```
from email.policy import default
from django.contrib import admin
from django.db import models
from django.db.models.base import Model
from django.db.models.deletion import CASCADE
from django.db.models.fields import CharField, DateTimeField, EmailField,
IntegerField, TextField
from django.db.models.fields.related import ForeignKey,
ManyToManyField,OneToOneField
from django.forms import BooleanField, DecimalField, URLField

import uuid





# Create your models here.

class Transaction(models.Model):

    # Choices for fields

    # Transaction Status Choices
    transaction_status_choices= (
        ('cancelled','Cancelled'),
        ('not_paid','Not Paid'),
        ('Paid','Paid'),
        ('pending_refund','Pending Refund'),
        ('refunded','Refunded'),
        ('restocked','Restocked'),
    )

    # Transaction Choices
    transaction_choices = (
        ('proposal','Proposal'),
        ('pending','Pending'),
        ('invoice','Invoice'),
        ('charge','Charge'),
        ('refund','Refund'),
    )

    # Transaction ID

    # Transaction Type
    type = CharField(max_length=200,choices=transaction_choices,blank=True)

    # Transaction Date Time

    # Transaction Provider
# Transaction Link
# Use case link to custom accounting software or stripe Invoice

# Payment Status

# Lined Items including:
# item description
# unit price
# quantity
# line total

# Subtotal
# Additional charges
# Tax
# Total

# Stripe

# Primary Contact -> Selling party / primary party
# Secondary Contact -> Purchasing party

# METHODS
# Stripe
#
#  Charge
#   create
#   retrieve
#   update
#   capture
#   List all charges
#
#  Customer
#   create
#   retrieve
#   update
#   delete
#   List all customers
#
#
#
#
#
#
#
#
#
#
#
#
```

```
#
#
#  Balance
#  Balance Transactions
#  Disputes
#  Events
#  Files
#  File Links
#  Mandates
#  Payment Intents
#  Payouts
#  Refunds
#  Payment Methods
#  Bank Accounts
#  Cards
#  Sources
#  Products
#  Price
#  Coupons
#  Promotion Codes
#  Discounts
#  Tax Codes
#  Tax Rates
#  Shipping Rates
#  Credit Notes
#  Customer Portal
#  Customer Tax IDs
#  Invoices
#  Invoice Items
#  Plans
#  Quotes
#  Subscriptions
#  Subscription Items
#  Subscription Schedule
#  Records
#  Fraud
#  Reports
#
#
#
#




pass
```

```
class Sale(models.Model):

    # Attributes

    # Invoice ID
    # Url to Invoice
    invoice_url = URLField()
    # Has invoice been paid?
    invoice_status =
CharField(max_length=200,choices=sale_status_choices,default='not_paid')
    # Buyers premium
    buyers_premium = DecimalField()


    # RELATED FIELDS
    # Auction Event ID for Invoice
    auction =
OneToOneField('Auction',related_name="auction_event_id",on_delete=models.CASCADE)
    # Customer
    customer =
ForeignKey('Contact',related_name="customer_name",on_delete=models.CASCADE)
    # Seller
    seller =
ForeignKey('Contact',related_name="seller_name",on_delete=models.CASCADE)



    # Methods
    # subtotal - Add sales prices of items and buyers premium before sales tax
    # total - total after text

    pass

# class SaleAdmin(admin.ModelAdmin):
#     pass

class Refund(models.Model):

    # Attributes

    # Refund ID
    refund_id = CharField(max_length=200,blank=True)
    # Contact
    contact = ForeignKey('Contact',on_delete=models.CASCADE)
    # Items for Refund
    # Date of Sale

    # Methods

    # Amount - GET STEP BY STEP PROCEDURE BY CHARLES Add up amounts for items
    # Days Since Sale - MATH BETWEEN CURRENT DAY AND DAY OF PURCHASE
    pass

# class RefundAdmin(admin.ModelAdmin):
#     pass


class Auction(models.Model):
    event = CharField(max_length=10)
    start_date_time = DateTimeField()
    end_date_time = DateTimeField()

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