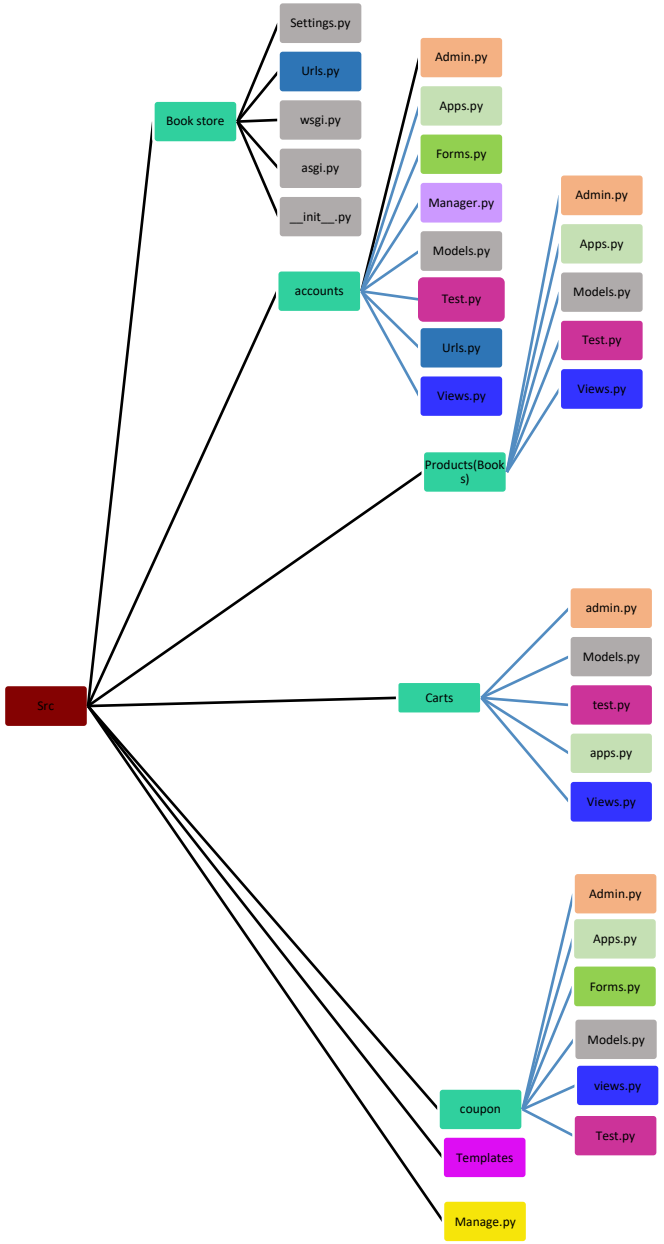
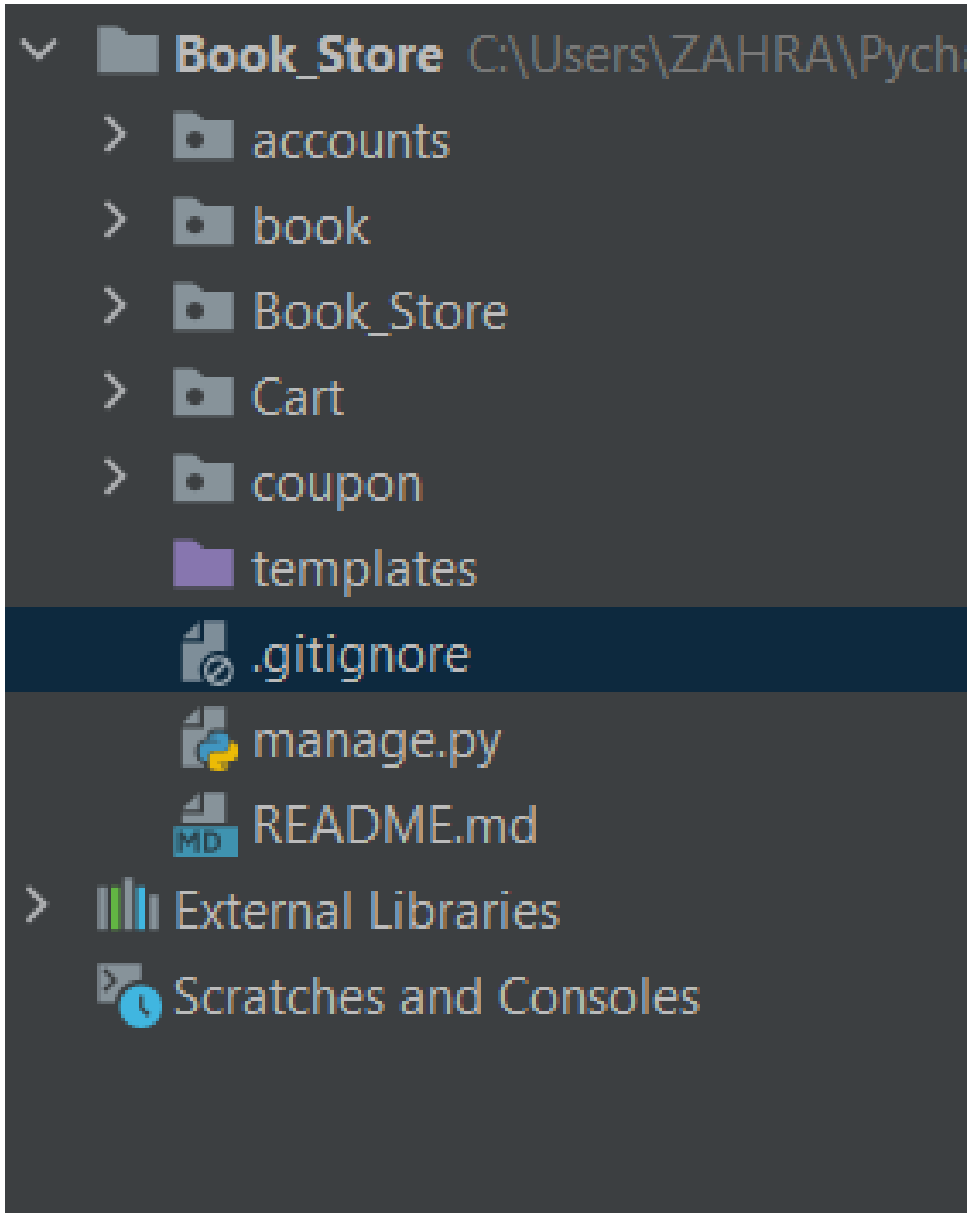

Src
Book store
Settings.py
Urls.py
wsgi.py
asgi.py
__init__.py
accounts
Admin.py
Apps.py
Forms.py
Manager.py
Models.py
Test.py
Urls.py
Views.py
Products(Books)
Admin.py
Apps.py
Models.py
Test.py
Views.py
Carts
admin.py
Models.py
test.py
apps.py
Views.py
coupon
Admin.py
Apps.py
Forms.py
Models.py
views.py
Test.py
Templates
Manage.py



نمای کلی پروژه

```python
# Create your models here.
from accounts.managers import MyUserManager


class MyUser(AbstractUser):
    username = None
    full_name = models.CharField(max_length=50, unique=True)
    date_of_birth = models.DateField(null=True, blank=True)
    email = models.EmailField(max_length=100, unique=True)
    USERNAME_FIELD = 'email'
    REQUIRED_FIELDS = []
    objects = MyUserManager()
```

ایجاد یوزر های مختلف: سوپر یوزر و کارمند و مشتری

از کلاس یوزر پیش فرض جنگو استفاده نشده و یک یوزر جدید ساخته شده و در فایل تنظیمات ثبت شده.

لاگین توسط ایمیل انجام می شود.
برای این کار نیاز به منیجر داریم تا ایمیل را در جای نام کاربری قرار دهد.

```python
class MyUserManager(BaseUserManager):
    def create_user(self, email, password):
        if not email:
            raise ValueError('users must have Email')

        user = self.model(email=self.normalize_email(email))
        user.set_password(password)
        user.save(using=self._db)
        return user

    def create_superuser(self, email, password):
        user = self.create_user(email, password)
        user.is_admin = True
        user.save(using=self._db)
        return user
```

منیجر اپ اکانت
در این منیجر دو حالت یوزر عادی و سوپر یوزر در نظر گرفته شده است.
آرگومان ورودی این توابع ایمیل و پسوورد است که با استفاده از انها یک یوزر می سازد.

```python
# Register your models here.
from accounts.models import MyUser


class UserAdmin(BaseUserAdmin):
    list_display = ['id', 'date_of_birth', 'email', 'is_staff', 'is_active']
    model = MyUser
    ordering = ('id',)


admin.site.register(MyUser, UserAdmin)
```

فایل ادمین اپ اکانت
لیست ستون های قابل نمایش و سپس نام مدلی ک مربوط به این ادمین است و اتریبیوتی برای ترتیب نمایش سطر آبجکت ها نوشته شده است.

```python
class UserChangeForm(forms.ModelForm):
    password = ReadOnlyPasswordHashField()

    class Meta:
        model = MyUser
        fields = ('email', 'password', 'full_name')

    def clean_password(self):
        return self.initial['password']
```

فایل فرم اپ اکانت
یک فرم برای تغییر پسوورد طراحی می گردد.

```python
class UserLoginForm(forms.Form):
    email = forms.EmailField(widget=forms.EmailInput(attrs={'class': 'form-control'}))
    password = forms.CharField(widget=forms.PasswordInput(attrs={'class': 'form-control'}))


class UserRegistrationForm(forms.Form):
    email = forms.EmailField(widget=forms.EmailInput(attrs={'class': 'form-control'}))
    full_name = forms.CharField(widget=forms.TextInput(attrs={'class': 'form-control'}))
    password = forms.CharField(widget=forms.PasswordInput(attrs={'class': 'form-control'}))

```

فایل فرم اپ اکانت
یک فرم برای لاگین افراد که فقط ایمیل و پسورد گرفته می شود.
یک فرم برای ساخت اکانت طراحی می شود.

```python
class Category(models.Model):
    class Meta:
        verbose_name_plural = 'categories'

    name = models.CharField(max_length=40, unique=True)
    slug = models.CharField()


class Book(models.Model):
    title = models.CharField(max_length=200, unique=True)
    author = models.CharField(max_length=200)
    price = models.DecimalField(max_digits=6, decimal_places=2)
    stock = models.IntegerField(default=0)
    created = models.DateTimeField(auto_now_add=True)
    description = models.CharField(max_length=1000)
    slug = models.SlugField()
    category = models.ManyToManyField(Category, on_delete=models.DO_NOTHING)

    def __str__(self):
        return self.title
```

اپ کتگوری
مدل کتگوری برای دسته بندی کتاب ها تعریف شده است که یک فیلد slugبرای ساخت url زیباتر دارد.

مدل کتاب که مشخصات محصول در ان تعریف می شود و با مدل کتگوری رابطه m2m دارد.

```python
# Register your models here.
from book.models import Book

admin.site.register(Book)
```

فایل ادمین اپ اکانت
رجیستر مدل های اپ کتاب در این جا انجام
می شود.

```python
class Cart(models.Model):
    user = models.ForeignKey(settings.AUTH_USER_MODEL, on_delete=models.CASCADE)
    STATUS_CHOICES = [('D', 'delete'), ('W', 'waiting'), ('O', 'ordered')]
    order_status = models.IntegerField(max_length=1, choices=STATUS_CHOICES)
    created = models.DateTimeField(auto_now_add=True)
    updated = models.DateTimeField(auto_now=True)
    paid = models.BooleanField(default=False)
    discount = models.IntegerField(blank=True, null=True, default=None)

    class Meta:
        ordering = ('-created',)

    def __str__(self):
        return f'{self.user} - {str(self.id)}'

    def get_total_price(self):
        pass
```

اپ سبد خرید
این مدل هنوز بسیار خام است و نیاز به تغییر دارد.

این مدل شامل دو کلاس است
یکی سبد خرید افراد و دیگری شامل تمام سفارش های سبد خرید هر فرد می باشد.

```python
class CartItems(models.Model):
    order = models.ForeignKey(Cart, on_delete=models.CASCADE)
    product = models.ForeignKey(Book, on_delete=models.CASCADE)
    # price = models.IntegerField()
    quantity = models.PositiveSmallIntegerField(default=1)

    def __str__(self):
        return str(self.id)

    def total(self):
        return self.quantity * self.product.price

    def name(self):
        return self.product.title
```

```python
from Cart.models import Cart, CartItems


admin.site.register(Cart)
admin.site.register(CartItems)
```

فایل ادمین اپ cart
رجیستر مدل های اپ سبد خرید در این جا انجام می شود.

```python
class CashCoupon(models.Model):
    code = models.CharField(max_length=30, unique=True)
    valid_from = models.DateTimeField()
    valid_to = models.DateTimeField()
    discount = models.IntegerField(validators=[MinValueValidator(0), MaxValueValidator(100)])
    active = models.BooleanField(default=False)

    def __str__(self):
        return self.code
```

اپ تخفیف ها
شامل 3 کلاس با تفاوت نوع تخفیف :
اول کلاسی برای محاسبه ی تخفیف نقدی برای هر کتاب

```python
class PercentCoupon(models.Model):
    code = models.CharField(max_length=30, unique=True)
    valid_from = models.DateTimeField()
    valid_to = models.DateTimeField()
    discount = models.IntegerField(max_length=2)
    active = models.BooleanField(default=False)

    def __str__(self):
        return self.code


class PercentCouponForCart(models.Model):
    code = models.CharField(max_length=30, unique=True)
    valid_from = models.DateTimeField()
    valid_to = models.DateTimeField()
    discount = models.IntegerField(max_length=2)
    active = models.BooleanField(default=False)
```

2- کلاسی برای محاسبه ی تخفیف درصدی برای هر کتاب

3-کلاسی برای محاسبه ی تخفیف نقدی برای یک سبد خرید

```python
class CouponApplyForm(forms.Form):
    code = forms.CharField()
    discount = forms.IntegerField()
```

اپ تخفیف ها
فایل فرم
فرمی برای ایجاد یک کد تخفیف

```python
# Create your views here.
def search_bar(request):
    context = {}
    return render(request, 'page.html', context)
```

برای نوار جستجو یک ویو در اپ کتاب در نظر گرفته شده (ویو ناقص است)