

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

1. Acknowledgement	3
2. Introduction	4
3. System Specifications	5
4. Modules and Functions	6
5. Database Structure	8
6. Python Code	9
7. Output	39
8. Conclusion	43
9. References	44

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.

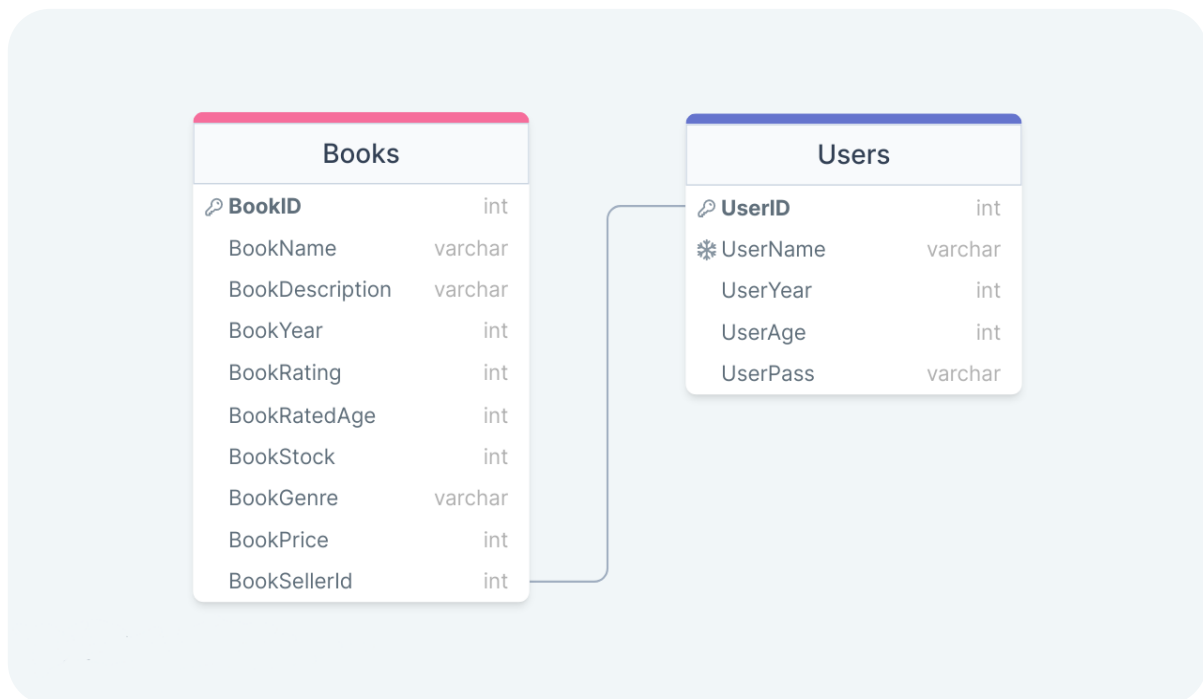
- getBook: This function is used to fetch information about the specified book from the database.
- getGenre: This function is used to fetch all the distinct genres mentioned in the database.
- updateBook: this function is used to update the stock of the mentioned book
- delAllUsers: Function used to 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

```
1 from xmlrpc.client import Boolean
2 import flet as ft
3 import sqlite3 as sql
4
5 from components.views.Homepage import HomePage
6 from components.views.signUpUser import signUpView
7 from components.views.bookAdd import bookAdd
8 from components.views.login import login
9 from components.views.bookBuy import buyBook
10 from components.views.manageBooks import manageBooks
11 from components.views.editBooks import editBooks
12 from components.views.search import search
13 from components.views.Checkout import Checkout
14
15 from database import ORM
16
17
18 def main(page: ft.Page):
19     page.theme_mode = "light"
20     page.title = "DEEZ Books"
21     page.vertical_alignment = ft.MainAxisAlignment.CENTER
22     page.horizontal_alignment = ft.CrossAxisAlignment.CENTER
23     page.scroll = "ALWAYS"
24     page.pick_files_dialog = ft.FilePicker()
25     page.overlay.append(page.pick_files_dialog)
26     page.db = ORM()
27     page.searchQuery = ""
28     page.snack_bar = ft.SnackBar(content=ft.Text(
29         "Book Deleted"), open=False, bgcolor="red")
30     # page.add(page.snack_bar)
31     page.fonts = {
32         "Bookerly Bold": "https://cdn.jsdelivr.net/gh/PAAN-
33         Projects/DeezBooks@master/src/assets/fonts/Bookerly-Bold.ttf",
34         "Bookerly Italic": "https://cdn.jsdelivr.net/gh/PAAN-
35         Projects/DeezBooks@master/src/assets/fonts/Bookerly Italic.ttf",
36         "Bookerly": "https://cdn.jsdelivr.net/gh/PAAN-
37         Projects/DeezBooks@master/src/assets/fonts/Bookerly.ttf"
38     }
```

main.py

```
1  def openSignUp(e):
2      page.go("/signup")
3
4  def openLogin(e):
5      page.go("/login")
6
7  def openEdit(e):
8      page.go("/books/edit")
9
10  homePage = HomePage()
11  signup = signUpView()
12  addbook = bookAdd()
13  loginpage = login()
14  buybook = buyBook()
15  managebooks = manageBooks()
16  editbooks = editBooks()
17  searchRoute = search()
18  checkout = Checkout()
19
20  def setSearchQuery(e):
21      page.searchQuery = e.control.value
22      page.update()
23
24  def goToSearch(e):
25      if (len(page.searchQuery) > 0):
26          page.go("/home")
27          page.go("/search")
28          # page.searchQuery = searchBar.value
29          # print(page.searchQuery)
```

```

1  def routeChange(route):
2      page.views.clear()
3      page.views.append(
4          ft.View(
5              "/",
6              [
7                  ft.AppBar(
8                      leading=ft.Icon(ft.icons.BOOK_ROUNDED),
9                      leading_width=40,
10                     title=ft.Text(
11                         "DEEZ Books", font_family="Bookerly", size=32),
12                     center_title=False,
13                     bgcolor=ft.colors.SURFACE_VARIANT,
14                     actions=[
15                         ft.IconButton(
16                             ft.icons.ADD, on_click=lambda _: page.go("/book/add")),
17                         ft.TextField(
18                             filled=True,
19                             hint_text="Search here",
20                             border_width=0,
21                             border_radius=0,
22                             on_change=setSearchQuery,
23                         ),
24                         ft.IconButton(ft.icons.SEARCH_ROUNDED,
25                                     on_click=goToSearch),
26                         ft.PopupMenuButton(
27                             items=[
28                                 ft.PopupMenuItem(
29                                     text="Login", on_click=openLogin),
30                                 ft.PopupMenuItem(
31                                     text="Sign Up", on_click=openSignUp),
32                                 ft.PopupMenuItem(
33                                     text="Edit Books", on_click=openEdit)
34                             ],
35                             icon=ft.icons.ACCOUNT_CIRCLE_ROUNDED
36                         )
37                     ],
38                 ),
39                 homePage
40             ],
41             scroll="ALWAYS",
42             padding=0
43         )

```

```

1  if page.route == "/signup":
2      page.views.append(ft.View(
3          "/signup",
4          [
5              ft.AppBar(
6                  title=ft.Text("Sign Up",
7                              font_family="Bookerly"),
8                  bgcolor=ft.colors.SURFACE_VARIANT,
9              ),
10             signup],
11             scroll="ALWAYS"
12         ))
13
14 elif page.route == "/login":
15     page.views.append(ft.View(
16         "/login",
17         [
18             ft.AppBar(
19                 title=ft.Text("Login",
20                             font_family="Bookerly"),
21                 bgcolor=ft.colors.SURFACE_VARIANT,
22             ),
23             loginpage],
24             scroll="ALWAYS"
25         ))
26
27 elif page.route == "/book/add":
28     page.views.append(ft.View(
29         "/book/add",
30         [
31             ft.AppBar(
32                 title=ft.Text("Add Book",
33                             font_family="Bookerly"),
34                 bgcolor=ft.colors.SURFACE_VARIANT,
35             ),
36             addbook],
37             scroll="ALWAYS"
38         ))
39 elif "/book/buy/" in page.route:
40     print(page.route)
41     page.views.append(ft.View(
42         page.route,
43         [
44             ft.AppBar(
45                 title=ft.Text("Book Details",
46                             font_family="Bookerly"),
47                 bgcolor=ft.colors.SURFACE_VARIANT,
48             ),
49             buybook],
50             scroll=ft.ScrollMode.ALWAYS
51         ))

```

```
1 elif "/checkout" in page.route:
2     page.views.append(ft.View(
3         page.route,
4         [
5             ft.AppBar(
6                 title=ft.Text("Checkout",
7                               font_family="Bookerly"),
8                 bgcolor=ft.colors.SURFACE_VARIANT,
9             ),
10            checkout],
11         scroll=ft.ScrollMode.ALWAYS
12     ))
13 elif "/book/edit" in page.route:
14     page.views.append(ft.View(
15         page.route,
16         [
17             ft.AppBar(
18                 title=ft.Text("Edit Book Stock",
19                               font_family="Bookerly"),
20                 bgcolor=ft.colors.SURFACE_VARIANT,
21             ),
22            editbooks],
23         scroll="ALWAYS"
24     ))
25 elif page.route == "/books/edit":
26     page.views.append(ft.View(
27         "/books/edit",
28         [
29             ft.AppBar(
30                 title=ft.Text("Edit Books",
31                               font_family="Bookerly"),
32                 bgcolor=ft.colors.SURFACE_VARIANT,
33             ),
34            managebooks],
35         padding=0,
36         scroll="ALWAYS"
37     ))
```

```

1 elif page.route == "/search":
2     page.views.append(ft.View(
3         "/search",
4         [
5             ft.AppBar(
6                 leading=ft.Icon(ft.icons.BOOK_ROUNDED),
7                 leading_width=40,
8                 title=ft.Text(
9                     "DEEZ Books", font_family="Bookerly", size=32),
10                center_title=False,
11                bgcolor=ft.colors.SURFACE_VARIANT,
12                actions=[
13                    ft.IconButton(
14                        ft.icons.HOME, on_click=lambda _: page.go("/home")),
15                    ft.IconButton(
16                        ft.icons.ADD, on_click=lambda _: page.go("/book/add")),
17                    ft.TextField(
18                        filled=True,
19                        hint_text="Search here",
20                        border_width=0,
21                        border_radius=0,
22                        on_change=setSearchQuery,
23                        value=page.searchQuery
24                    ),
25                    ft.IconButton(ft.icons.SEARCH_ROUNDED,
26                                on_click=goToSearch),
27                    ft.PopupMenuButton(
28                        items=[
29                            ft.PopupMenuItem(
30                                text="Login", on_click=openLogin),
31                            ft.PopupMenuItem(
32                                text="Sign Up", on_click=openSignUp),
33                            ft.PopupMenuItem(
34                                text="Edit Books", on_click=openEdit)
35                        ],
36                        icon=ft.icons.ACCOUNT_CIRCLE_ROUNDED
37                    )
38                ],
39            ),
40            searchRoute],
41            padding=0,
42            scroll="ALWAYS"
43        ))
44 page.padding = 0
45 page.update()

```


main.py

```
1  def view_pop(e):
2      print("View pop:", e.view)
3      page.views.pop()
4      top_view = page.views[-1]
5      page.go(top_view.route)
6
7  page.on_route_change = routeChange
8  page.on_view_pop = view_pop
9  page.padding = 10
10 page.go(page.route)
11
12
13 ft.app(target=main, assets_dir="assets",
14         upload_dir="assets/uploads", view=ft.WEB_BROWSER)
```


File: bookAdd.py

bookAdd.py

```
1 import flet as ft
2
3 from random import randint
4
5
6 class bookAdd(ft.UserControl):
7     def build(self):
8         self.selected_files = ft.Text(weight=ft.FontWeight.W_500, size=24)
9         self.book_cover = ft.Image(
10             visible=False, height=340, border_radius=ft.border_radius.all(10),
11             fit=ft.ImageFit.CONTAIN
12         )
13         self.file_picker = ft.Row(controls=[
14             ft.FilledButton(
15                 "Add cover image",
16                 icon=ft.icons.UPLOAD_FILE,
17                 on_click=self.handlePickFiles,
18             ),
19             self.selected_files,
20         ])
21
22         self.book_name = ft.TextField(label="Name")
23         self.book_desc = ft.TextField(label="Description", multiline=True)
24         self.book_year = ft.TextField(label="Publish year")
25         self.book_age_rating = ft.TextField(label="Age rating")
26         self.book_genre = ft.TextField(label="Genre")
27         self.book_stock = ft.TextField(label="Stock")
28         self.book_price = ft.TextField(label="Price")
29
30         self.add_book_button = ft.FilledButton(
31             "Add book", icon=ft.icons.ADD_ROUNDED, on_click=self.addBookDb)
```

```
1     self.warningSnackBar = ft.SnackBar(  
2         ft.Text("Something went wrong"))  
3     return ft.Container(content=ft.Column(controls=[  
4         self.book_cover,  
5         self.file_picker,  
6         self.book_name,  
7         self.book_desc,  
8         self.book_year,  
9         self.book_age_rating,  
10        self.book_genre,  
11        self.book_stock,  
12        self.book_price,  
13        self.add_book_button,  
14        self.warningSnackBar  
15    ], alignment=ft.MainAxisAlignment.CENTER), alignment=ft.alignment.center,  
padding=50)  
16  
17     def handlePickFiles(self, e):  
18         self.page.pick_files_dialog.on_result = self.pick_files_result  
19         self.page.pick_files_dialog.pick_files(  
20             allow_multiple=False,  
21             allowed_extensions=['png']  
22         ),  
23  
24     def pick_files_result(self, e: ft.FilePickerResultEvent):  
25         self.selected_files.value = f"{e.files[0].name} - {e.files[0].size / 1000}kb"  
26         self.book_cover.src = e.files[0].path  
27         self.book_cover.visible = True  
28  
29         self.book_cover.update()  
30         self.selected_files.update()
```

```

1  def upload_files(self):
2      upload_list = []
3      if self.page.pick_files_dialog.result != None and
self.page.pick_files_dialog.result.files != None:
4          for f in self.page.pick_files_dialog.result.files:
5              upload_list.append(
6                  ft.FilePickerUploadFile(
7                      f.name,
8                      upload_url=self.page.get_upload_url(
9                          f"{self.book_name.value}.png", 600),
10                     method="PUT"
11                 )
12             )
13         self.page.pick_files_dialog.upload(upload_list)
14
15     def addBookDb(self, e):
16         self.warningSnackBar.content = ft.Text("One or more fields empty")
17         if self.page.session.get("current_user") == None:
18             self.warningSnackBar.content = ft.Text("Please login.")
19             self.warningSnackBar.open = True
20             self.warningSnackBar.bgcolor = "red"
21             self.update()
22         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
== "":
23             self.warningSnackBar.open = True
24             self.warningSnackBar.bgcolor = "red"
25             self.update()
26             return
27         else:
28             try:
29                 book_id = randint(1_000_000, 9_999_999)
30                 self.page.db.addBook(self.page.session.get("current_user"), book_id,
self.book_name.value, self.book_desc.value, int(
31                     self.book_year.value), 0, int(self.book_age_rating.value),
int(self.book_stock.value), self.book_genre.value, self.book_price.value)
32                 self.upload_files()
33
34                 self.warningSnackBar.content = ft.Text(
35                     "Book added successfully!")
36                 self.warningSnackBar.open = True
37                 self.warningSnackBar.bgcolor = "green"
38                 self.update()
39             except:
40                 print("ERR")
41                 self.warningSnackBar.open = True
42                 self.update()

```

File: bookBuy.py

bookBuy.py

```
1 import flet as ft
2
3 from database import ORM
4
5
6 class buyBook(ft.UserControl):
7     def did_mount(self):
8         self.book_id = self.page.route.removeprefix("/book/buy/")
9         self.book = self.db.getBook(self.book_id)
10        self.content.append(
11            ft.Column(
12                controls=[
13                    ft.Row(
14                        controls=[
15                            ft.Card(content=ft.Image(
16                                src=f"src\\assets\\uploads\\{self.book[0][1]}.png",
17                                height=464, width=290, border_radius=ft.BorderRadius(top_left=10, top_right=10,
18                                bottom_right=10, bottom_left=10), fit=ft.ImageFit.COVER),
19                                elevation=10, shadow_color="black"
20                            ),
21                            ft.Column(
22                                controls=[
23                                    ft.Text(
24                                        value=self.book[0][1], size=54,
25                                        weight=ft.FontWeight.W_600, width=1000, font_family="Bookerly"),
26                                    ft.Row(controls=[
27                                        ft.Icon(name=ft.icons.CURRENCY_RUPEE_ROUNDED), ft.Text(
28                                            value=self.book[0][8], size=18,
29                                            weight=ft.FontWeight.W_700),], spacing=0, alignment=ft.MainAxisAlignment.CENTER),
30                                        ft.Container(content=ft.Markdown(
31                                            value=self.book[0][2], selectable=True,
32                                            extension_set="gitHubWeb",
33                                        ), width=900),
```

```

bookBuy.py

1         ft.Text(
2             value=f"Genre: {self.book[0][7]}", size=16,
width=800, color="#414141", weight=ft.FontWeight.W_600),
3         ft.Text(
4             value=f"Publish Year: {self.book[0][3]}",
size=16, width=800, color="#414141", weight=ft.FontWeight.W_600),
5         ft.Text(
6             value=f"Rating: {self.book[0][4]}", size=16,
width=800, color="#414141", weight=ft.FontWeight.W_600),
7         ft.Text(
8             value=f"Minimum Age: {self.book[0][5]}",
size=16, width=800, color="#414141", weight=ft.FontWeight.W_600),
9         ft.Text(
10            value=f"Currently only {self.book[0][6]}
books left", size=20, width=800, color="#914141", weight=ft.FontWeight.W_700),
11        ft.FilledButton(
12            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)
13        ],
14        # scroll=ft.ScrollMode.ALWAYS, height=620
15        )
16    ], alignment=ft.MainAxisAlignment.START,
vertical_alignment=ft.CrossAxisAlignment.START
17    ),
18    ]
19    )
20    )

```

```

bookBuy.py

1    self.update()
2    return super().did_mount()
3
4    def build(self):
5        self.db = ORM()
6        self.books = self.db.getAllBooks()
7
8        self.content = []
9        return self.content
10
11    def goToBill(self, e, book_id):
12        self.page.go(f"/checkout/{book_id}")
13

```


File: Checkout.py

Checkout.py

```
1 import flet as ft
2
3 from database import ORM
4
5
6 class Checkout(ft.UserControl):
7     def did_mount(self):
8         self.book_id = self.page.route.removeprefix("/checkout/")
9         self.bookdetails = self.db.getBook(self.book_id)
10        self.content.append(
11            ft.Card(
12                content=ft.Container(
13                    content=ft.Column(
14                        controls=[
15                            ft.Image(src=f"src\\assets\\uploads\\{self.bookdetails[0]
16                                [1]}.png", border_radius=ft.BorderRadius(
17                                    top_left=20, top_right=20, bottom_right=20,
18                                    bottom_left=20), fit=ft.ImageFit.COVER, height=403, width=230),
19                            ft.Text(
20                                value=f"Book Name      : \t\t\t{self.bookdetails[0]
21                                    [1]}", font_family="Bookerly", size=30),
22                            ft.Text(
23                                value=f"Quantity          : 1",
24                                font_family="Bookerly", size=30),
25                            ft.Text(
26                                value=f"Book Price      : \t\t\tRs.
27                                    {self.bookdetails[0][8]} * (1)", font_family="Bookerly", size=30),
28                            ft.Text(
29                                value=f"Discount        : \t\t\t0% (Rs. 0)",
30                                font_family="Bookerly", size=30),
31                            ft.Text(
32                                value=f"GST              : \t\t\tRs.
33                                    {(self.bookdetails[0][8])*(0.18)}", font_family="Bookerly", size=30),
34                            ft.Text(
35                                value=f"Final Amount : \t\t\tRs. {(self.bookdetails[0]
36                                    [8])*(0.18)+(self.bookdetails[0][8])}", font_family="Bookerly", size=30),
37                            ft.ElevatedButton(
38                                text="Confirm Purchase", width=330,
39                                on_click=self.confirm),
40                            ft.Row(
41                                controls=[ft.Text("For further inquiry contact us
42                                    at:"), ft.TextButton(text="here", url="https://www.youtube.com/video/dQw4w9WgXcQ?
43                                    autoplay=1&disablekb=1")]]
44                        )
45                    )
46                )
47            )
48        ),
```

Checkout.py

```
1         padding=10
2     ),
3     elevation=10,
4     shadow_color="black",
5     surface_tint_color="#4f4f4f",
6     margin=20,
7 )
8 ),
9
10 print(self.bookdetails)
11
12 self.update()
13 return super().did_mount()
14
15 def build(self):
16     self.db = ORM()
17     self.books = self.db.getAllBooks()
18     self.dialog = ft.AlertDialog(
19         title=ft.Text("Thank you!!"), on_dismiss=lambda e: self.page.go("/"),
20         content=ft.Image(src="\\assets\\images\\rickroll-roll.gif"),
21     )
22
23     self.content = [self.dialog]
24     return self.content
25
26 def confirm(self, e):
27     self.dialog.open = True
28     if (self.bookdetails[0][6] == 1):
29         self.db.delBook(self.book_id)
30     else:
31         stock = self.bookdetails[0][6] - 1
32         self.db.updateBook(self.book_id, stock)
33     self.update()
```

File: editBooks.py

editBook.py

```
1 import flet as ft
2
3 from database import ORM
4
5
6 class editBooks(ft.UserControl):
7     def did_mount(self):
8         self.book_id = self.page.route.removeprefix("/book/edit/")
9         self.book = self.db.getBook(self.book_id)
10
11         self.book_stock = ft.TextField(label="Stock", value=self.book[0][6])
12         self.updateButton = ft.FilledButton(
13             text="Update", on_click=self.updateStock)
14         self.warn = ft.SnackBar(content=ft.Text(
15             "Updated Stock!"), bgcolor="green")
16         self.layout = ft.Column(
17             controls=[self.book_stock, self.updateButton, self.warn])
18
19         self.content.append(self.layout)
20         self.update()
21         return super().did_mount()
22
23     def build(self):
24         self.db = ORM()
25         self.books = self.db.getAllBooks()
26
27         self.content = []
28         return self.content
29
30     def updateStock(self, e):
31         self.page.db.updateBook(self.book_id, self.book_stock.value)
32         self.warn.open = True
33         self.update()
```


File: Homepage.py

Homepage.py

```
1 import flet as ft
2
3 from database import ORM
4
5
6 class HomePage(ft.UserControl):
7
8     def build(self):
9         self.db = ORM()
10        self.books = self.db.getAllBooks()
11        self.genre = self.db.getGenre()
12        self.BookRow = ft.Row(scroll=ft.ScrollMode.ALWAYS,
13                                alignment=ft.MainAxisAlignment.START,
14                                vertical_alignment=ft.CrossAxisAlignment.START)
15
16        self.BookColumn = []
17
18        for j in self.genre:
19            self.BookColumn.append(ft.Text(
20                value=f"| {j[0]}", weight=ft.FontWeight.W_600,
21                text_align=ft.TextAlign.END, size=28))
22
23            for i in self.books:
24                if i[7] == j[0]:
25                    self.BookRow.controls.append(
26                        ft.TextButton(
27                            content=ft.Column(controls=[
28                                ft.Image(src=f"\\assets\\uploads\\{i[1]}.png",
29                                    height=260,
30                                    fit=ft.ImageFit.COVER,
31                                    border_radius=ft.BorderRadius(10, 10, 10, 10)),
32                                ft.Row(controls=[
33                                    ft.Icon(name=ft.icons.CURRENCY_RUPEE_ROUNDED),
34                                    ft.Text(
35                                        value=i[8], size=18,
36                                        weight=ft.FontWeight.W_700),], spacing=0, alignment=ft.MainAxisAlignment.START),
37                                    ft.Text(value=i[1], size=18,
38                                        weight=ft.FontWeight.W_500,
39                                        text_align=ft.TextAlign.START),
40                                ], alignment=ft.MainAxisAlignment.CENTER),
41                            style=ft.ButtonStyle(bgcolor="#f0f0f0", color="black", shape={
42                                ft.MaterialState.DEFAULT:
43                                    ft.RoundedRectangleBorder(radius=10),
44                                }, padding=15), on_click=lambda e, book_id=i[0]:
45                                self.goToBook(e, book_id), width=190)
46                    )
```

```
1         self.BookColumn.append(self.BookRow)
2         self.BookRow = ft.Row(scroll=ft.ScrollMode.ALWAYS,
3         alignment=ft.MainAxisAlignment.START,
4         vertical_alignment=ft.CrossAxisAlignment.START)
5         self.BookShelf = ft.Column(controls=self.BookColumn,
6         alignment=ft.MainAxisAlignment.START,
7         horizontal_alignment=ft.CrossAxisAlignment.START)
8         self.BookContainer = ft.Container(
9         content=self.BookShelf, padding=0, margin=ft.Margin(left=12, top=0,
10         right=12, bottom=0), alignment=ft.alignment.center_left)
11         return self.BookContainer
12
13     def goToBook(self, e, book_id):
14         self.page.go(f"/book/buy/{book_id}")
```

File: login.py

login.py

```
1 import flet as ft
2
3 from random import randint
4 import datetime
5
6
7 class login(ft.UserControl):
8     def build(self):
9         self.loggedIn = False
10        self.header = ft.Text(
11            "Login", weight=ft.FontWeight.W_200, size=50)
12        self.userNameInput = ft.TextField(label="User Name")
13        self.userPassInput = ft.TextField(
14            label="User Password", password=True, can_reveal_password=True)
15        self.warningSnackBar = ft.SnackBar(
16            ft.Text("Wrong username or password"))
17        self.loginBtn = ft.FilledTonalButton(
18            "Login", icon=ft.icons.ARROW_FORWARD_ROUNDED, on_click=self.login)
19
20        return ft.Container(content=ft.Column(controls=[
21            self.header,
22            self.userNameInput,
23            self.userPassInput,
24            self.loginBtn,
25            self.warningSnackBar
26
27        ], width=640), alignment=ft.alignment.center, padding=50)
```

```
1  def login(self, e):
2      self.warningSnackBar.open = False
3      self.warningSnackBar.content = ft.Text("Wrong username or password")
4      self.warningSnackBar.bgcolor = "red"
5      self.update()
6      try:
7          req = self.page.db.getUser(user_name=self.userNameInput.value, )
8          if req[4] == self.userPassInput.value:
9              self.page.session.set("current_user", req[1])
10             self.warningSnackBar.content = ft.Text(
11                 f"Logged in as {req[1]}")
12             self.warningSnackBar.open = True
13             self.warningSnackBar.bgcolor = "green"
14         else:
15             self.warningSnackBar.open = True
16     except:
17         self.warningSnackBar.open = True
18         pass
19     self.update()
```

File: manageBooks.py

manageBooks.py

```
1 import flet as ft
2
3 from database import ORM
4
5
6 class manageBooks(ft.UserControl):
7     def build(self):
8         self.db = ORM()
9         self.books = self.db.getAllBooks()
10        self.genre = self.db.getGenre()
11        self.BookRow = ft.Row(scroll=ft.ScrollMode.ALWAYS,
12                               alignment=ft.MainAxisAlignment.START,
13                               vertical_alignment=ft.CrossAxisAlignment.START)
14        self.BookColumn = []
15
16        for j in self.genre:
17            self.BookColumn.append(ft.Text(
18                value=f"| {j[0]}", weight=ft.FontWeight.W_600,
19                text_align=ft.TextAlign.END, size=28))
20
21            for i in self.books:
22                if i[7] == j[0]:
23                    self.BookRow.controls.append(
24                        ft.Column(
25                            controls=[
26                                ft.Image(src=f"\\assets\\uploads\\{i[1]}.png",
27                                         height=260, width=170,
28                                         fit=ft.ImageFit.COVER,
29                                         border_radius=ft.BorderRadius(10, 10, 10, 10), ),
30                                ft.Text(value=self.books[self.books.index(
31                                    i)][1], size=18, weight=ft.FontWeight.W_500,
32                                    text_align=ft.TextAlign.START, width=170),
33                                ft.Row(controls=[
34                                    ft.IconButton(icon=ft.icons.DELETE,
35                                                    style=ft.ButtonStyle(color=ft.colors.RED_400), on_click=lambda e, book_id=i[0]:
36                                                    self.deleteBook(e, book_id)), ft.IconButton(icon=ft.icons.EDIT,
37                                                    style=ft.ButtonStyle(color=ft.colors.AMBER_400), on_click=lambda e, book_id=i[0]:
38                                                    self.goToBook(e, book_id))]),
39                                    ft.Text(value=" ", size=18,
40                                            weight=ft.FontWeight.W_500,
41                                            text_align=ft.TextAlign.START, width=170,
42                                            max_lines=1)
43                                ]
44                            )
45                    )
46            )
47        )
48    )
```

manageBooks.py

```
1         self.BookColumn.append(self.BookRow)
2         self.BookRow = ft.Row(scroll=ft.ScrollMode.ALWAYS,
3         alignment=ft.MainAxisAlignment.START,
4         vertical_alignment=ft.CrossAxisAlignment.START)
5         self.BookShelf = ft.Column(controls=self.BookColumn,
6         alignment=ft.MainAxisAlignment.START,
7         horizontal_alignment=ft.CrossAxisAlignment.START)
8         self.BookContainer = ft.Container(
9         content=self.BookShelf, padding=0, margin=ft.Margin(left=12, top=0,
10         right=12, bottom=0), alignment=ft.alignment.center_left)
11         return self.BookContainer
12
13 def goToBook(self, e, book_id):
14     self.page.go(f"/book/edit/{book_id}")
15
16 def deleteBook(self, e, book_id):
17     self.page.db.delBook(book_id)
18     self.page.snack_bar.open = True
19
20     # Refreshes view to get new books
21     self.page.go("/book/")
22     self.page.go("/books/edit")
```


File: search.py

search.py

```
1 import flet as ft
2
3 from database import ORM
4
5
6 class search(ft.UserControl):
7     def did_mount(self):
8         self.book_id = self.page.route.removeprefix("/book/edit/")
9         self.books = self.db.searchBook(self.page.searchQuery)
10
11         self.layout = ft.Row(
12             controls=[],
13             scroll=ft.ScrollMode.ALWAYS, alignment=ft.MainAxisAlignment.START,
14             vertical_alignment=ft.CrossAxisAlignment.START)
15
16         for i in self.books:
17             self.layout.controls.append(
18                 ft.TextButton(
19                     content=ft.Column(controls=[
20                         ft.Image(src=f"\\assets\\uploads\\{i[1]}.png", height=260,
21                             fit=ft.ImageFit.COVER,
22                             border_radius=ft.BorderRadius(10, 10, 10, 10)),
23                         ft.Row(controls=[
24                             ft.Icon(name=ft.icons.CURRENCY_RUPEE_ROUNDED), ft.Text(
25                                 value=i[8], size=18, weight=ft.FontWeight.W_700),],
26                             spacing=0, alignment=ft.MainAxisAlignment.START),
27                             ft.Text(value=i[1], size=18, weight=ft.FontWeight.W_500,
28                                 text_align=ft.TextAlign.START),
29                             ], alignment=ft.MainAxisAlignment.CENTER),
30                             style=ft.ButtonStyle(bgcolor="#f0f0f0", color="black", shape={
31                                 ft.MaterialState.DEFAULT:
32                                 ft.RoundedRectangleBorder(radius=10),
33                             }, padding=15), on_click=lambda e, book_id=i[0]: self.goToBook(e,
34                                 book_id), width=190)
35                     )
```

search.py

```
1     if len(self.books) == 0:
2         self.layout.controls.append(ft.Text(value="No result"))
3
4     self.content.content = self.layout
5     self.update()
6     return super().did_mount()
7
8     def build(self):
9         self.db = ORM()
10
11         self.content = ft.Container(padding=ft.Padding(
12             left=10, top=10, right=10, bottom=10))
13         return self.content
14
15     def goToBook(self, e, book_id):
16         self.page.go(f"/book/buy/{book_id}")
17
```


File: signUpUser.py

signUpUser.py

```
1import flet as ft
2from utils.password import checkStrength
3
4from random import randint
5import datetime
6
7
8class signUpView(ft.UserControl):
9    def build(self):
10        self.userNameInput = ft.TextField(label="Name")
11        self.userAgeInput = ft.TextField(
12            label="Age", keyboard_type=ft.KeyboardType.NUMBER, error_text="",
13            on_change=self.check_int_age)
14        self.userPasswordInput = ft.TextField(
15            label="Password", password=True, can_reveal_password=True,
16            on_change=self.check_pass_strength)
17        self.passwordStrengthBar = ft.ProgressBar(value=0)
18        self.passwordStrengthText = ft.Text("")
19        self.submitButton = ft.FilledTonalButton(
20            text="Sign Up", icon=ft.icons.ARROW_FORWARD_ROUNDED,
21            on_click=self.add_user)
22        self.warningText = "Please enter a valid password"
23        self.warningSnackBar = ft.SnackBar(ft.Text(self.warningText))
24        return ft.Container(content=ft.Column(controls=[
25            self.userNameInput,
26            self.userAgeInput,
27            self.userPasswordInput,
28            self.passwordStrengthBar,
29            self.passwordStrengthText,
30            self.submitButton,
31            self.warningSnackBar
32        ], width=640), alignment=ft.alignment.center, padding=50)
```

```

1  def check_int_age(self, e):
2      try:
3          self.userAgeInput.error_text = ""
4          temp_int = int(e.control.value)
5      except:
6          self.userAgeInput.error_text = "Please enter a valid age"
7          self.update()
8
9  def check_int_year(self, e):
10     try:
11         self.userYearInput.error_text = ""
12         temp_int = int(e.control.value)
13     except:
14         self.userYearInput.error_text = "Please enter a valid year"
15         self.update()
16
17  def check_pass_strength(self, e):
18     try:
19         st = checkStrength(e.control.value)
20         if len(st["suggestion"]) > 0:
21             self.passwordStrengthText.value = f"Tip: {st['suggestion'][0]}"
22         else:
23             self.passwordStrengthText.value = ''
24         if st["strength"] >= 3:
25             self.passwordStrengthBar.color = ft.colors.GREEN
26         elif st['strength'] == 2:
27             self.passwordStrengthBar.color = ft.colors.AMBER
28         elif st['strength'] <= 1:
29             self.passwordStrengthBar.color = ft.colors.RED
30         self.passwordStrengthBar.value = st["strength"] / 4
31         self.update()
32     except:
33         self.passwordStrengthBar.value = 0
34         self.update()

```

```
1  def add_user(self, e):
2      try:
3          user_id = randint(1_000_000, 9_999_999)
4          currentDate = datetime.date.today()
5          if self.passowordStrengthBar.value * 4 >= 2:
6              self.page.db.addUser(user_id=user_id,
7              user_name=self.userNameInput.value, user_year=currentDate.year,
7              user_age=self.userAgeInput.value,
7              user_pass=self.userPasswordInput.value)
8
9              self.warningSnackBar.content = ft.Text("User Added!")
10             self.warningSnackBar.open = True
11             self.update()
12         else:
13             self.warningSnackBar.content = ft.Text(
14                 "Please enter a valid password")
15             self.warningSnackBar.open = True
16             self.update()
17     except:
18         pass
19
```

File: password.py

password.py

```
1 from zxcvbn import zxcvbn
2
3
4 def checkStrength(userPass):
5     """Gives Password strength and suggestion to improve the password
6
7     Args:
8         password (str): password entered by the user
9
10    Returns:
11        dict: {"strength": int, "suggestion": str}
12    """
13
14    result = zxcvbn(password=userPass, user_inputs=[])
15    return {"strength": result["score"], "suggestion": result["feedback"]
16            ["suggestions"]}
```

File: database.py

database.py

```
1import os
2import sqlite3
3class ORM:
4    def __init__(self):
5        self.db_location = './database.db'
6        self.db_connection = sqlite3.connect(
7            self.db_location, check_same_thread=False)
8        self.db_cursor = self.db_connection.cursor()
9        empty = self.db_cursor.execute(
10            "SELECT name FROM sqlite_schema").fetchall()
11        if empty == []:
12            self.__createDefaultTable()
13    def __createDefaultTable(self):
14        """
15        Private function: this function can't be accessed outside of this class
16
17        @summary: This Function creates basic database struct if database is not
18        found
19        """
20        self.db_cursor.executescript("""
21            CREATE TABLE Users (
22                UserID int,
23                UserName varchar(255),
24                UserYear int,
25                UserAge int,
26                UserPass varchar(255),
27                PRIMARY KEY (UserID),
28                UNIQUE(UserName)
29            );
30            CREATE TABLE Books (
31                BookID int,
32                BookName varchar(255),
33                BookDescription varchar(255),
34                BookYear int,
35                BookRating int,
36                BookRatedAge int,
37                BookStock int,
38                BookGenre varchar(255),
39                BookPrice int,
40                BookSellerID int,
41                PRIMARY KEY (BookID),
42                FOREIGN KEY (BookSellerID) REFERENCES
43                    Users(UserID)
44            );
45        """)
46        self.db_connection.commit()
```

```

1  def addBook(self, user_name, book_id, book_name, book_description, book_year,
2  book_rating, book_ratedage, book_stock, book_price, book_genre):
3      self.db_cursor.execute("""
4          SELECT * FROM Users
5          WHERE UserName=?
6          """, (user_name,))
7      user = self.db_cursor.fetchall()
8      self.db_cursor.execute("""
9          INSERT INTO Books(
10             BookID,
11             BookName,
12             BookDescription,
13             BookYear,
14             BookRating,
15             BookRatedAge,
16             BookStock,
17             BookGenre,
18             BookPrice,
19             BookSellerId
20         )
21         VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?);
22         """, (book_id, book_name, book_description, book_year,
23             book_rating, book_ratedage, book_stock, book_genre, book_price, user[0][0]))
24     self.db_connection.commit()
25
26     def addUser(self, user_id, user_name, user_year, user_age, user_pass):
27         self.db_cursor.execute("""
28             INSERT INTO Users(
29                 UserID,
30                 UserName,
31                 UserYear,
32                 UserAge,
33                 UserPass
34             )
35             VALUES (?, ?, ?, ?, ?);
36             """, (user_id, user_name, user_year, user_age,
37                 user_pass))
38     self.db_connection.commit()

```

```
1  def getUser(self, user_name):
2      self.db_cursor.execute("""
3          SELECT * FROM Users
4          WHERE UserName=?
5          """, (user_name,))
6      user = self.db_cursor.fetchall()
7      return user[0]
8
9  def getAllBooks(self):
10     self.db_cursor.execute("""
11         SELECT * FROM Books
12         """)
13     books = self.db_cursor.fetchall()
14     return books
15
16  def getBook(self, book_id):
17     self.db_cursor.execute("""
18         SELECT * FROM Books WHERE BookID=?
19         """, (book_id,))
20     books = self.db_cursor.fetchall()
21     return books
22
23  def getGenre(self):
24     self.db_cursor.execute("""
25         SELECT DISTINCT BookGenre FROM BOOKS
26         """)
27     genre = self.db_cursor.fetchall()
28     return genre
29
30  def updateBook(self, book_id, book_stock):
31     self.db_cursor.execute("""
32         UPDATE Books
33         SET BookStock=?
34         WHERE BookID=?
35         """, (book_stock, book_id))
36     self.db_connection.commit()
```


Output

Homepage

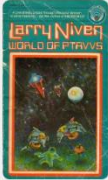
DEEZ Books

+ Search here

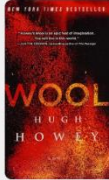
🔍

Sci-fi


Show Menu button




₹ 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

Book Display Card


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


Login

Sign Up


Edit Books

Fantasy

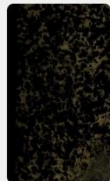
Popup Menu




₹ 685
Alice's Adventures in Wonderland




₹ 777
Treasure Island



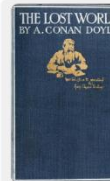
₹ 514
Gulliver's Travels




₹ 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 avventure di Pinocchio

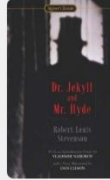
Horror




₹ 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

← Login

Login

User Name

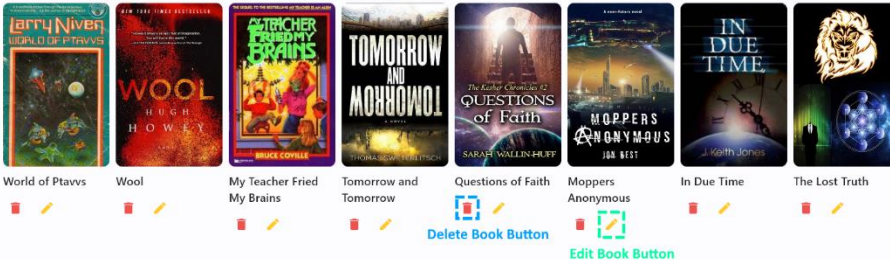
User Password

→ Login

Book Edit Page

← Edit Books

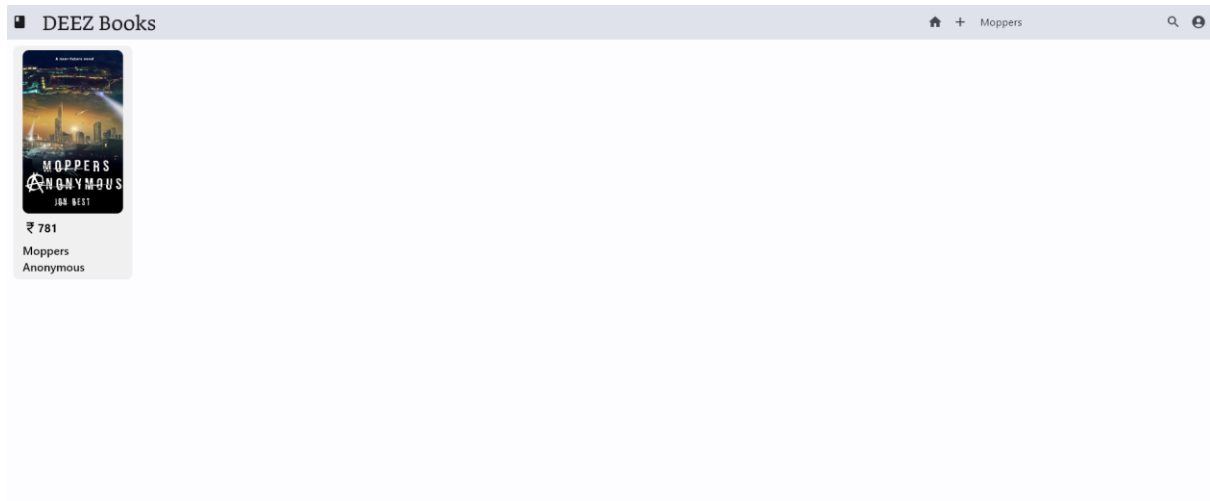
| Sci-fi



| Fiction

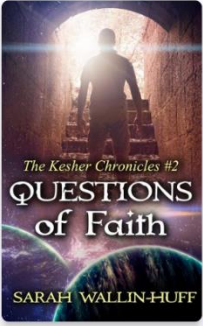


Search Page



Book View

[←](#) Book Details

The book cover for 'Questions of Faith' by Sarah Wallin-Huff. It features a silhouette of a person standing in a doorway, looking out at a bright, hazy landscape. The title 'QUESTIONS of Faith' is prominently displayed in the center, with 'The Keshar Chronicles #2' above it and the author's name 'SARAH WALLIN-HUFF' at the bottom.

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 Hōkūtan 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.

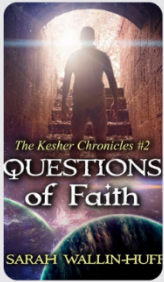
Genre: Sci-Fi
Publish Year: 2018
Rating: 2
Minimum Age: 7

Currently only 16 books left

[Buy](#)

Checkout Page

← Checkout



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

[Confirm Purchase](#)

For further inquiry contact us at: [here](#)

Add Book Page

← Add Book

 Add cover image

Name

Description

Publish year

Age rating

Genre

Stock

Price

 Add book

Conclusion

In this project, the basic principles of python programming, sql database 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 referred 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>