Dharmsinh Desai University,Nadiad

Faculty Of Technology

Department Of Computer Engineering

B. Tech. CE Semester – IV

Subject: Software Engineering

Project title: *SKOOB(Online Book Reselling Portal)*

Developed By:

Bhakti K. Sanghani (CE-218)
Namrata T. Vyas (CE-228)

Guided By:

Prof. Ankit P. Vaishnav
Prof. Jigar Pandya

Contents:-

1.Abstract

SKOOB is an online book reselling portal which acts as a central database containing a wide collection of books and genres to choose from, including fiction and fantasy, crime and thrillers, cooking, horror, textbooks, historical books and many more.

SKOOB acts as a medium between owner and buyer of the book. The owner can sell the used books which, despite having been preloved, are still in optimum condition and at a great price.The user visiting the website can see the wide range of books arranged in respective categories.The user may select the desired book and view its price and can purchase it.

2.Introduction

Hi.We, Namrata Vyas and Bhakti Sanghani had taken up this project as a part of our curriculum in the 4th semester.The project that we selected involves creating an online portal for reselling books.

Reselling a book offline is a very time and tedious task for both the owner and the buyer.Hence we had come up with an alternative that would smoothen the process ,Where books can be sold and bought from the comfort of home through the internet.

Through SKOOB,the Owner could add the book by providing the book details,upload some pictures and set a price.The Administrator approve the books and manage the users.The Buyer can view or search for specific book using search bar and filters,select the book, add it into the cart and can purchase it by providing the payment details.After receiving the book,the buyer can also provide feedback and rating.

3.Software Requirement Specification

R1.Manage Books information :

Description : Add books as per its categories and update its details , view books and search books ,sell books and collect feedback of books, delete books if required.

R1.1 Add books :

Input : seller provides details of the books.

Output : books added.

R1.2 display books :

Input : user can see books.

Output : books are displayed.

R1.2.1 search books :

Input : Provide name of the book to search.

Output : display book that user search.

R1.3 Update details :

Input : seller update details of his/her own book.

Output : details updated.

R1.4 delete books :

Input : delete books that authentication person want to delete.

Output : book deleted.

R2.Manage Categories :

Description : Add category and its subcategory and remove it.

R2.1 Add categories and subcategories :

Description:we can add categories comedy,horror,fictional,etc.
Input : add various categories and subcategories.
Output : added.

R2.2 Remove categories :

Input : remove particular category.

Output : deleted.

R3.Manage Selling Books :

Description : Users can sell books , add to their cart , purchase books and payment amount for it.

R3.1 Sell books :

Description : user provides books for selling.

R3.1.1 Add books by seller :

Input : add book and its details by seller.

Output : confirmation message for adding book.

R3.1.2 : display book :

Input : user can see books.

Output : show books.

R3.1.3 : Update book details :

Input : seller update details of his/her own book.

Output : details updated.

R3.1.4 delete books :

Input : delete books that authentication person want to delete.

Output : book deleted.

R3.2 Purchase books :

Input : select books that the user wants to purchase.

Output : add it to the cart.

R3.2.1 Add it to cart :

Input : add books to cart that we select for purchase.
Output : added in cart.

R3.2.2 get Payment details:

Input : payment for the book that we purchase.

Output : payment confirmation message.

R3.2.3 Delete books from cart :

Input : delete books that we don’t want.

Output : deleted from cart.

R4.Manage users :

Description : register users and view user details and its books and see its feedback , if the user is not appropriate then delete him/her.

R4.1 Register :

Description : users must be registered to buy books and sell.

Input : user must enter details.

Output : confirmation message displayed.

R4.2 Login :

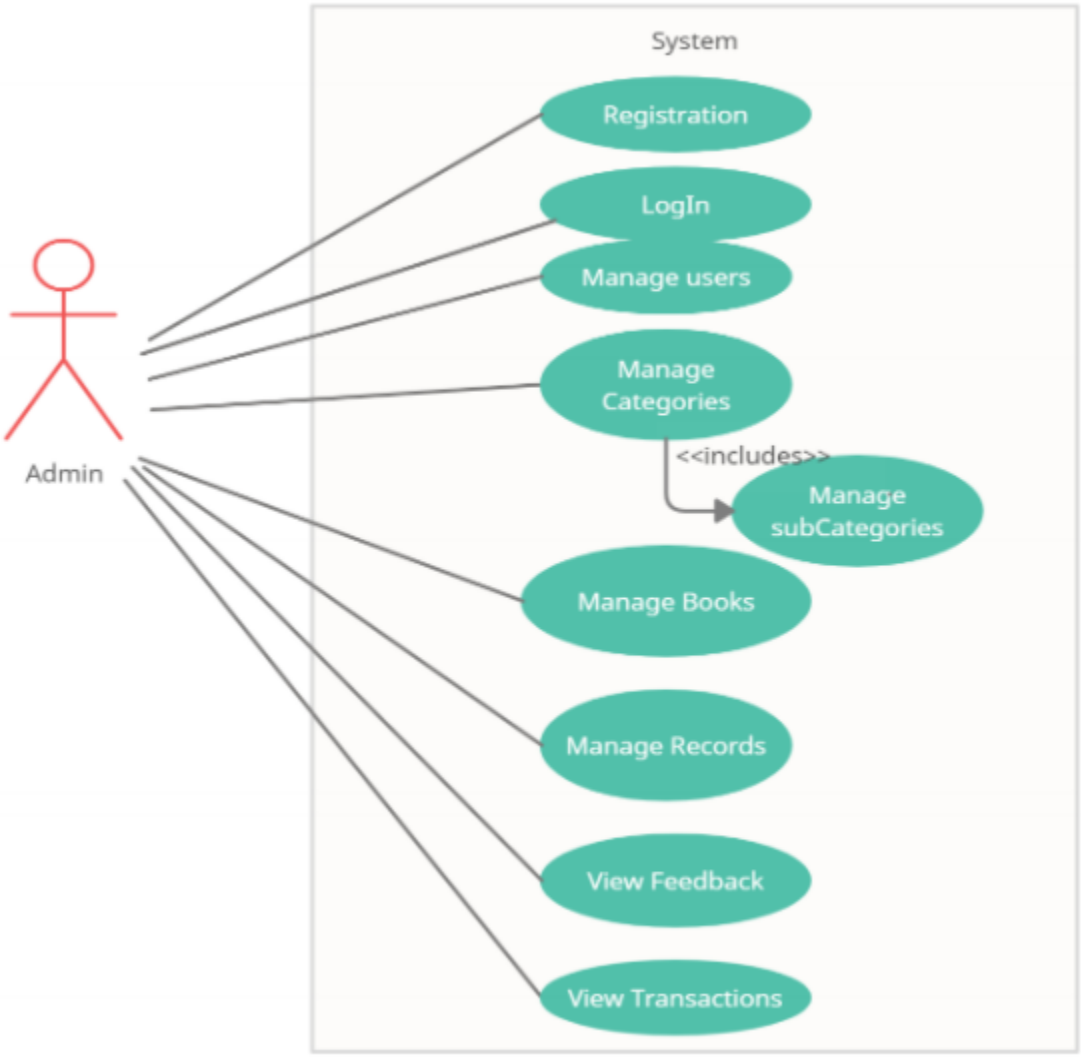
Input : user has to enter username and password.

Output : confirmation successfully and main page displayed.

4.Design

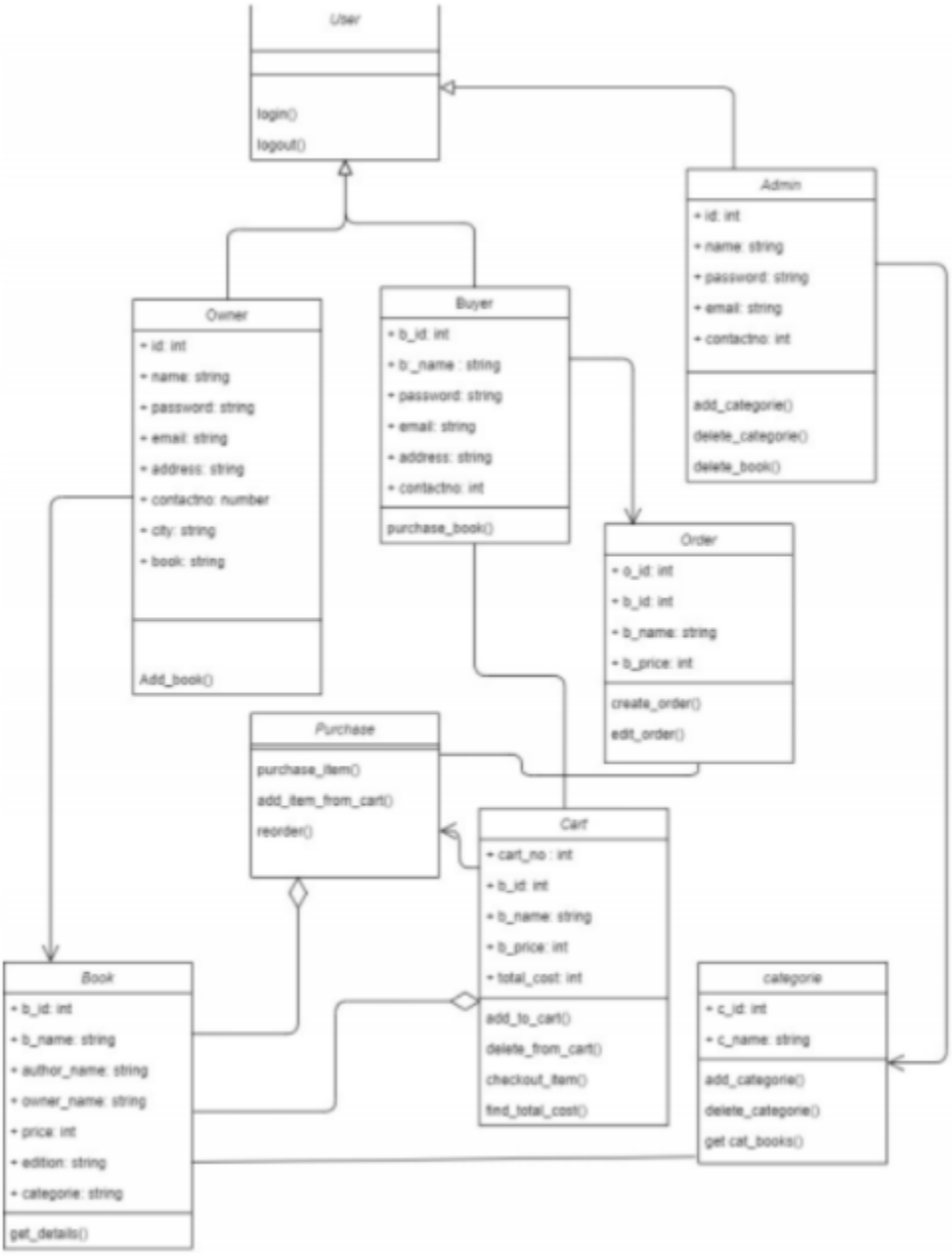
4.1 Use Case diagram

- ***Admin***

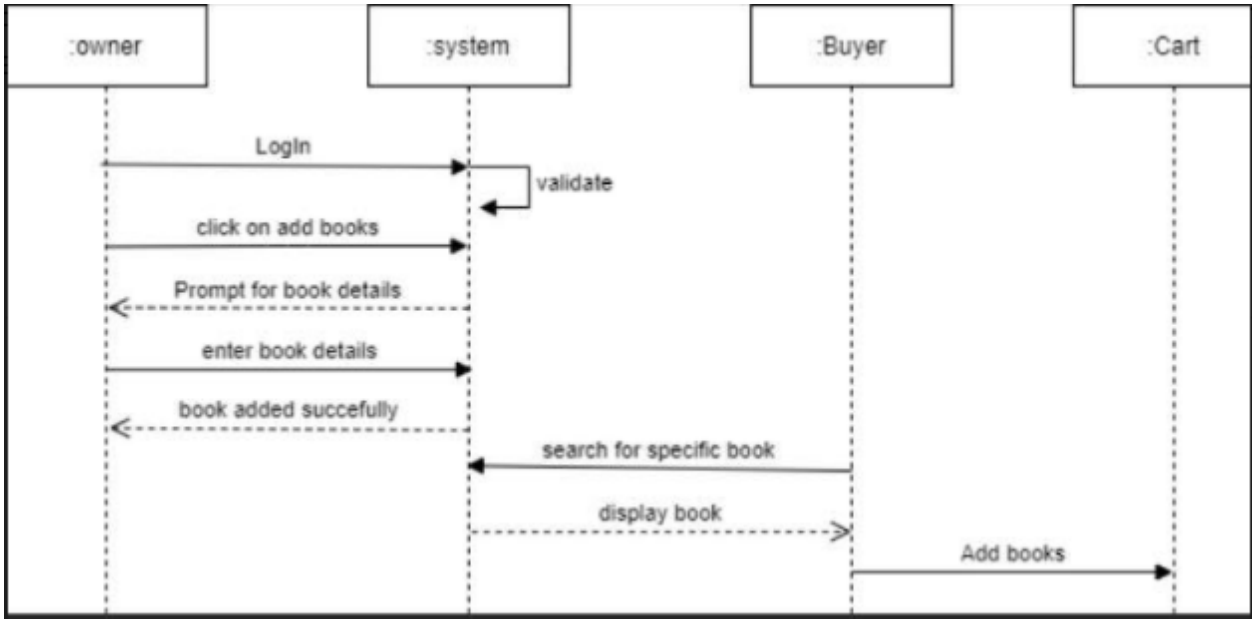


- ***User(Buyer/Seller)***

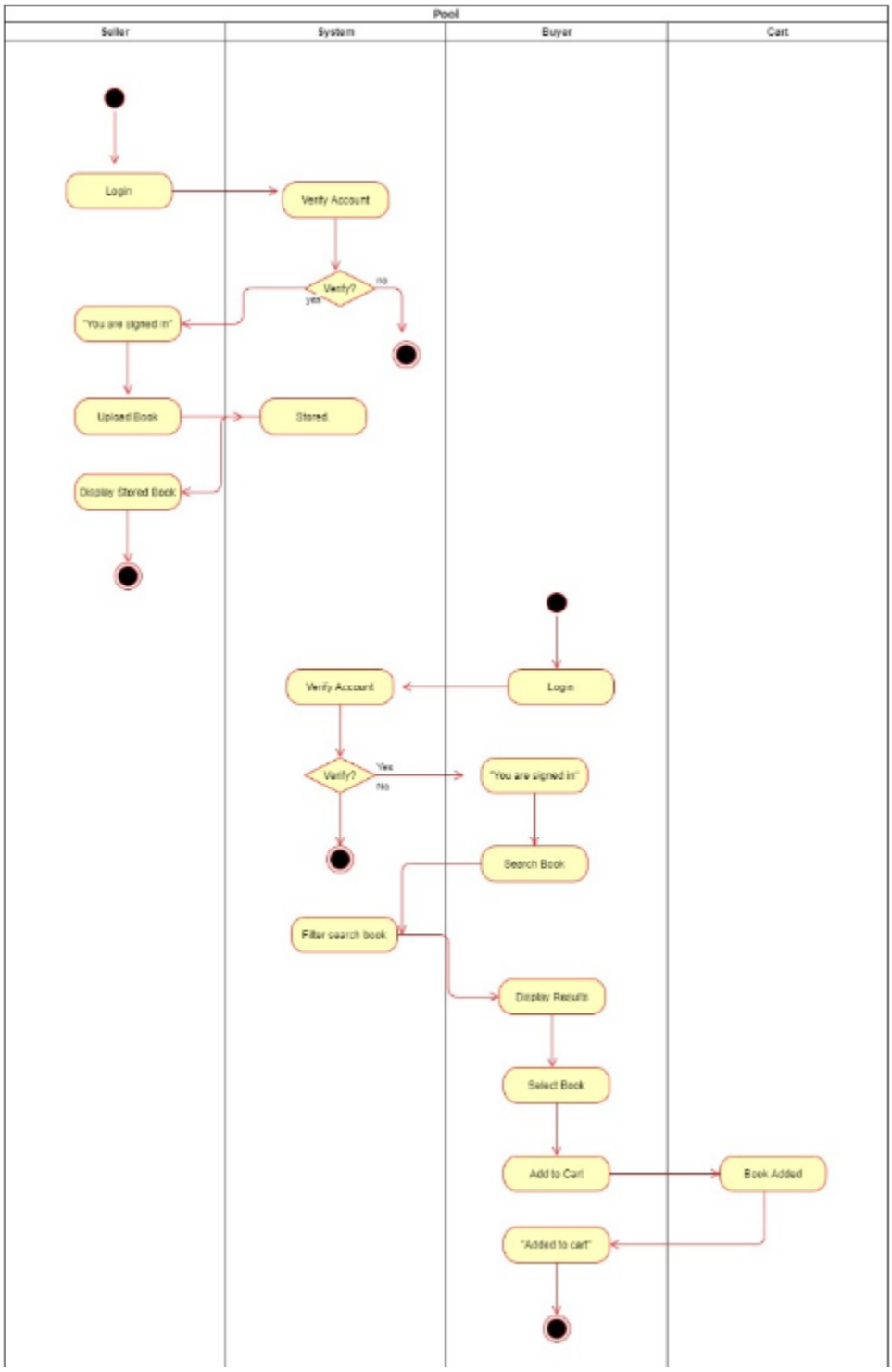
4.2 Class diagram



4.3 Sequence diagram

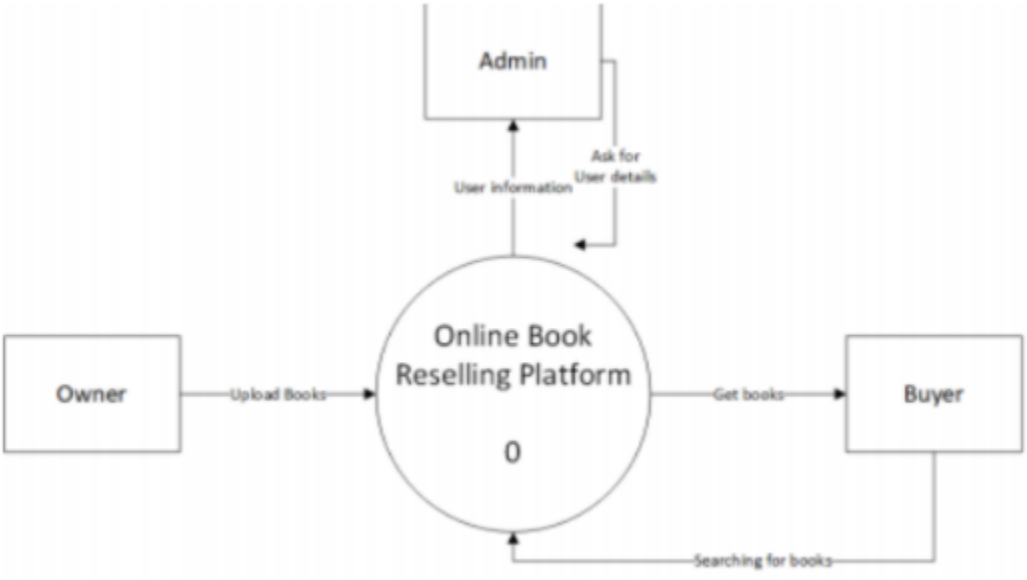


4.4 Activity diagram

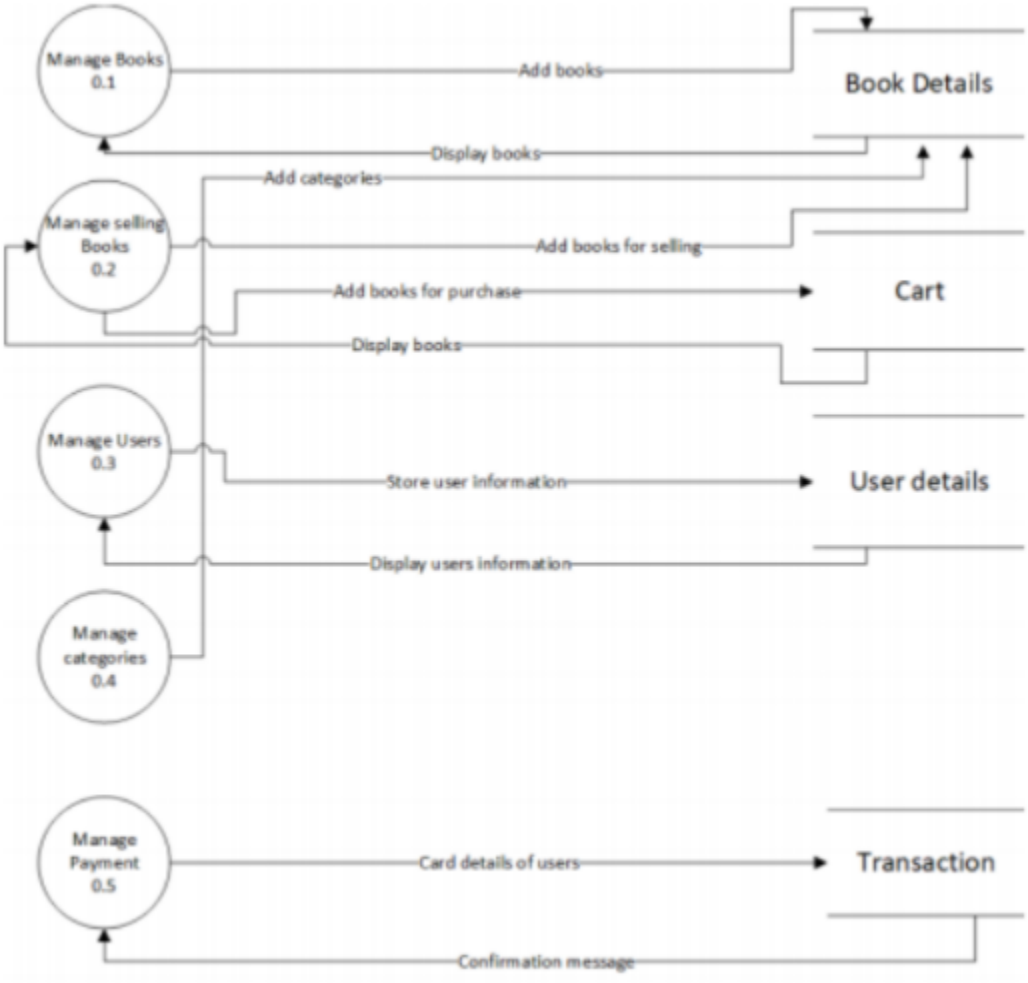


4.5 Data Flow diagram

- ***Level 0(Context Diagram)***



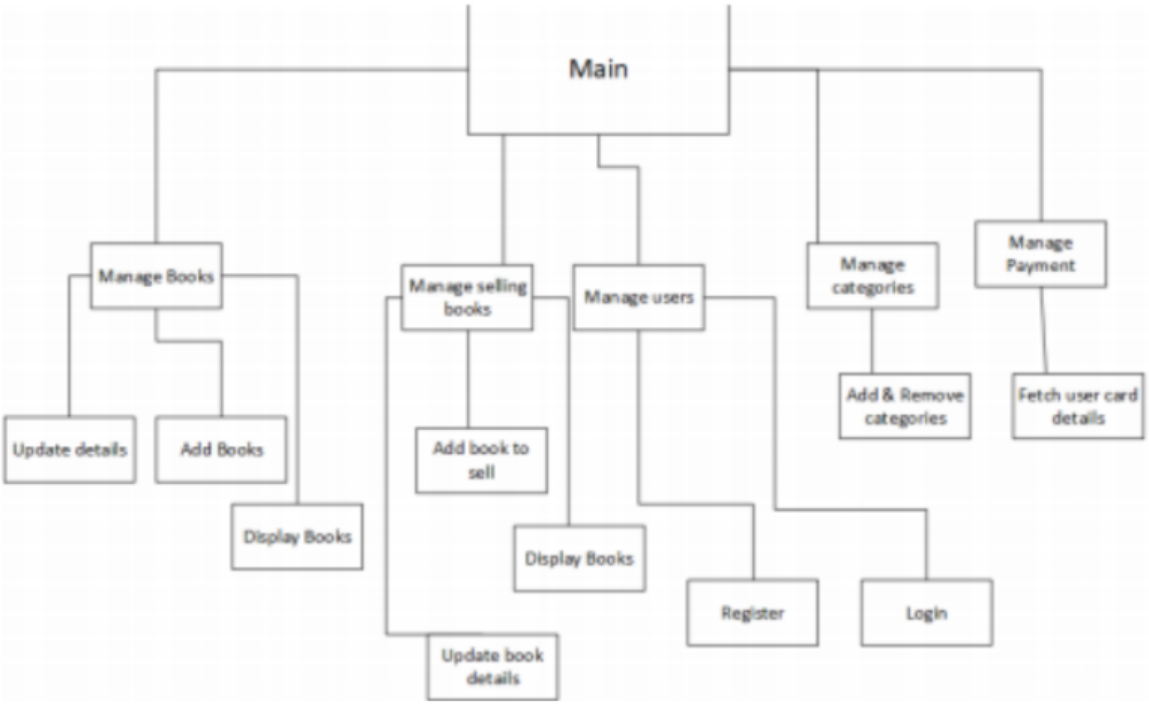
- ***Level 1***


Manage Books
0.1
Add books
Book Details
Display books
Add categories
Manage selling
Books
0.2
Add books for selling
Add books for purchase
Cart
Display books
Manage Users
0.3
Store user Information
User details
Display users information
Manage
categories
0.4
Manage
Payment
0.5
Card details of users
Transaction
Confirmation message

- ***Level 2***


Registration Details
Registration
0.1.1
Update Book information
Add books
Book Details
Delete book
View Books
0.1.2
Book details
Registration Details
Registration
0.2.1
Search books
Book Details
Add books to cart that user want to buy
Cart
Show books
View books
0.2.2
Cart Items
0.2.3
Cart
User
Cart Items
0.3.1
Book details
Book details and
user card details
Payment
0.3.3
Transaction details
Purchase Book
0.3.2
Transaction

Structure Chart



4.6 Data dictionary

Books

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**b_id**	int(20)			No	*None*		AUTO_INCREMENT
2	**b_name**	varchar(20)	utf8mb4_general_ci		No	*None*		
3	**b_category**	varchar(20)	utf8mb4_general_ci		No	*None*		
4	**b_author**	varchar(20)	utf8mb4_general_ci		No	*None*		
5	**b_user**	varchar(20)	utf8mb4_general_ci		No	*None*		
6	**b_price**	int(20)			No	*None*		
7	**b_edition**	int(20)			No	*None*		
8	**b_image**	varchar(50)	utf8mb4_general_ci		No	*None*		
9	**b_description**	varchar(50)	utf8mb4_general_ci		No	*None*		

Cart

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**cart_no**	int(20)			No	*None*		AUTO_INCREMENT
2	**b_id**	int(20)			No	*None*		
3	**b_name**	varchar(20)	utf8mb4_general_ci		No	*None*		
4	**b_price**	int(20)			No	*None*		
5	**user**	varchar(20)	utf8mb4_general_ci		No	*None*		
6	**active**	tinyint(1)			No	*None*		
7	**order_date**	date			No	*None*		
8	**total_cost**	int(20)			No	*None*		

BookOrder

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**book**	int(20)			No	*None*		
2	**cart**	int(20)			No	*None*		
3	**quantity**	int(20)			No	*None*		

☐ *Category*

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**c_id** 🔑	int(20)			No	*None*		AUTO_INCREMENT
2	**c_name**	varchar(20)	utf8mb4_general_ci		No	*None*		

☐ *Users*

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**u_id** 🔑	int(20)			No	*None*		AUTO_INCREMENT
2	**u_name**	varchar(20)	utf8mb4_general_ci		No	*None*		
3	**u_email**	varchar(20)	utf8mb4_general_ci		No	*None*		
4	**u_pass**	varchar(20)	utf8mb4_general_ci		No	*None*		
5	**u_add**	varchar(20)	utf8mb4_general_ci		No	*None*		
6	**u_city**	varchar(20)	utf8mb4_general_ci		No	*None*		
7	**u_contact**	int(20)			No	*None*		
8	**u_book**	varchar(20)	utf8mb4_general_ci		No	*None*		

☐ *Admin*

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**a_id**	int(20)			No	*None*		
2	**a_name**	varchar(20)	utf8mb4_general_ci		No	*None*		
3	**a_email**	varchar(20)	utf8mb4_general_ci		No	*None*		
4	**a_pass**	varchar(20)	utf8mb4_general_ci		No	*None*		

5.Implementation

Code :

- *User side register,login,add book,store book,show book,cart,buyer side,seller side*

```
from django.shortcuts import render,redirect
from django.utils.datastructures import MultiValueDictKeyError
from django.contrib import messages
from django.contrib.auth.models import User,auth
from firstapp.models import books,cart,category,admin,users,bookorder
from django.contrib.auth.decorators import login_required
from .decorators import requiredregister,allowed_users
from django.views.generic import ListView
from firstapp.forms import BooksForm
from django.utils import timezone
from django.http import HttpResponse
# Create your views here.
def index(request):
    return render(request, "index.html")

def register(request):
    if request.method=='POST':
        username=request.POST.get('username')
        address=request.POST.get('address')
        city=request.POST.get('city')
        contact=request.POST.get('contact')
        email=request.POST.get('email')
        password=request.POST.get('password')
        cpassword=request.POST.get('cpassword')

        user=User.objects.create_user(username=username,password=password,email=email)
        user.save()
        print("user created")
        return redirect('login')
    else:
        return render(request, 'register.html')
```

```python
@requiredregister
@allowed_users(allowed_roles=['admin','buyer','seller'])
def login(request):
    if request.method=="POST":
        username=request.POST['username']
        password=request.POST['password']
        user = auth.authenticate(username=username,password=password)
        if user is not None:
            auth.login(request, user)
            return render(request, 'who.html')
        else:
            messages.info(request, 'invalid creditionals')
            return redirect('login')
    else:
        return render(request, 'login.html')

@login_required(login_url='login')
def logout(request):
    auth.logout(request)
    return render(request, 'index.html')

def about(request):
    return render(request, 'about.html')

@login_required(login_url='login')
@allowed_users(allowed_roles=['buyer','seller'])
def who(request):
    return render(request, 'who.html')
```

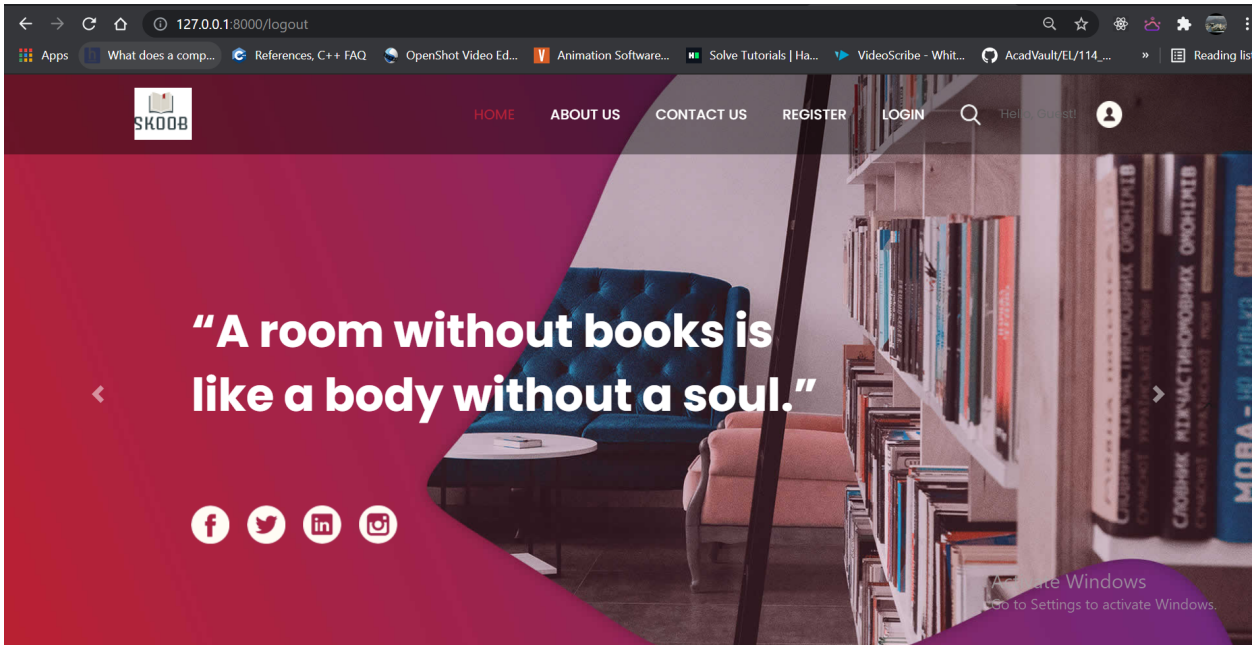
```python
def contact(request):
    return render(request, 'contact.html')

@login_required(login_url='login')
def buyer(request):
    book=books.objects.all()
    #for b in book
    #print b.b_name,b.b_author,b.b_category,b.b_user,b.b_price,b.b_edition,b.b_image
    context={'b':book}
    return render(request, 'buyer.html',context)
    #return render(request, 'buyer.html')

@login_required(login_url='login')
def Home_Buyer(request):
    return render(request, 'Home_Buyer.html')


@login_required(login_url='login')
def profile(request):
    return render(request, 'profile.html')
```

```python
#book adding process
def bookform(request):
    if request.method == "POST":
        form = BooksForm(request.POST)
        if form.is_valid():
            try:
                form.save()
                return redirect('seller')
            except:
                pass
    else:
        form = BooksForm()

    return render(request, 'add_book.html',{'form':form})

def storebook(request):
    if request.method == 'POST':
        book=books()
        book.b_name=request.POST['b_name']
        book.b_category=request.POST['b_category']
        book.b_author=request.POST['b_author']
        book.b_price=request.POST['b_price']
        book.b_edition=request.POST['b_edition']
        book.b_image=request.POST['b_image']
        book.save()
    return render(request,'index.html')
```

```python
@login_required(login_url='login')
def seller(request):
    book=books.objects.all()
    #for b in book
    #print b.b_name,b.b_author,b.b_category,b.b_user,b.b_price,b.b_edition,b.b_image
    context={'b':book}
    return render(request, 'seller.html',context)

def category_list(request):
    category = None
    categories = category.objects.all()
    return render(request, 'bookform.html',{'category':category})

#@login_required(login_required='login')
def Home_seller(request):
    return render(request, 'Home_seller')

@login_required(login_url='login')
class Search(ListView):
    model = books
    template_name = 'search.html'

    def get_queryset(self): # new
        query = self.request.GET.get('q')
        return books.objects.filter(Q(b_name=query) | Q(b_author=query))
```

Output :

Home Page



Registration Page

Enter user Details

username

address

city

contact

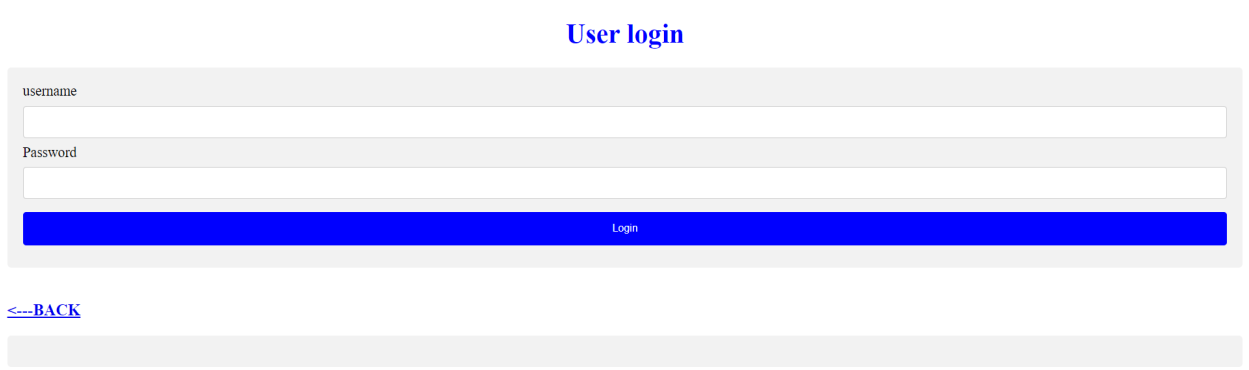
email

Password

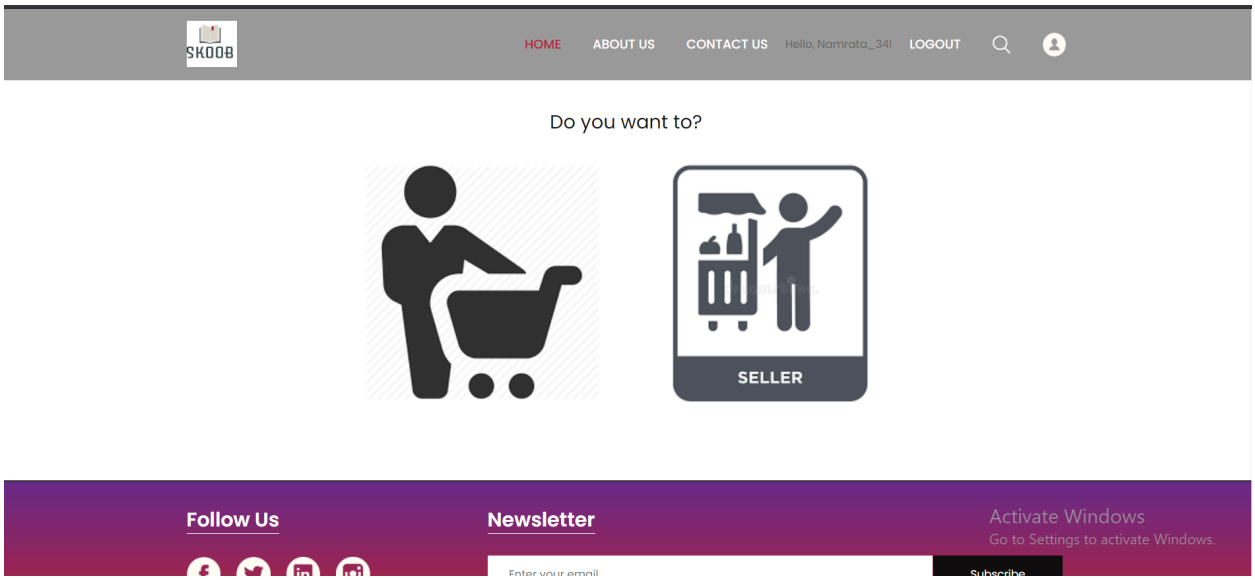
CPassword

Register me

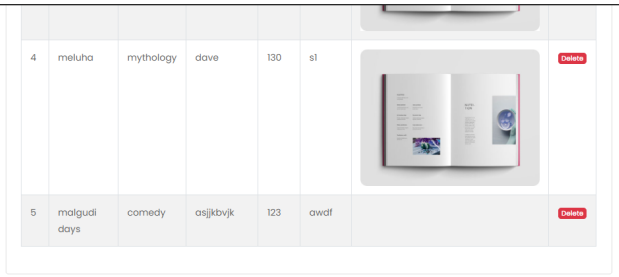
Login Page



Role Page



Seller side

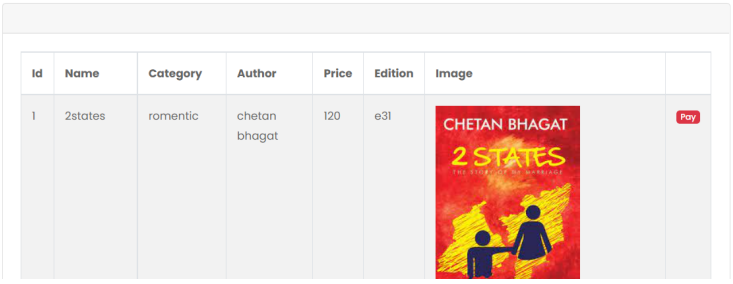


Add book

Buyer side :

You can search for book here....

Search..



Payment

Order Summary

Success! Your order has been completed

7. Conclusion

The Online Reselling Portal "SKOOB" provides a one place solution to all the processes occurring during the offline mode.It provides an easy way of selling,searching, keeping record and purchasing of books "ANYTIME.. ANYWHERE".

In brief,It increases both the ease of seller and buyer.

8. Limitations and Future Enhancements and Scope

- ***Limitations***

Here,we are unable to do Real Payment through a bank account so we created a dummy account for it.

Here,we are not providing any facility for the delivery.

- ***Future Enhancement***

We can provide the actual payment option.

We can provide the facility for the delivery.

Admin or Owner could provide discounts for some customers

- ***Future Scope***

This website can be used under various situations. Any private organization can make use of it for providing information about the author, content of the books in their database. Modifications can be easily done according to user requirements. It can be used in any type of Book Shop System for managing the transactions and purchasing activities by users and managing the information related to Books.

9. Reference / Bibliography

Bibliography:-

- ***Software Engineering - A Practitioner's Approach***
  Author: Roger S. Pressman
- ***Fundamentals of Software Engineering***
  Author: Rajib Mall

URLs:-

https://www.youtube.com/c/DennisIvy/videos
https://www.youtube.com/c/Telusko/videos
Stackoverflow.com
Github.com
Docs.djangoproject.com
Bootstarp.com

**** THANK YOU *****