WEB FRAMEWORKS

Software Requirements Specifications

BOOK STORE

Kokila K N. (PES2201800625)

Swathi (PES2UG19CS807)

Monica shree (PES2UG19CS811)

Deena (PES2UG19CS802)

INDEX

1. INTRODUCTION

The document aims at defining the overall software requirements for on-line book store. Efforts have been made to define the requirements exhaustively and accurately.

Project Definition

We find so many ebooks on internet but for every ebook we are going to different website and also few links lead to some adds and waste our time in search of the particular ebook .

The features available on many online bookstores also allow customers to compare similar titles with the click of a mouse and read reviews from professionals and customers.

1.1.Purpose

To get all ebooks on internet to one place.
This specification document describes the capabilities that will be provided by the software application ‘Book store’. It also states the various required constraints by
which the system will abide. The intended audience for this document are the development team, testing team and end users of the product.

1.2 Scope

The software system being produced is called book store .It is
being produced for a customer interested in selling books via the Internet. The Book E-Commerce System will allow any user to create an account to become a customer. The customer, through the process of account creation, will have the option to become a member of the site. The system will allow customers to browse, search, select, bookmark , read and download books .maintains. The system also allows a manager to manage the inven-

tory with full create, retrieve, update and delete functionality with regards to books in the system.

1.3 Overview

The SRS is divided into three major sections introduction, overall description and specific.

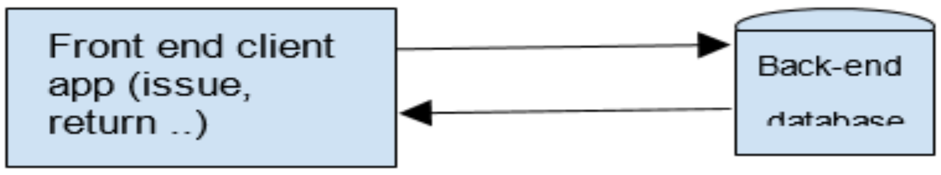
1.4 Business Context

Our entire project is available to users for free of costs we collect free ebook s on internet and present it to people.

2. GENERAL DESCRIPTION

2.1 Product Perspective:

The Online Shopping Bookstore shall be developed using client server architecture and will be compatible with Microsoft Windows operating system. Front end of the system will be developed using HTML5, CSS, javaScript, bootstrap, Django , react and for back end we will be concentrating on php



2.2 Product functions:

The Online Bookstore will allow access only to the authorised users with specific roles(system administrator, customer). Depending upon the users role, he /she will be able to access only specific modules of the system.
A summary of major functions that the Online Bookstore shall perform includes:

1. Login facility for enabling only authorised access to the system.
2. System administrator will be able to add modify delete view product and checks the uploaded books and confirms the
3. The customer will be able to read, download and upload book.

2.3User characteristics

Qualification: At least matriculation and comfortable with English.
Experience: Should be well versed about the process of University library.
Technical experience: Elementary knowledge of computers.

2.4Constraints

1. The software does not maintain records of periodicals.
2. There will be only one administrator.
3. The user will not be allowed to update the user id.
4. To reduce the complexity of the system, there is no check on delete operation.
5. The administrator should be very careful before editing of any product.

2.5User Objectives:-

1. Any person with a valid email id can create a account and use our website.

2.As there is only one admin there is no signing for admin .

3.Records of books read ,downloaded and uploaded will kept in count.

3.FUNCTIONAL REQUIREMENTS.

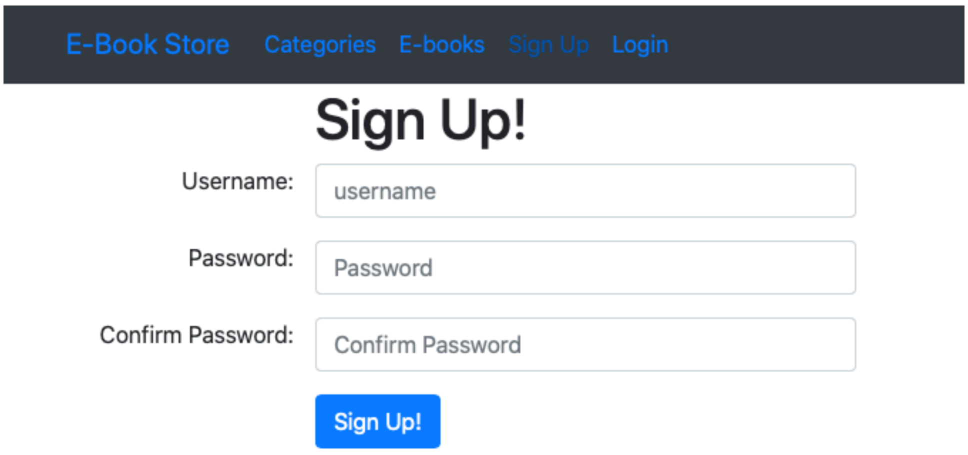
- Provides the searching facilities based on various factors such as genres ,highest rating, most searched.
- This will keep track of the books read and downloaded which can be further used most read and downloaded books.
- It tracks all the information of types of books read by customer and their interests.
- Shows the information and description of the Books to customers.
- Adding books , deleting books and uploading books can be done (only by admin)

4.USER INTERFACE

User is the person who want to study any Books. The main functions of the users are:

- User Registration

This interface will allow the customer to have an account for them to be easier to check in, the customer just need to fill up the empty fields to have an account, but if they already have an account the guest just need to click the link below create account button to proceed to Sign In page.



Sign Up!

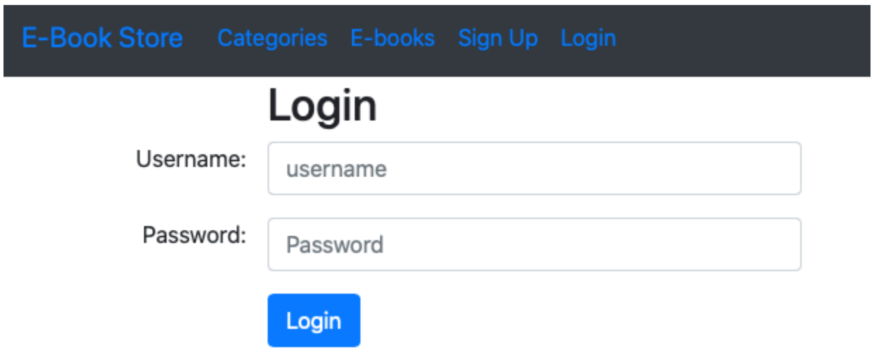
Username: username

Password: Password

Confirm Password: Confirm Password

- User Login

  This interface will allow the user to log in to the system, after you create an account only the username and password are required for you to log in.



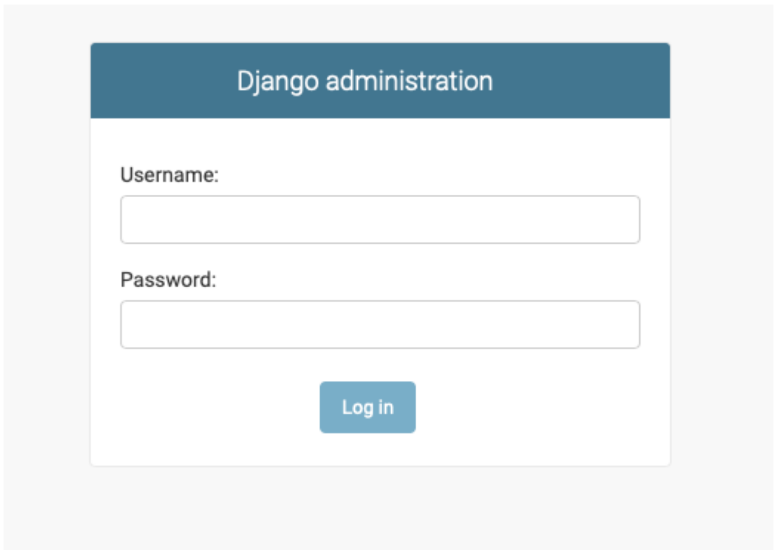
Login

Username: username

Password: Password

- Admin Login

  This interface will allow the admin to log in to the system, username and password are required for you to log in.



Check the details about Book

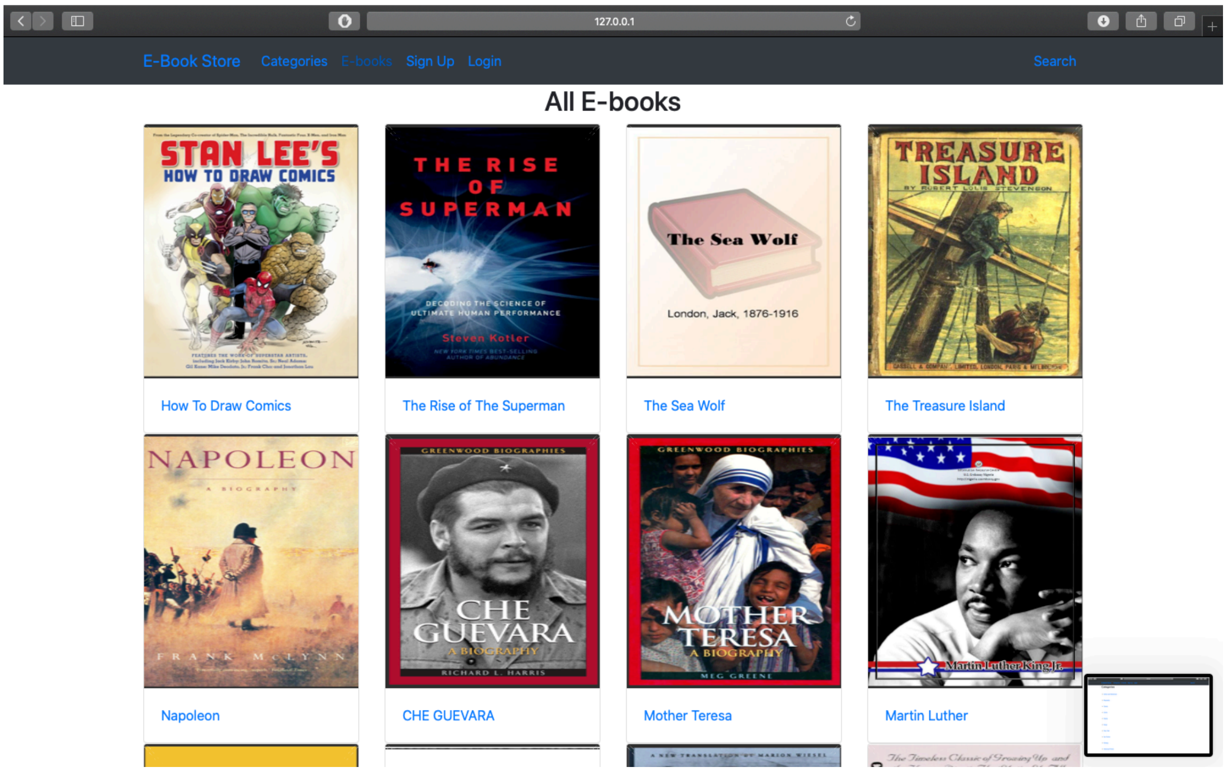
This interface will help customer to check the details of each book

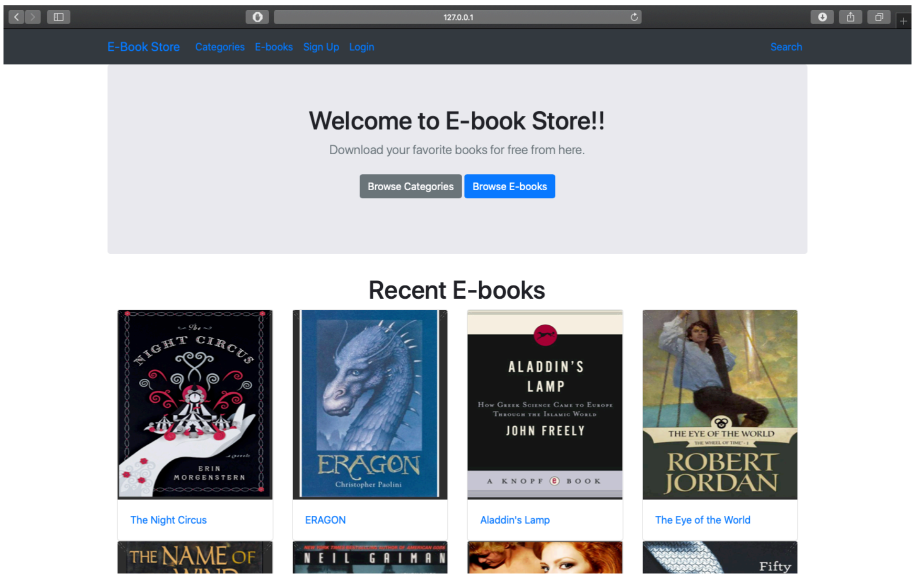
This Interface will help the customer to contact the admin about the books

This interface will help to search the book.

This interface will help to select required book read and download it.

Before implementing the actual design of the project, a few user interface designs were constructed to visualize the user interaction with the system as they browse for books.The user interface design will closely follow our Functional requirements.




E-Book Store Categories E-books Sign Up Login Search

Welcome to E-book Store!!

Download your favorite books for free from here.

Browse Categories Browse E-books

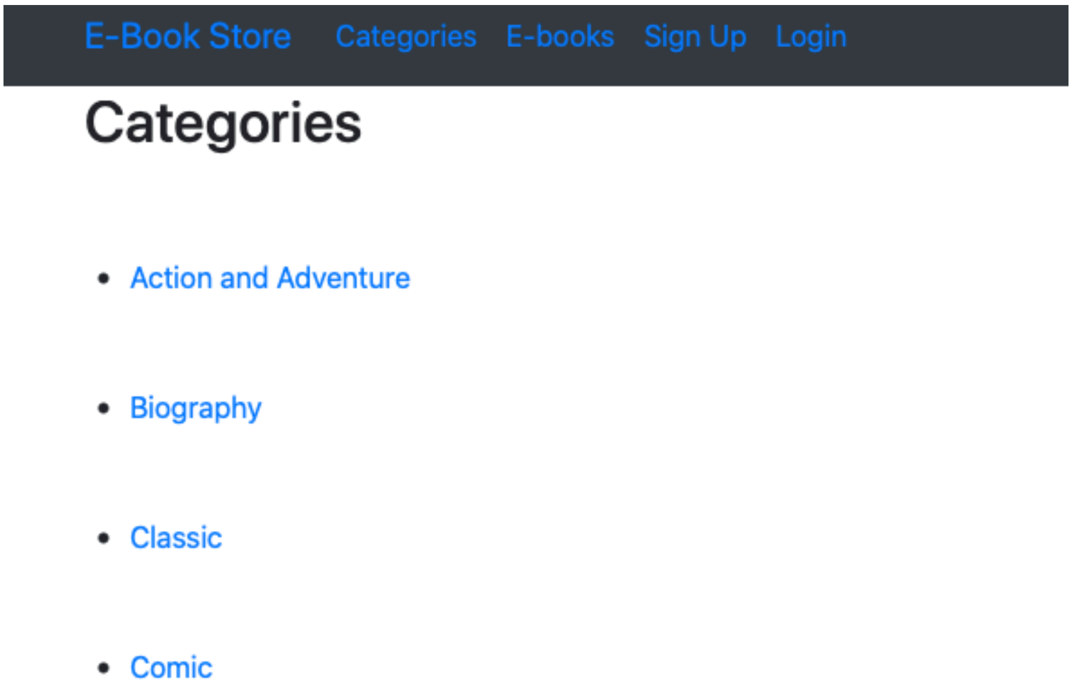
Recent E-books

The Night Circus

ERAGON

Aladdin's Lamp

The Eye of the World


E-Book Store Categories E-books Sign Up Login

Categories

- Action and Adventure
- Biography
- Classic
- Comic

5.PERFORMANCE REQUIREMENTS.

Performance requirement defines acceptable response times for system functionality. Although the system developed suiting for the least system performance, the performance of the system will highly depend on the performance of the hardware and software components of the installing computer. When consider about the timing relationship of the system the load time for user interface screen shall take no longer than two seconds. It makes fast access to system functions. The log in information shall be verified within five seconds. Returning query results within five seconds makes search function more accurate.

6.OTHER NON -FUNCTIONAL ATTRIBUTES

Security Requirement:

There are several user levels in the hotel management system, Access to the various subsystems will be protected by a user log in screen that requires a user name and password.

Customer service representatives will have access to the reservation/ booking sub system. Customer's personal details shall be encrypted.

Portability Requirement:

System shall be accessible on mobile devices. System shall also be accessible on Google Chrome, Firefox, Safari, Opera and Internet explorer.

Reliability Requirements:

System shall be accessible 98% of time.

7. OPERATIONAL SCENARIOS

Read or download book

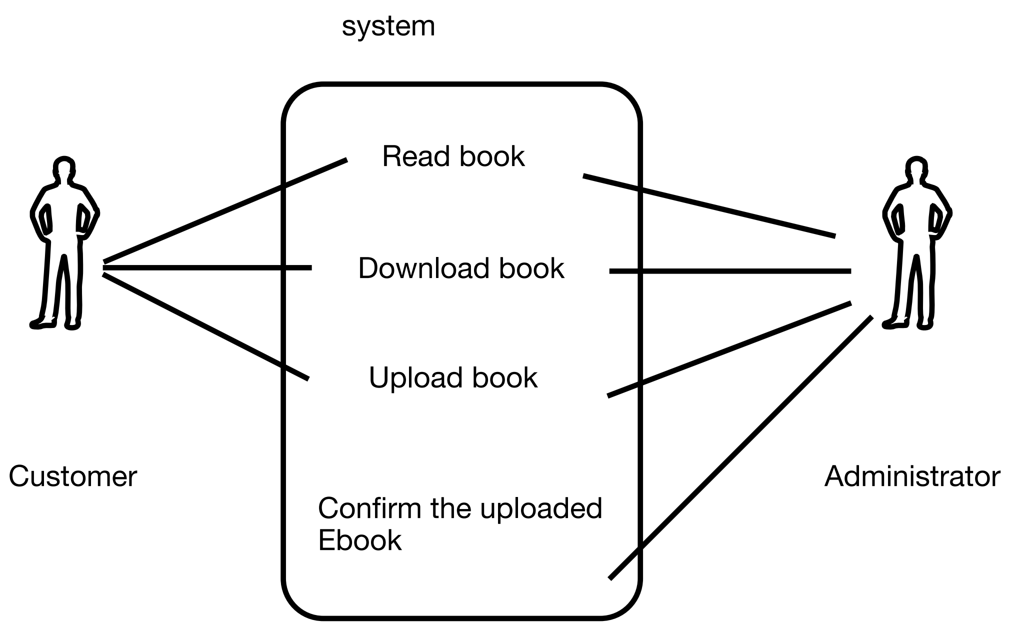
Use case	Check Ebook
Goal	To check the particular ebook availability
Actor	Admin
Pre condition	Login to system
Post condition	Read or Download particular e-book
Main Flow	Steps
	Display all categories
	Select required category
	Display all available books
	Opening the wanted book to read or download
Extensions	No book will be shown on absence such book

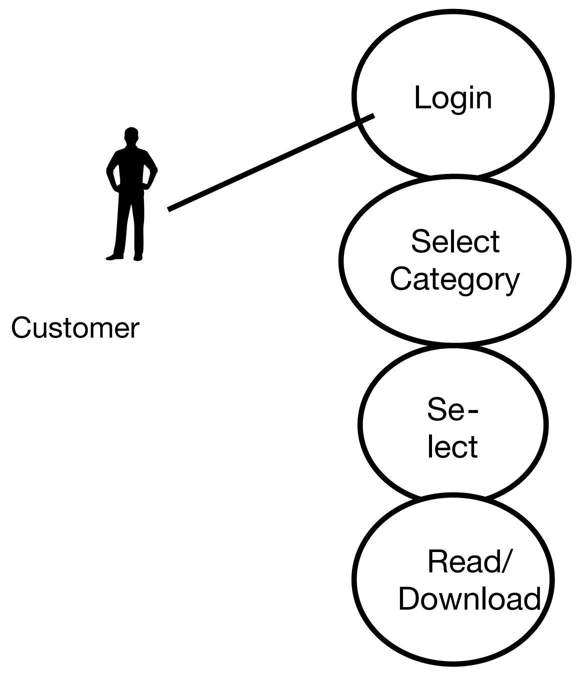
Use case	Upload ebook
Goal	To upload the ebook user have
Actor	Admin
Pre condition	Login to system
Post condition	Upload ebook
Main Flow	Steps
	Provide option to upload
	Take description author and other details of the book
	Admin will check the terms and upload the book
Extensions	Book won’t be uploaded if terms are not met

Upload book

8. PRELIMINARY USE CASE MODELS AND SEQUENCE DIAGRAMS

8.1 Use Case Model





8.2 Sequence Diagrams

User

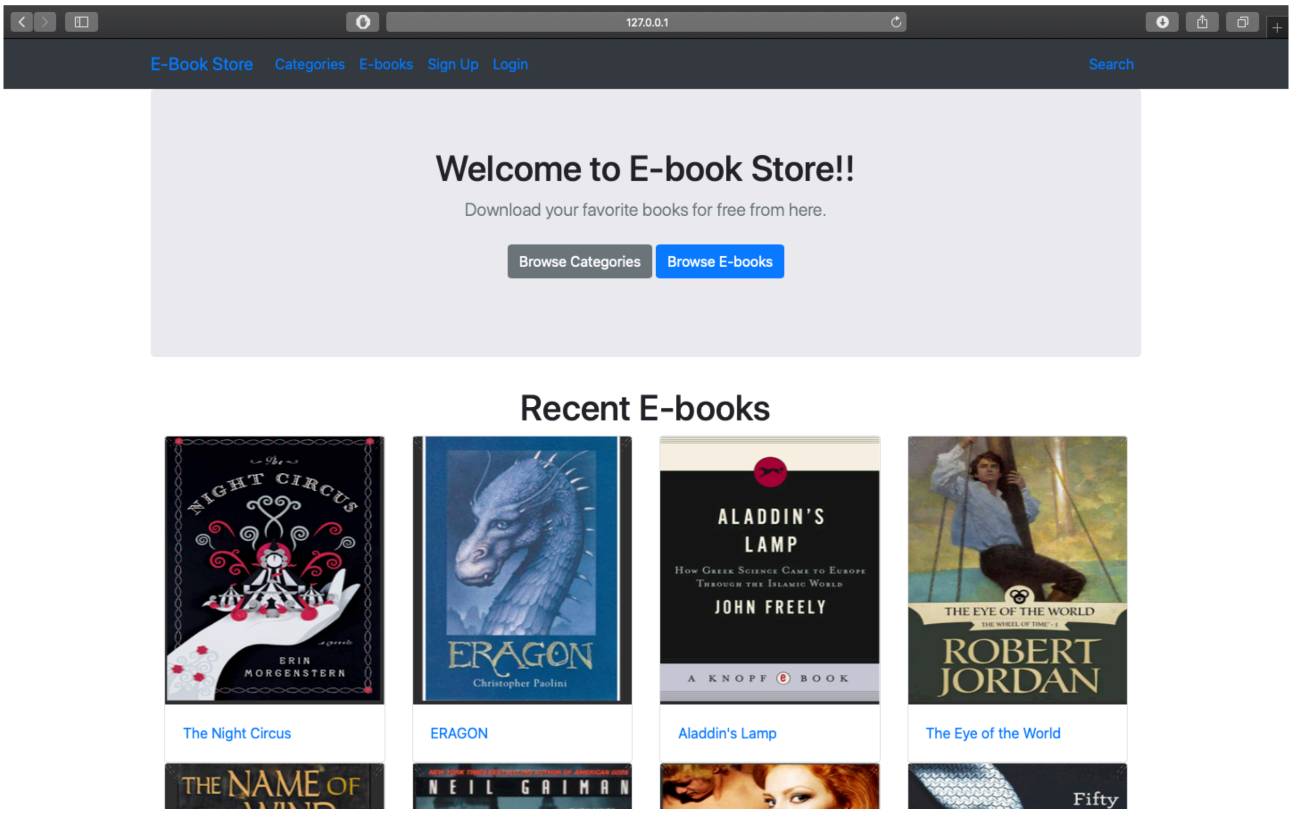
login		
search for a book	home page	home page
read	genres	upload
download	wanted book	
	read/download	

Admin

checks the terms of the uploaded book and upload to the database

9. UPDATED SCHEDULE

Home page:


127.0.0.1

E-Book Store Categories E-books Sign Up Login Search

Welcome to E-book Store!!

Download your favorite books for free from here.

Browse Categories Browse E-books

Recent E-books

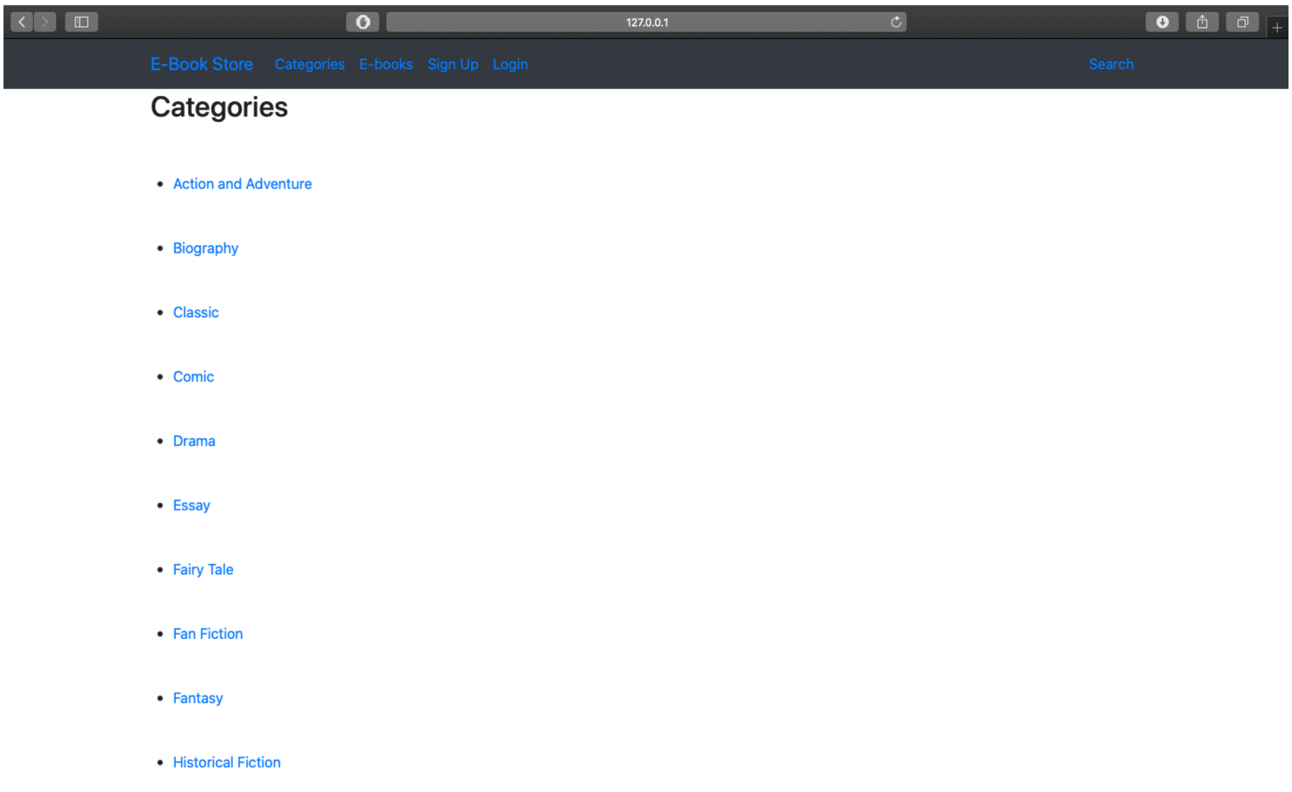
The Night Circus

ERAGON

Aladdin's Lamp

The Eye of the World

Categories Page:

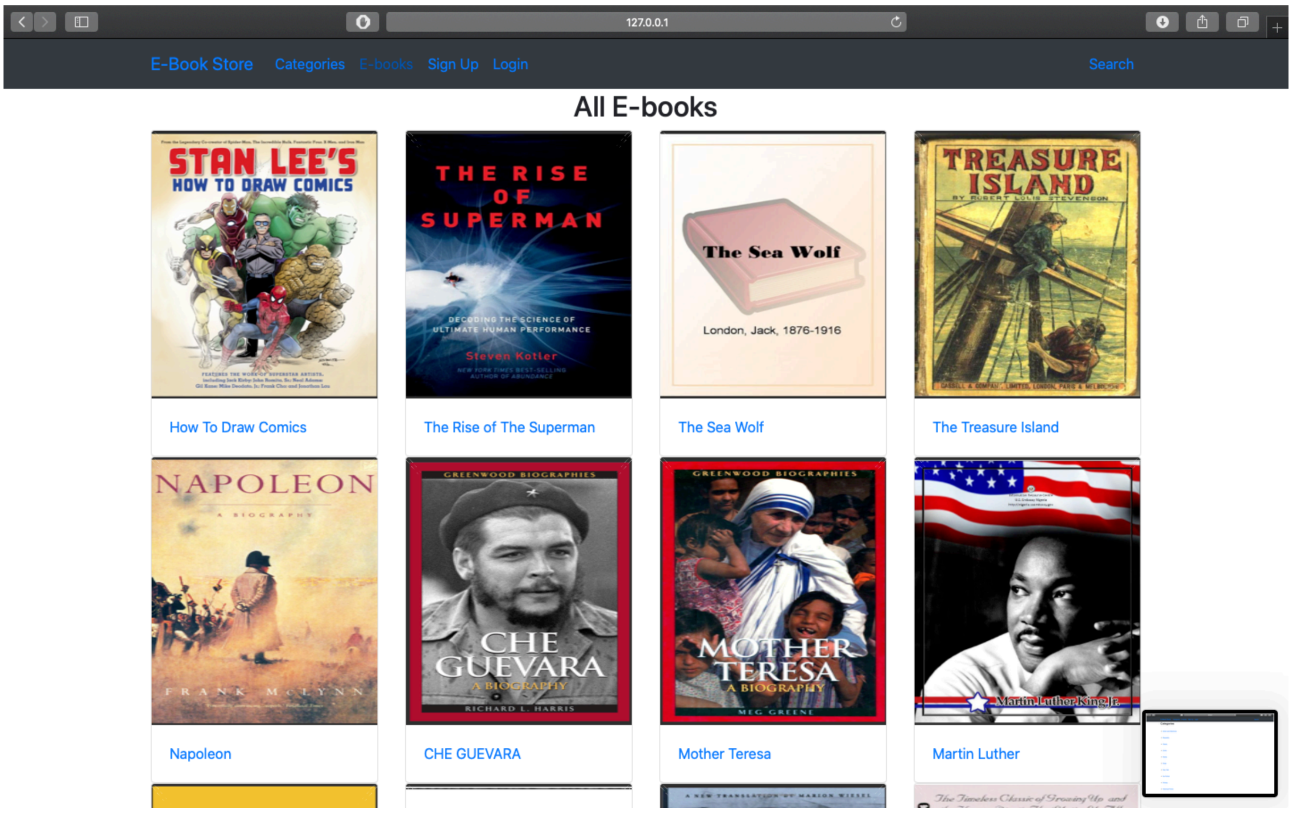

127.0.0.1

E-Book Store Categories E-books Sign Up Login Search

Categories

- Action and Adventure
- Biography
- Classic
- Comic
- Drama
- Essay
- Fairy Tale
- Fan Fiction
- Fantasy
- Historical Fiction

Ebooks page:



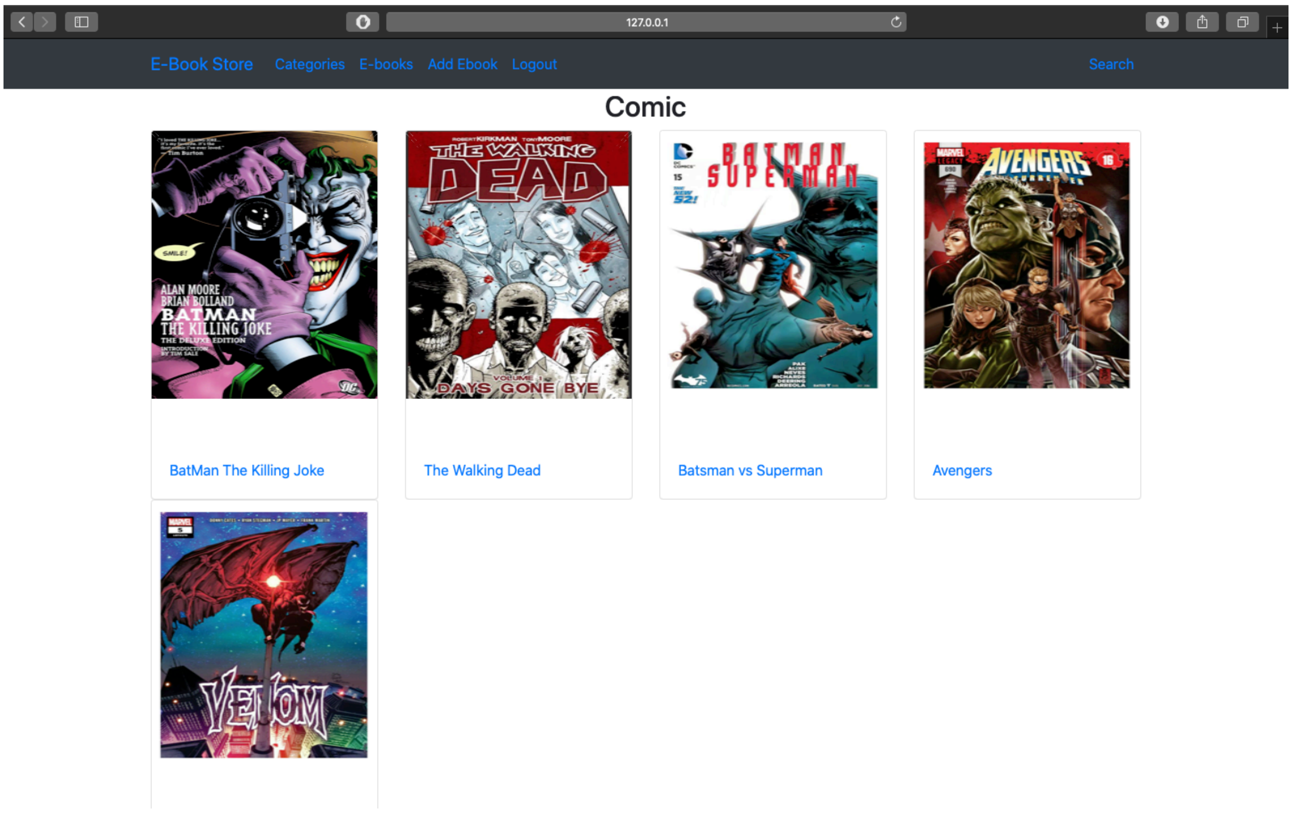
127.0.0.1

E-Book Store Categories E-books Sign Up Login Search

All E-books

How To Draw Comics
The Rise of The Superman
The Sea Wolf
The Treasure Island
Napoleon
CHE GUEVARA
Mother Teresa
Martin Luther

Category page:



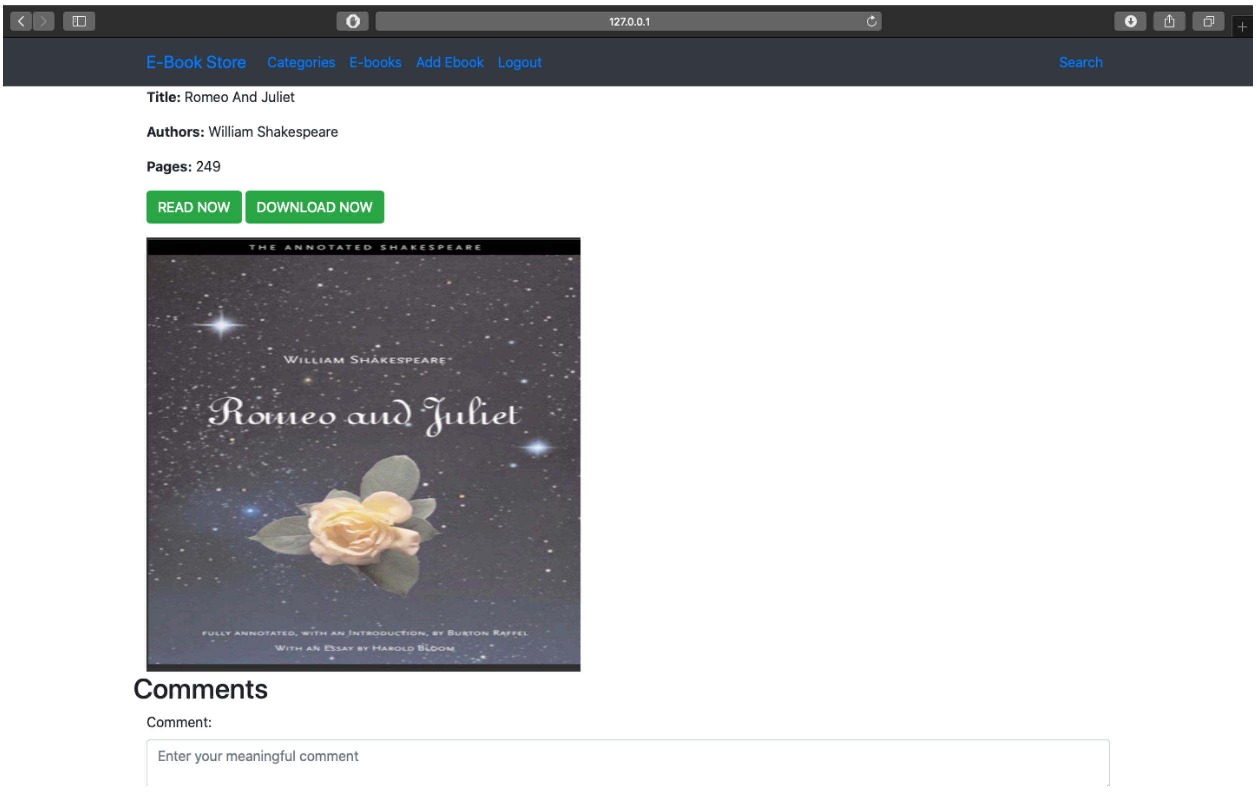
127.0.0.1

E-Book Store Categories E-books Add Ebook Logout Search

Comic

BatMan The Killing Joke
The Walking Dead
Batsman vs Superman
Avengers

Ebook page:


127.0.0.1
E-Book Store
Categories
E-books
Add Ebook
Logout
Search

Title: Romeo And Juliet

Authors: William Shakespeare

Pages: 249

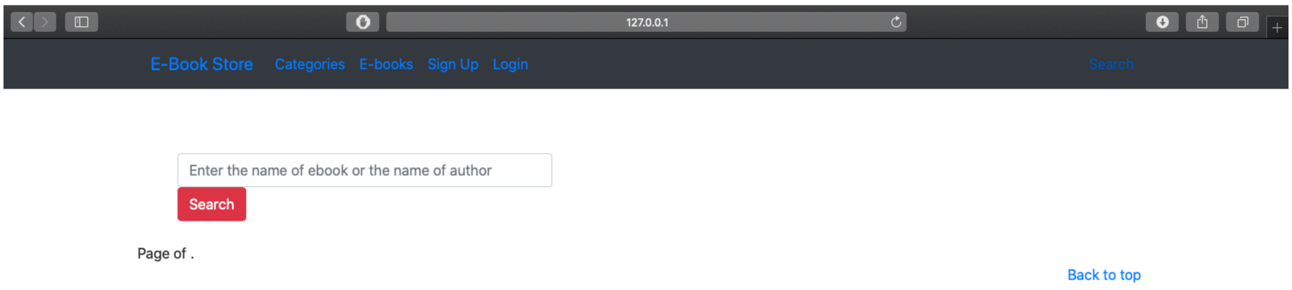
READ NOW
DOWNLOAD NOW
THE ANNOTATED SHAKESPEARE
William Shakespeare
Romeo and Juliet
Fully annotated, with an introduction, by Burton Raffel
With an essay by Harold Bloom

Comments

Comment:

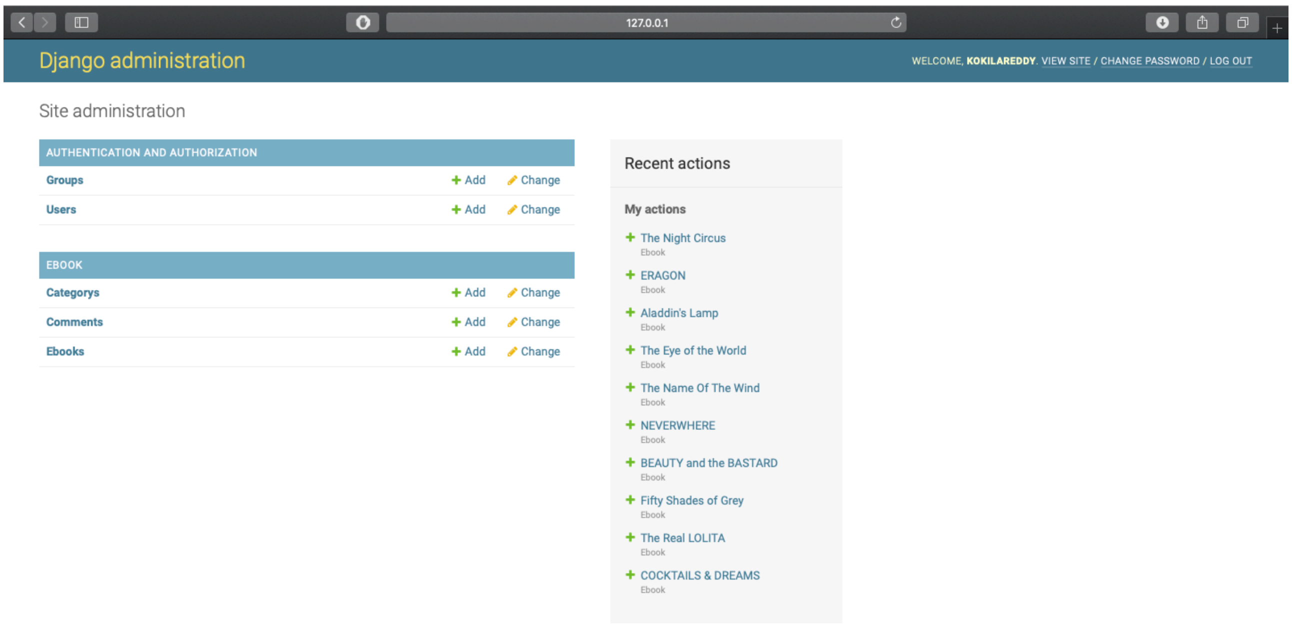
Enter your meaningful comment

Search Page:



Page of .

Admin Page:



127.0.0.1

Django administration

WELCOME, **KOKILAREDDY**. VIEW SITE / CHANGE PASSWORD / LOG OUT

Site administration

AUTHENTICATION AND AUTHORIZATION		
Groups	Add	Change
Users	Add	Change

EBOOK		
Categorys	Add	Change
Comments	Add	Change
Ebooks	Add	Change

Recent actions

My actions

- The Night Circus
  Ebook
- ERAGON
  Ebook
- Aladdin's Lamp
  Ebook
- The Eye of the World
  Ebook
- The Name Of The Wind
  Ebook
- NEVERWHERE
  Ebook
- BEAUTY and the BASTARD
  Ebook
- Fifty Shades of Grey
  Ebook
- The Real LOLITA
  Ebook
- COCKTAILS & DREAMS
  Ebook

Accounts App:

Views.py

```
from django.shortcuts import render, redirect

from django.contrib import auth

from django.contrib.auth.models import User

from django.contrib.auth.models import Permission




def signup(request):
    if request.method == 'POST':

        # sign up the user

        if request.POST['password1'] == request.POST['password2']:

            try:

                user = User.objects.get(username=request.POST['username'])

                return render(request, 'accounts/signup.html', {'error': 'Username is already taken!'})

            except User.DoesNotExist:

                user = User.objects.create_user(request.POST['username'],
password=request.POST['password1'])

                # this will mark the user as staff

                user.is_staff = True

                permissions = Permission.objects.filter(codename__in = ['add_ebook'])

                # set the permissions to user

                user.user_permissions.set(permissions)

                # save the user

                user.save()

                auth.login(request, user)

                return render(request, 'ebook/home.html', {'success': 'You are successfully registered and
logged in!'})

        else:

            return render(request, 'accounts/signup.html', {'error': 'Passwords aren\'t matched!'})

    else:

        # user wants to sign up

        return render(request, 'accounts/signup.html')
```

```
def login(request):

    if request.method == 'POST':

        # login user

        user = auth.authenticate(username=request.POST['username'], password=request.POST['pass-
word'])

        if user is not None:

            auth.login(request, user)

            return render(request, 'ebook/home.html', {'success': 'You are successfully logged in!'})

        else:

            return render(request, 'accounts/login.html', {'error': 'Username/Password doesn\'t match'})

    else:

        # user wants to login

        return render(request, 'accounts/login.html')




def logout(request):

    if request.method == 'POST':

        auth.logout(request)

    return redirect('home')
```

urls.py

```
from django.urls import path

from . import views



urlpatterns = [
    path('login', views.login, name='login'),

    path('logout', views.logout, name='logout'),

    path('signup', views.signup, name='signup'),

]
```

Ebook App

Views.py

```
from django.shortcuts import render, get_object_or_404, redirect

from .models import Category, Ebook, Comment

from django.core.paginator import Paginator

from django.contrib import auth

from django.utils import timezone

from django.db.models import Q

import operator


def home(request):

    ebooks = Ebook.objects.order_by('-id')[:8]

    return render(request, 'ebook/home.html', {

        'ebooks' : ebooks

    })


def categories(request):

    categories = Category.objects.order_by('name')
    return render(request, 'ebook/categories.html', {
                    'categories': categories,
                    })


def category(request, category_id):
    category = get_object_or_404(Category, pk = category_id)
    ebooks = category.ebook_set.all()


    paginator = Paginator(ebooks, 12)
    try:
        page = request.GET['page']
    except:
        page = 1
    ebooks = paginator.get_page(page)


    return render(request, 'ebook/category.html', {
                    'category' : category,
                    'ebooks' : ebooks
                    })


def ebooks(request):
    ebooks = Ebook.objects.all()
    paginator = Paginator(ebooks, 12)
    try:
        page = request.GET['page']
    except:
        page = 1
    ebooks = paginator.get_page(page)
    return render(request, 'ebook/ebooks.html', {
                    'ebooks': ebooks,
                    })

def ebook(request, ebook_id):
    ebook = get_object_or_404(Ebook, pk = ebook_id)
    comments = ebook.comment_set.all()
    return render(request, 'ebook/ebook.html', {
                    'ebook' : ebook,
        'comments' : comments,
                    })

def comment(request, ebook_id):
    if request.method == 'POST':
        user = auth.get_user(request)
        ebook = get_object_or_404(Ebook, pk = ebook_id)

        # create the comment
        comment = Comment()
        comment.body = request.POST['body']
        comment.pub_time = timezone.datetime.now()

        comment.ebook = ebook

        comment.user = user

        comment.save()


        return redirect('ebook', ebook_id)

    else:

        return redirect('home')


def staff(request):

    return render(request,'admin/')


def search(request):

    return render(request,'ebook/search.html')


def search_books(request):


    if request.method=='POST':

        srch=request.POST.get('srh')


        if srch:

            match=Ebook.objects.filter(Q(title=srch)|Q(authors=srch)|Q(title__icontains=srch)|Q(au-
thors__icontains=srch)|Q(title__istartswith=srch)|Q(authors__istartswith=srch))


            paginator = Paginator(match, 12)
            try:
                page = request.GET['page']
            except:
                page = 1
            match = paginator.get_page(page)


            if match:
                return render(request, 'ebook/search.html', {
                              'ebooks': match,
                              })
            else:
                return render(request,'ebook/search.html',{'error':"No result found"})
        else:
            return render(request,'ebook/search.html')


    return render(request,'ebook/search.html')
```

urls.py

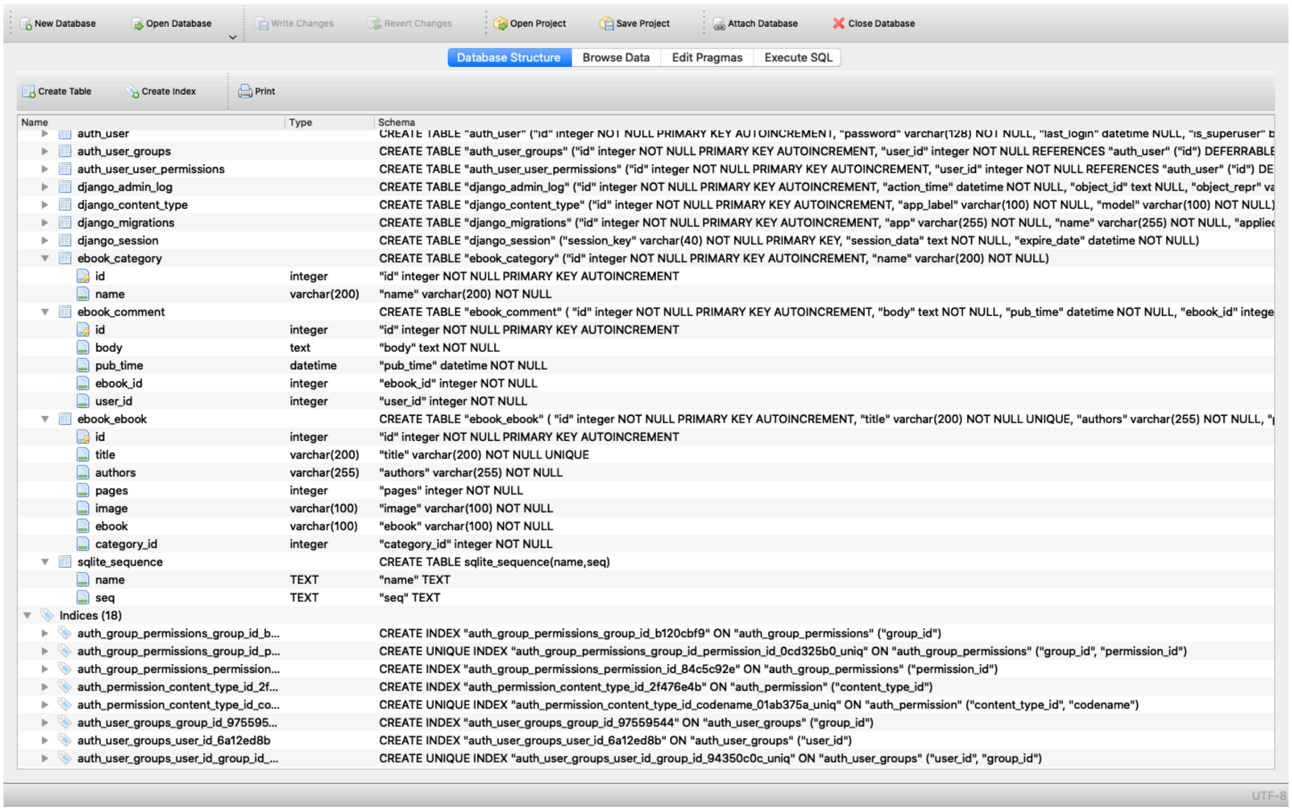
```
urlpatterns = [
    path('', views.home, name='home'),
    path('categories/', views.categories, name='categories'),
    path('category/<int:category_id>', views.category, name='category'),
    path('ebooks/', views.ebooks, name='ebooks'),
    path('ebook/<int:ebook_id>', views.ebook, name='ebook'),
    path('ebook/<int:ebook_id>/comment', views.comment, name='comment'),
    path('admin',views.staff,name='staff'),

    path('search/',views.search,name='search'),

    path('search_books/',views.search_books,name='search_books')

    ]
```

Ebookstore

Urls.py

```
from django.contrib import admin

from django.urls import path, include

from django.conf import settings

from django.conf.urls.static import static

import ebook

from ebook import urls

import accounts

from accounts import urls

urlpatterns = [

    path('admin/', admin.site.urls),

    path('', include(ebook.urls)),

    path('accounts/', include(accounts.urls))

] + static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

Database



Name	Type	Schema
auth_user		CREATE TABLE "auth_user" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "password" varchar(128) NOT NULL, "last_login" datetime NULL, "is_superuser" b
auth_user_groups		CREATE TABLE "auth_user_groups" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "user_id" integer NOT NULL REFERENCES "auth_user" ("id") DEFERRABLE
auth_user_user_permissions		CREATE TABLE "auth_user_user_permissions" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "user_id" integer NOT NULL REFERENCES "auth_user" ("id") DE
django_admin_log		CREATE TABLE "django_admin_log" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "action_time" datetime NOT NULL, "object_id" text NULL, "object_repr" va
django_content_type		CREATE TABLE "django_content_type" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "app_label" varchar(100) NOT NULL, "model" varchar(100) NOT NULL)
django_migrations		CREATE TABLE "django_migrations" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "app" varchar(255) NOT NULL, "name" varchar(255) NOT NULL, "applied
django_session		CREATE TABLE "django_session" ("session_key" varchar(40) NOT NULL PRIMARY KEY, "session_data" text NOT NULL, "expire_date" datetime NOT NULL)
ebook_category		CREATE TABLE "ebook_category" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "name" varchar(200) NOT NULL)
id	integer	"id" integer NOT NULL PRIMARY KEY AUTOINCREMENT
name	varchar(200)	"name" varchar(200) NOT NULL
ebook_comment		CREATE TABLE "ebook_comment" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "body" text NOT NULL, "pub_time" datetime NOT NULL, "ebook_id" intege
id	integer	"id" integer NOT NULL PRIMARY KEY AUTOINCREMENT
body	text	"body" text NOT NULL
pub_time	datetime	"pub_time" datetime NOT NULL
ebook_id	integer	"ebook_id" integer NOT NULL
user_id	integer	"user_id" integer NOT NULL
ebook_ebook		CREATE TABLE "ebook_ebook" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "title" varchar(200) NOT NULL UNIQUE, "authors" varchar(255) NOT NULL, "
id	integer	"id" integer NOT NULL PRIMARY KEY AUTOINCREMENT
title	varchar(200)	"title" varchar(200) NOT NULL UNIQUE
authors	varchar(255)	"authors" varchar(255) NOT NULL
pages	integer	"pages" integer NOT NULL
image	varchar(100)	"image" varchar(100) NOT NULL
ebook	varchar(100)	"ebook" varchar(100) NOT NULL
category_id	integer	"category_id" integer NOT NULL
sqlite_sequence		CREATE TABLE sqlite_sequence(name,seq)
name	TEXT	"name" TEXT
seq	TEXT	"seq" TEXT
Indices (18)		
auth_group_permissions_group_id_b...		CREATE INDEX "auth_group_permissions_group_id_b120cbf9" ON "auth_group_permissions" ("group_id")
auth_group_permissions_group_id_p...		CREATE UNIQUE INDEX "auth_group_permissions_group_id_permission_id_0cd325b0_uniq" ON "auth_group_permissions" ("group_id", "permission_id")
auth_group_permissions_permission...		CREATE INDEX "auth_group_permissions_permission_id_84c5c92e" ON "auth_group_permissions" ("permission_id")
auth_permission_content_type_id_2f...		CREATE INDEX "auth_permission_content_type_id_2f476e4b" ON "auth_permission" ("content_type_id")
auth_permission_content_type_id_co...		CREATE UNIQUE INDEX "auth_permission_content_type_id_codename_01ab375a_uniq" ON "auth_permission" ("content_type_id", "codename")
auth_user_groups_group_id_975595...		CREATE INDEX "auth_user_groups_group_id_97559544" ON "auth_user_groups" ("group_id")
auth_user_groups_user_id_6a12ed8b		CREATE INDEX "auth_user_groups_user_id_6a12ed8b" ON "auth_user_groups" ("user_id")
auth_user_groups_user_id_group_id_...		CREATE UNIQUE INDEX "auth_user_groups_user_id_group_id_94350c0c_uniq" ON "auth_user_groups" ("user_id", "group_id")

UTF-8

10. APPENDICES

10.1 Definitions, Acronyms, Abbreviations

SRS: Software requirement Specification

Non-functional attributes: A Non-Functional Requirement (NFR) defines the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other non-functional standards that are critical to success of the software system.

Use case model: A use case describes a sequence of actions a system performs that yields an observable result of value to a particular actor.

Sequence Diagram: a system sequence diagram (SSD) is a sequence diagram that shows, for a particular scenario of a use case, the events that external actors generate, their order, and possible inter-system events. System sequence diagrams are visual summaries of the individual use cases.

10.2References:

https://readthedocs.org/projects/r2e2-docs/downloads/pdf/latest/

https://courses.cs.ut.ee/MTAT.03.306/2018_fall/uploads/Main/team5.pdf