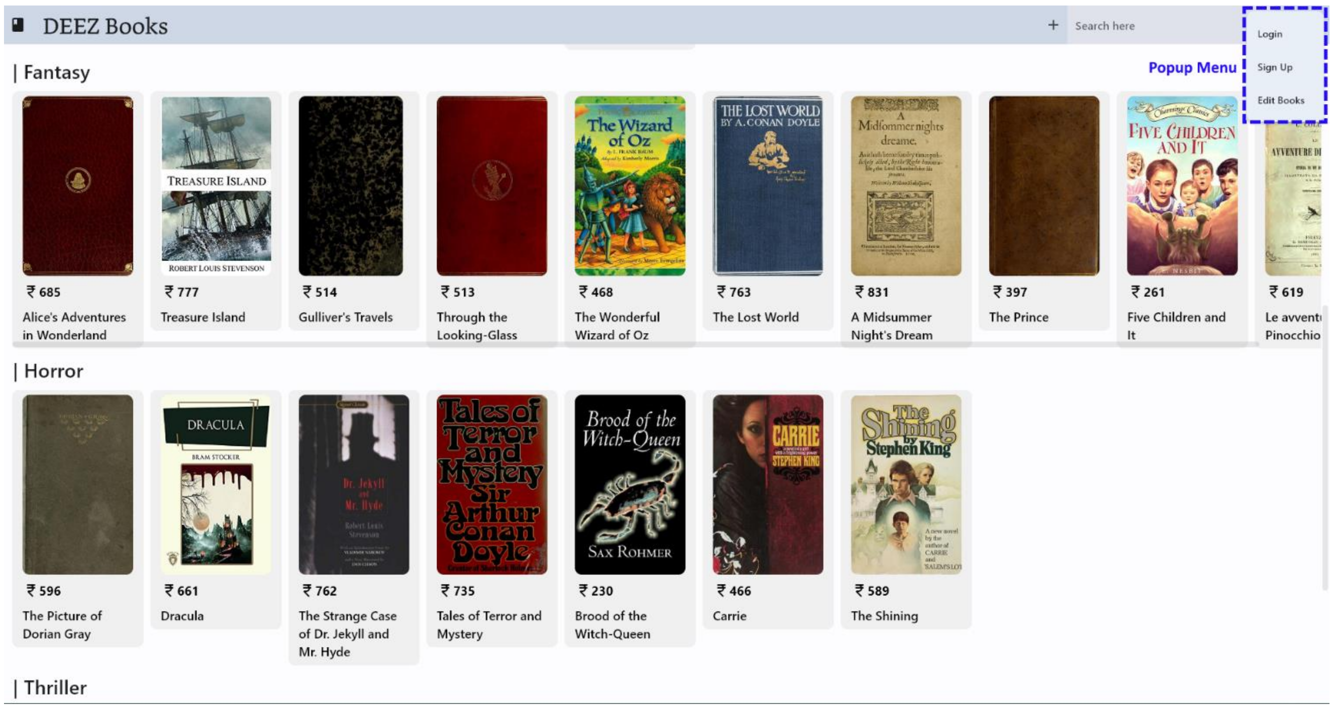
DEEZ Books + Search here

| Sci-fi

₹ 827 World of Ptavvs
₹ 445 Wool
₹ 587 My Teacher Fried My Brains
₹ 334 Tomorrow and Tomorrow
₹ 814 Questions of Faith
₹ 781 Moppers Anonymous
₹ 425 In Due Time
₹ 619 The Lost Truth

| Fiction

₹ 455 Pride and Prejudice
₹ 146 Alice's Adventures in Wonderland
₹ 585 Adventures of Huckleberry Finn
₹ 305 Emma
₹ 462 Frankenstein or The Modern Prometheus
₹ 821 The Picture of Dorian Gray
₹ 312 Wuthering Heights
₹ 301 Sense and Sensibility



DEEZ Books + Search here

| Fantasy

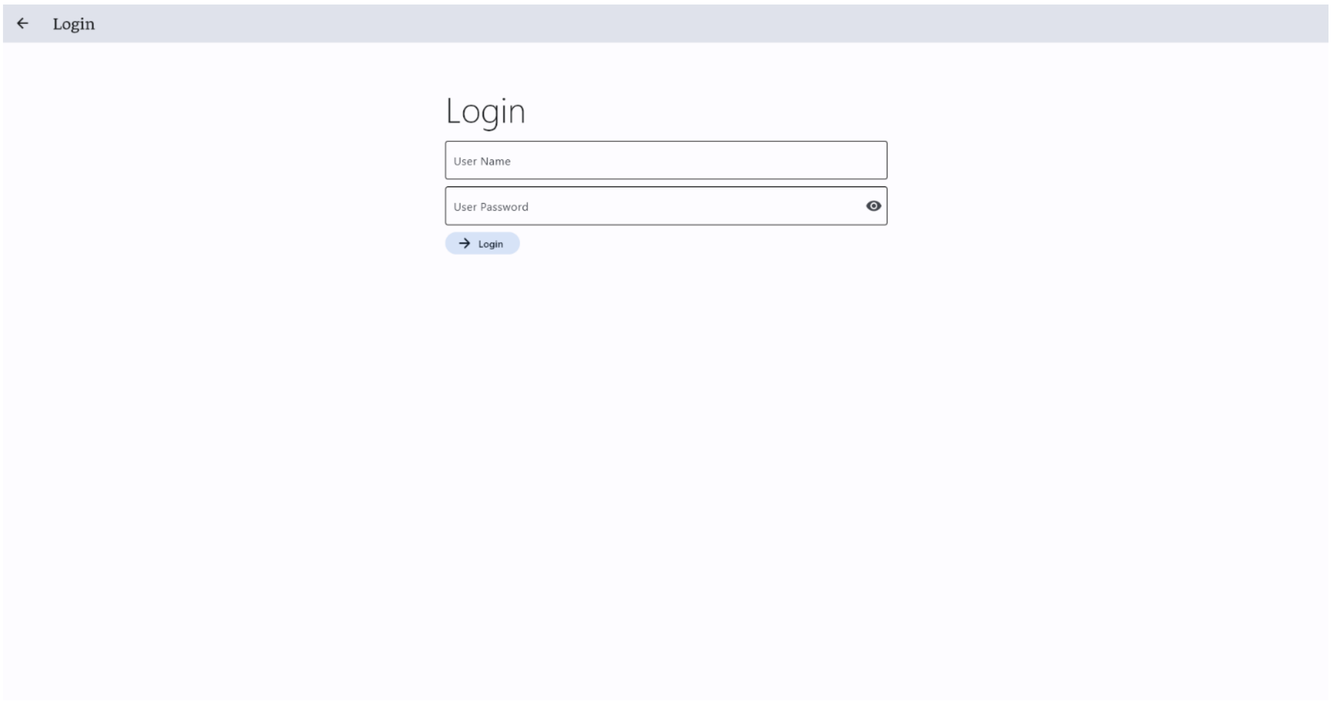
₹ 685 Alice's Adventures in Wonderland
₹ 777 Treasure Island
₹ 514 Gulliver's Travels
₹ 513 Through the Looking-Glass
₹ 468 The Wonderful Wizard of Oz
₹ 763 The Lost World
₹ 831 A Midsummer Night's Dream
₹ 397 The Prince
₹ 261 Five Children and It
₹ 619 Le avventu Pinocchio

| Horror

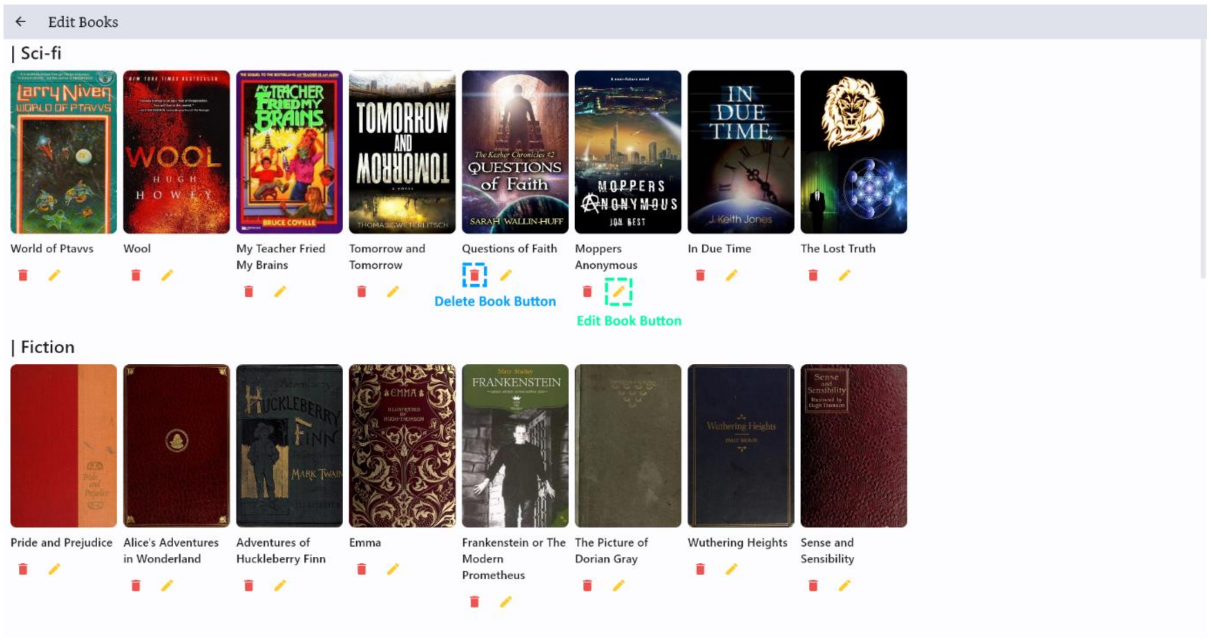
₹ 596 The Picture of Dorian Gray
₹ 661 Dracula
₹ 762 The Strange Case of Dr. Jekyll and Mr. Hyde
₹ 735 Tales of Terror and Mystery
₹ 230 Brood of the Witch-Queen
₹ 466 Carrie
₹ 589 The Shining

| Thriller

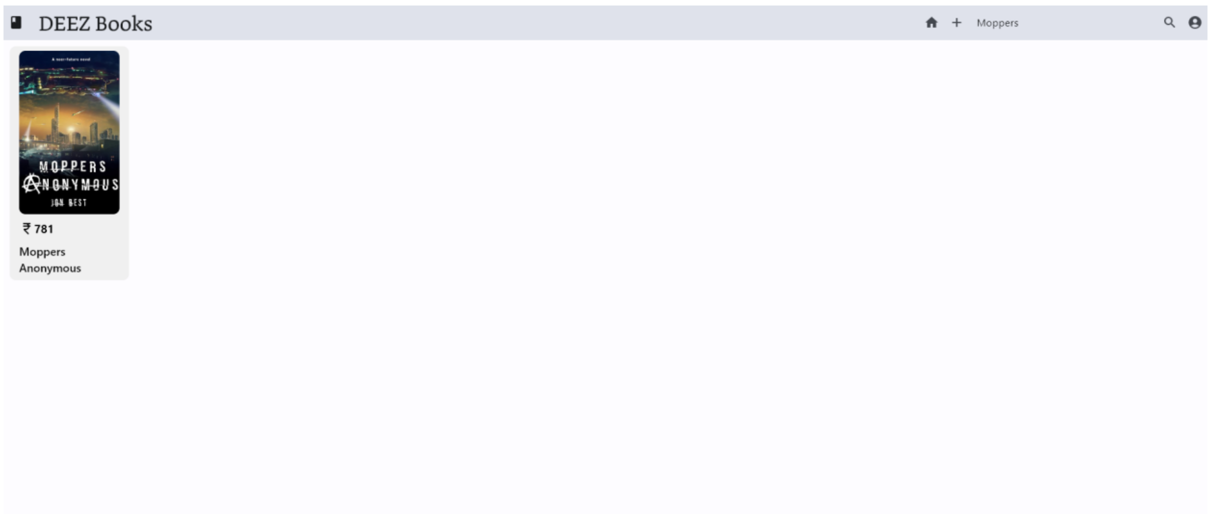
Login Page



Book Edit Page



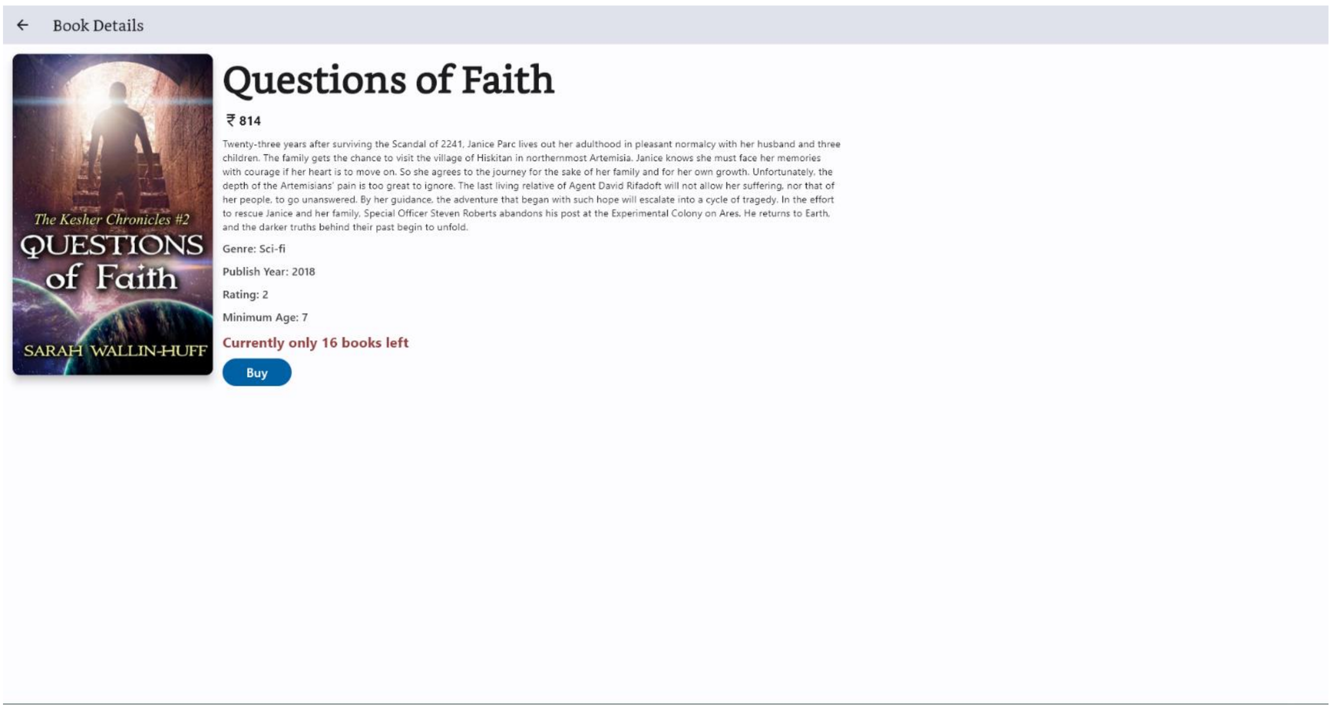
Search Page



₹ 781

Moppers Anonymous

Book View



Questions of Faith

₹ 814

Twenty-three years after surviving the Scandal of 2241, Janice Parc lives out her adulthood in pleasant normalcy with her husband and three children. The family gets the chance to visit the village of Hiskitan in northernmost Artemisia. Janice knows she must face her memories with courage if her heart is to move on. So she agrees to the journey for the sake of her family and for her own growth. Unfortunately, the depth of the Artemisians' pain is too great to ignore. The last living relative of Agent David Rifadoft will not allow her suffering, nor that of her people, to go unanswered. By her guidance, the adventure that began with such hope will escalate into a cycle of tragedy. In the effort to rescue Janice and her family, Special Officer Steven Roberts abandons his post at the Experimental Colony on Ares. He returns to Earth, and the darker truths behind their past begin to unfold.

Genre: Sci-fi

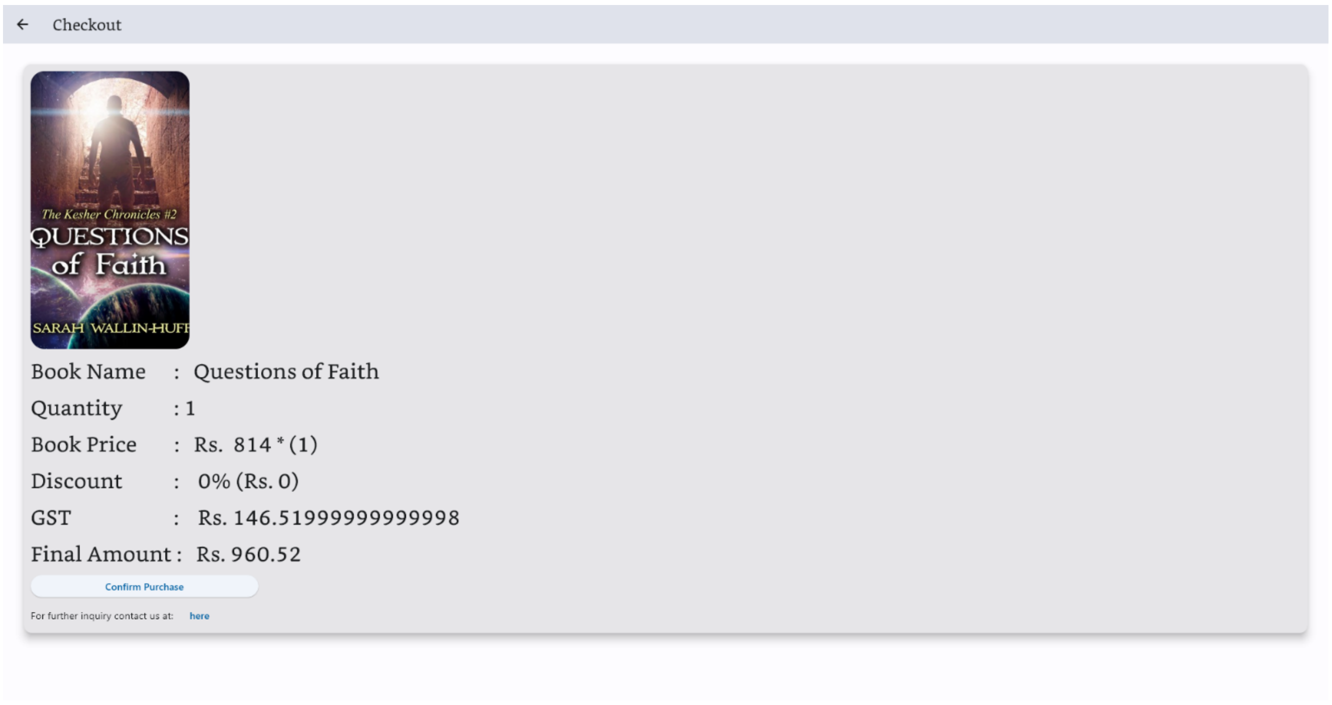
Publish Year: 2018

Rating: 2

Minimum Age: 7

Currently only 16 books left

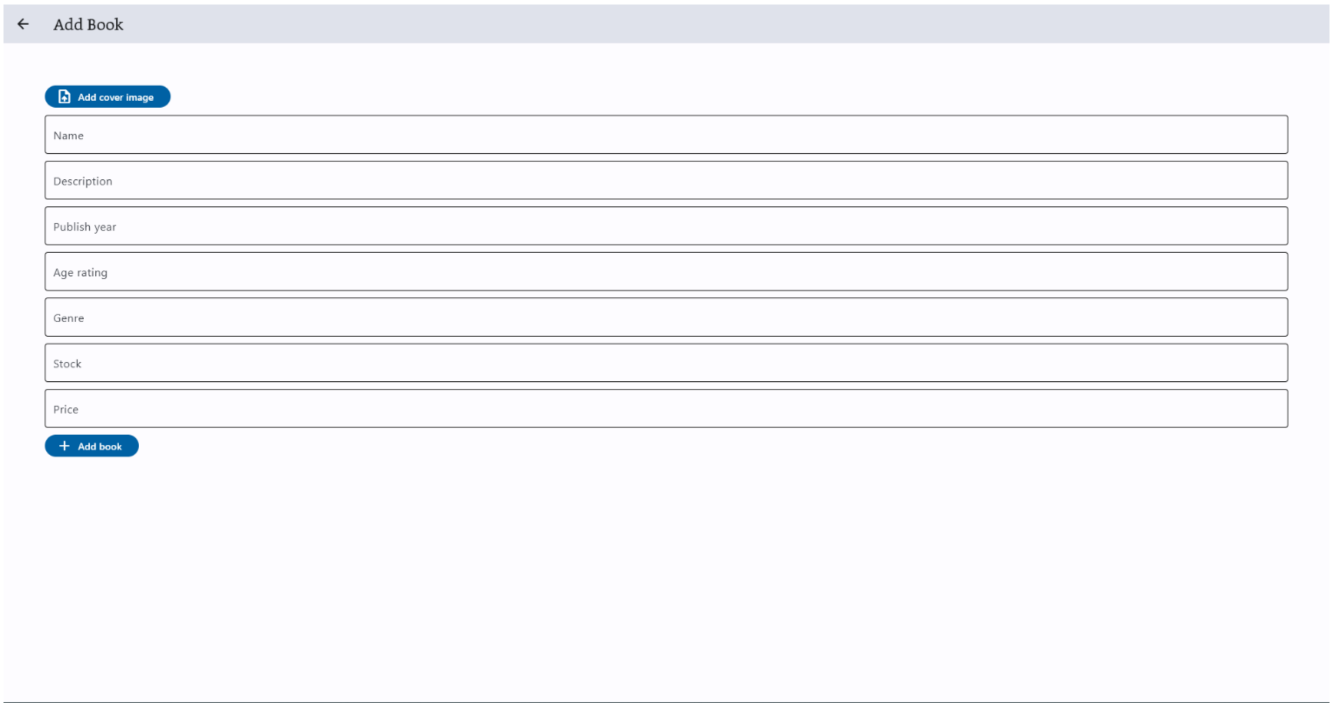
Checkout Page



Book Name	: Questions of Faith
Quantity	: 1
Book Price	: Rs. 814 * (1)
Discount	: 0% (Rs. 0)
GST	: Rs. 146.51999999999998
Final Amount	: Rs. 960.52

For further inquiry contact us at: here

Add Book Page



Conclusion

In this project, the basic principles of python programming, sql databse and database connectivity were implemented, the application allows the user to search for books, create an account and log in, add books to the marketplace, view book information and purchase books. The application is displayed in form a neat user interface made using the Flet UI framework in python.

References

The material/sites/books refered during this project:

- https://flet.dev/docs/
  {for learning the flet framework}

- Class 12 Computer Science Textbook

- Project source code can be found at https://github.com/PAN-Project/DeezBooks