

ISE 2 DBMS PROJECT**Topic : Sales And Inventory Management System**

Site is hosted on following web address :

<https://sales-inventory-management.herokuapp.com/>

Code for the site is available on following link :

<https://github.com/kamatyogesh7/sales-and-inventory-management-system>

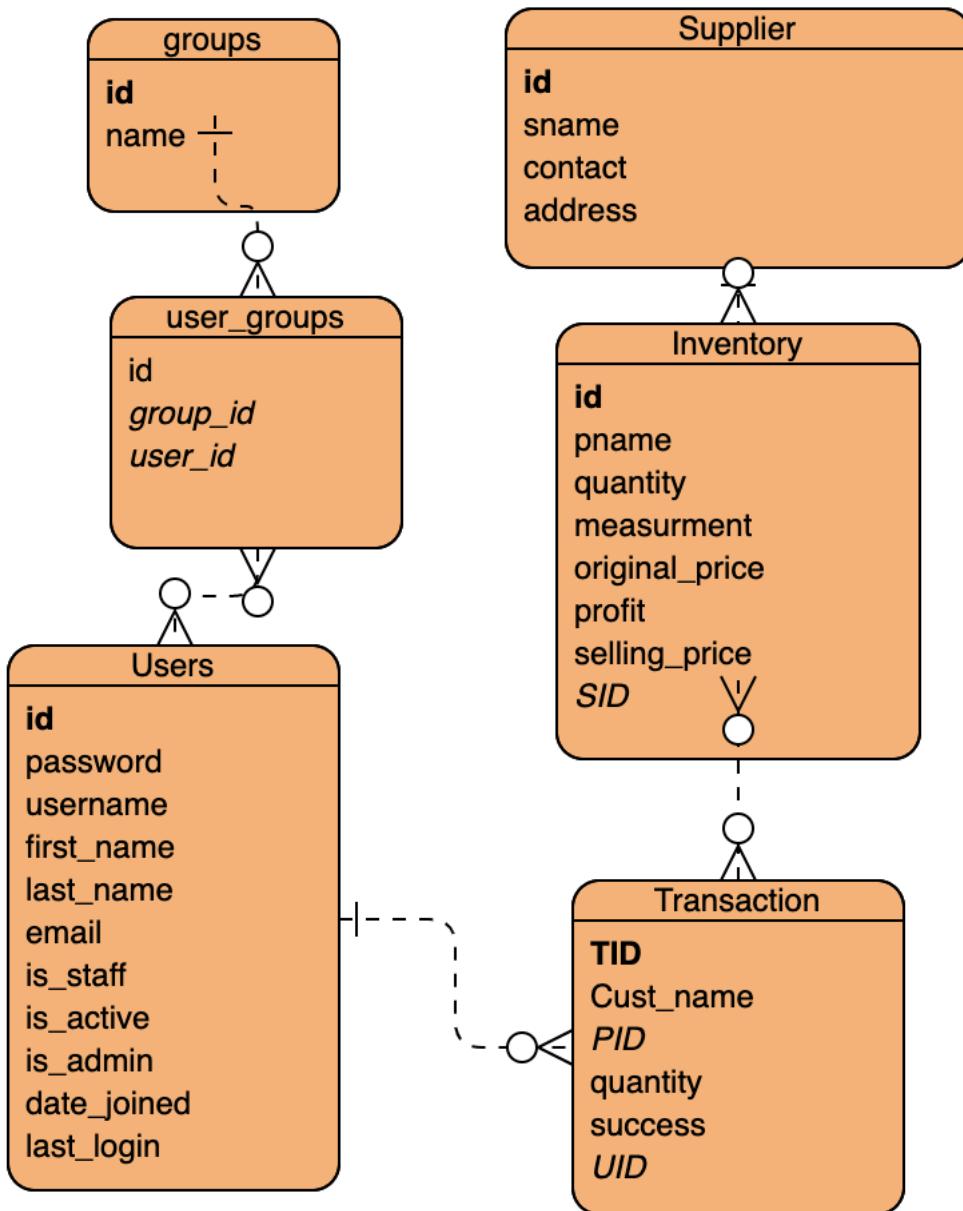
PURPOSE

PURPOSE Inventory is the amount of tangible goods, products or services you offer your customers. These goods are itemized and catalogued into an inventory management system.

E-commerce businesses most likely deal with the latter, or finished goods that are currently in their possession. Inventory is a portion of a company's assets that are either ready to order or will be ready for sale soon. Product inventory is a tangible asset because it's a real-time evaluation of the revenue a company is generating. If an e-commerce store can't move product from its inventory, it is clearly struggling to make sales and must adjust its operations, otherwise it won't obtain bottom-line revenue.

Inventory is often the largest item a business has in its current assets, meaning it must be accurately monitored. Inventory is counted and valued at the end of each accounting period to determine the company's profits or losses. In the e-commerce sector, a day-to-day management of inventory helps increase business intelligence and visibility. E-commerce business owners have better ongoing control when they have constant eyes on inventory. And that's why the need of "Inventory Management System".

E-R diagram



OUTPUT SCREENSHOTS WITH EXPLANATION

Login Page : (required without which we can not go to home page)

The screenshot shows a web browser window for the SIMS application at <https://sales-inventory-management.herokuapp.com>. The title bar says "SIMS". The navigation menu includes "Suppliers", "Inventory", "Checkout", "Transaction", "Generate Report", "Charts", and "LOGOUT". A "DaDT" icon is in the top right. The main content is titled "Supplier Details" and displays a table of supplier information:

#	Supplier Name	Contact	Address
1	Smart Suppliers	7021426622	dadar west mumbai 400028
2	MLP Distributors	7021426623	dadar west mumbai 400027
3	LK Supplier	7021426621	dadar west mumbai 400021
4	MN suppliers	7021426620	dadar west mumbai 400022

Home Page : (Shows supplier details)

Home Page contains following tabs as well :

1. Inventory (shows inventory details)
2. Checkout page for sales.
3. Transaction page for successful transactions.
4. Report generating page
5. Graph analytics
6. Debug tool to show raw sql queries

1.

The screenshot shows a web browser window for the SIMS application at <https://sales-inventory-management.herokuapp.com/inventory/>. The title bar says "SIMS". The navigation menu includes "Suppliers", "Inventory", "Checkout", "Transaction", "Generate Report", "Charts", and "LOGOUT". A "DaDT" icon is in the top right. The main content is titled "Inventory Details" and displays the following information:

Total Inventory Value : 7485

Total Products : 5

#	Product Name	Quantity	Measurement	Original Price	Profit	Selling Price	Supplier
1	Pen	35	Piece	25	10	35	SSmart Suppliers
2	Sugar	50	Kilogram	52	8	60	SMLP Distributors
3	Eraser	50	Piece	5	2	7	SSmart Suppliers
4	Pencil	47	Piece	7	3	10	SLK Supplier
5	Bread	61	Piece	32	8	40	SMLP Distributors

2.

Add to cart

#	Customer Name	Product
1	ramesh	2

CHECKOUT

	Bread	Piece	Kilogram
Quantity Available	50	61	40
Quantity Required	10	0	0

Selling Price Supplier

- SSmart Suppliers +
- SMLP Distributors +
- SSmart Suppliers +
- SLK Supplier +
- SMLP Distributors +

2.

Checkout

Cart				
#	Customer Name	Product	Quantity	Price
2	suresh	2	10	600
1	ramesh	2	10	600

CHECKOUT

#	Product Name	Quantity	Measurement	Selling Price	Supplier
1	Pen	35	Piece	35	SSmart Suppliers +
2	Sugar	50	Kilogram	60	SMLP Distributors +
3	Eraser	50	Piece	7	SSmart Suppliers +
4	Pencil	47	Piece	10	SLK Supplier +
5	Bread	61	Piece	40	SMLP Distributors +

3.

The screenshot shows a web browser window titled "SIMS" with the URL <https://sales-inventory-management.herokuapp.com/transaction/>. The page has a blue header bar with the SIMS logo and navigation links: Suppliers, Inventory, Checkout, Transaction, Generate Report, Charts, and LOGOUT. A watermark for "DRAFT" is visible in the top right corner. The main content area is titled "Transaction Details" and contains a table with the following data:

Customer Name	Quantity	Measurment	Price	Supplier	Action
suresh	10	Kilogram	600	MLP Distributors	DELETE
ramesh	10	Kilogram	600	MLP Distributors	DELETE

4.

The screenshot shows a web browser window titled "SIMS" with the URL <https://sales-inventory-management.herokuapp.com/report/>. The page has a blue header bar with the SIMS logo and navigation links: Suppliers, Inventory, Checkout, Transaction, Generate Report, Charts, and LOGOUT. A watermark for "DRAFT" is visible in the top right corner. The main content area is titled "Report" and contains two blue buttons:

[INVENTORY REPORT](#) [TRANSACTION REPORT](#)

SIMS Reports

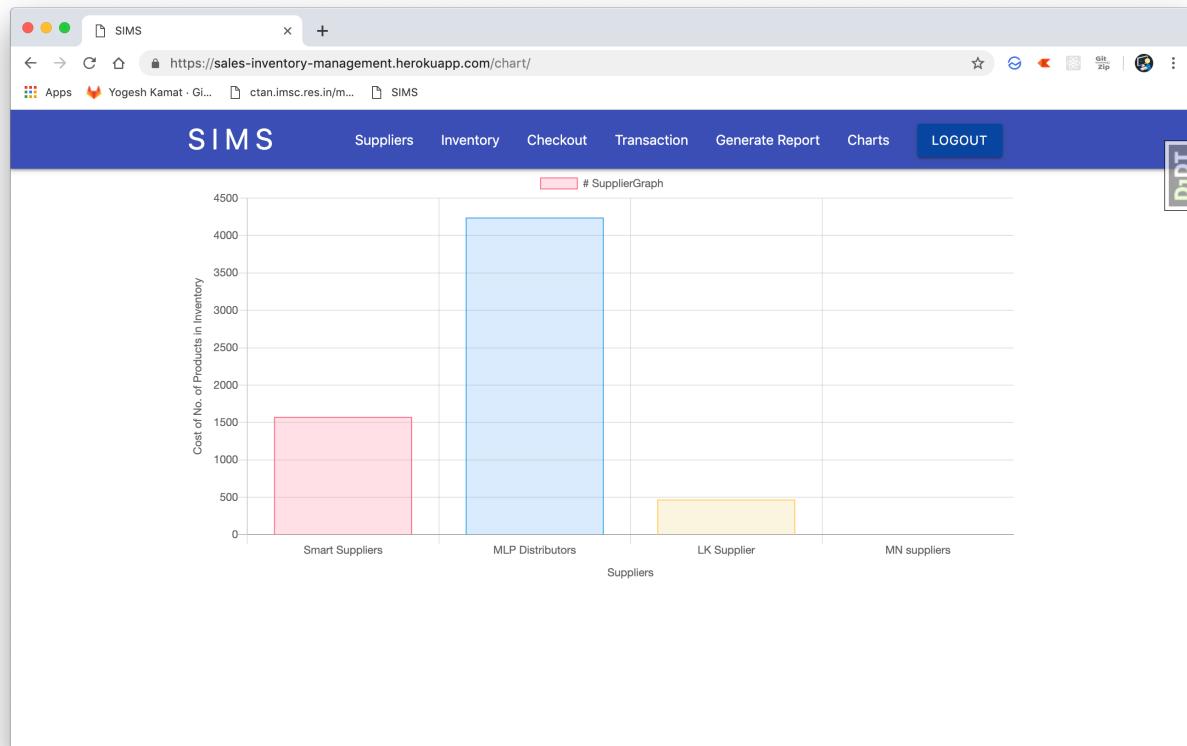
Username : yogesh

Inventory Details

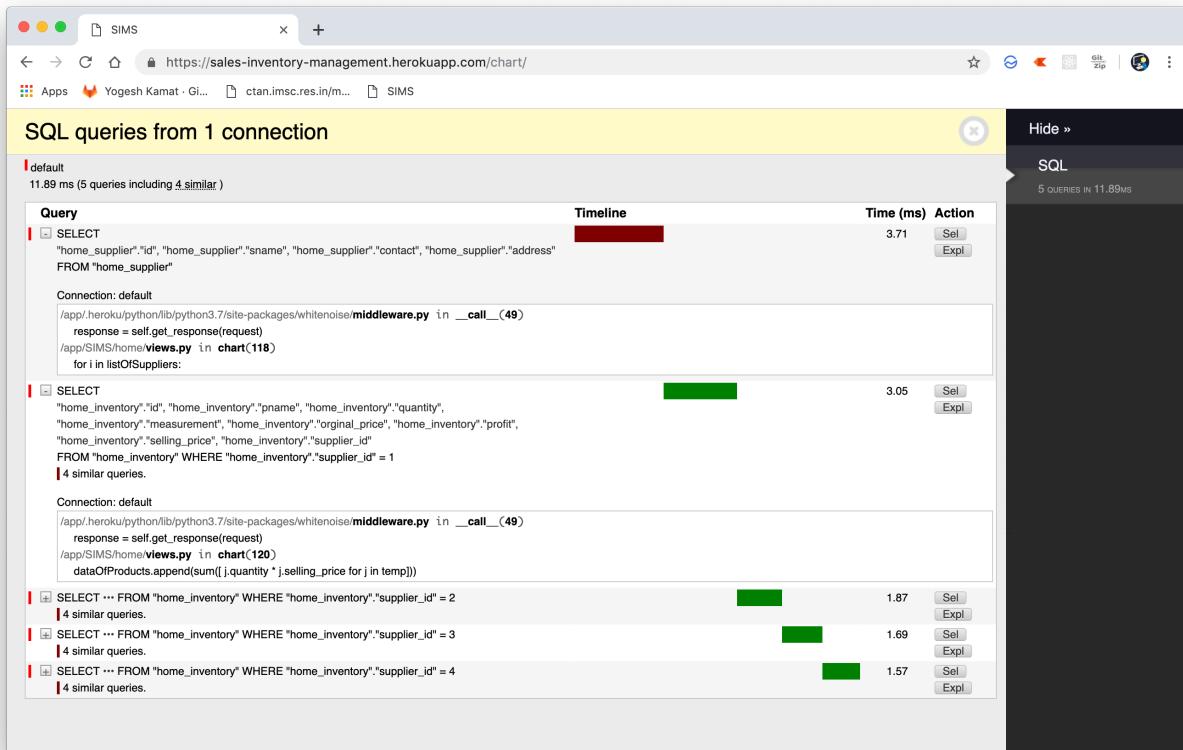
14th of April

Product ID	Product Name	Quantity	Measurement	Original Price	Profit	Selling Price	Supplier
1	Pen	35	Piece	25	10	35	Smart Suppliers
3	Eraser	50	Piece	5	2	7	Smart Suppliers
4	Pencil	47	Piece	7	3	10	LK Supplier
5	Bread	61	Piece	32	8	40	MLP Distributors
2	Sugar	30	Kilogram	52	8	60	MLP Distributors

5.



6.



**Admin Panel to add or delete inventories or suppliers or transactions.
In Admin Panel Only admin user can login.**

HomePage for admin panel :

SIMS Admin Panel

Site administration

AUTHENTICATION AND AUTHORIZATION

- Groups [+ Add](#) [Change](#)
- Users [+ Add](#) [Change](#)

HOME

- Inventorys [+ Add](#) [Change](#)
- Suppliers [+ Add](#) [Change](#)
- Transactions [+ Add](#) [Change](#)

Recent actions

My actions

- ✗ Iphone Inventory
- + Iphone Inventory
- + sampleuser User
- + sejal User

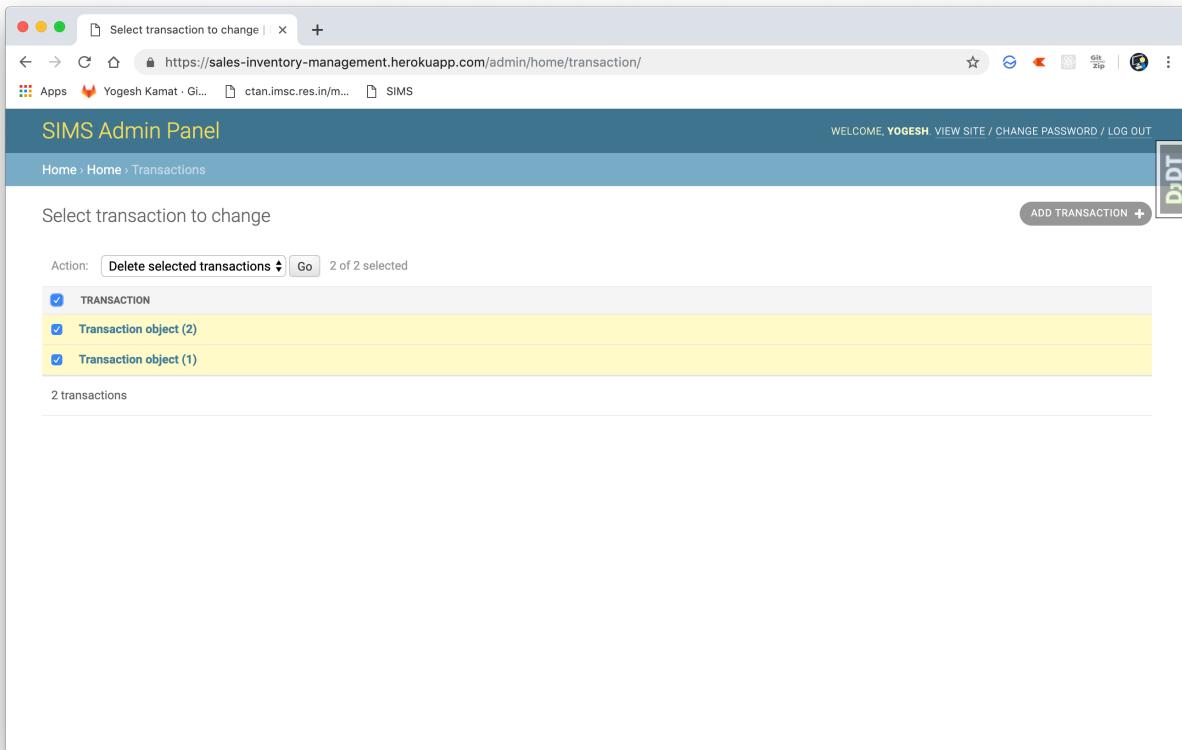
Add Inventory form :

The screenshot shows a web browser window titled 'Add inventory | Django site admin'. The URL is <https://sales-inventory-management.herokuapp.com/admin/home/inventory/add/>. The page is part of the 'SIMS Admin Panel' and is titled 'Add inventory'. It contains fields for Pname, Quantity, Measurement, Orginal price, Profit, Selling price, and Supplier. At the bottom right are buttons for 'Save and add another', 'Save and continue editing', and a large blue 'SAVE' button.

Supplier add form :

The screenshot shows a web browser window titled 'Add supplier | Django site admin'. The URL is <https://sales-inventory-management.herokuapp.com/admin/home/supplier/add/>. The page is part of the 'SIMS Admin Panel' and is titled 'Add supplier'. It contains fields for Sname, Contact, and Address. At the bottom right are buttons for 'Save and add another', 'Save and continue editing', and a large blue 'SAVE' button.

Delete Transaction page (In similar manner every inventory and supplier object can be deleted as well)



TECHNOLOGIES USED :

1. PostgreSQL
2. Django web framework
3. Html and CSS
4. Heroku web hosting

CODE

(Complete code is available on GitHub link provided above.)

Home App :

Model :

```
from django.db import models
from django.contrib.auth.models import User
# Create your models here.
```

```
class Supplier(models.Model):
```

```
sname          = models.CharField(max_length=255, unique=True)
contact       = models.CharField(max_length=255)
address       = models.TextField(max_length=255)
```

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

```
class Inventory(models.Model):
    pname          = models.CharField(max_length=255)
    quantity       = models.PositiveIntegerField()
    measurement   = models.CharField(max_length=255)
    orginal_price = models.PositiveIntegerField()
    profit         = models.PositiveIntegerField()
    selling_price = models.PositiveIntegerField()
    supplier       = models.ForeignKey(Supplier, on_delete=models.CASCADE)
```

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

```
class Transaction(models.Model):
    cust_name      = models.CharField(max_length=255)
    pid            = models.ForeignKey(Inventory, on_delete=models.CASCADE)
    quantity_r     = models.PositiveIntegerField()
    success        = models.BooleanField(default=False)
    uid            = models.ForeignKey(User,
on_delete=models.CASCADE)

    def actual_price(self):
        return int(self.quantity_r) * int(self.pid.selling_price)
```

Url for Routing :

```
from django.urls import path
from home import views
urlpatterns = [
    path('',views.index,name='home'),
    path('inventory/',views.inventory,name='inventory'),
    path('checkout/',views.checkout,name='checkout'),
    path('transaction/',views.transaction,name='transaction'),
    path('report/',views.report,name='report'),
    path('chart/',views.chart,name='chart'),
```

]

Views :

```
from django.shortcuts import render, HttpResponseRedirect
from django.contrib.auth.decorators import login_required
from django.contrib.auth.models import User
from django.conf import settings
from easy_pdf import rendering
from django.http import JsonResponse
```

Create your views here.

from home.models import Supplier, Inventory, Transaction

```
@login_required
def index(request):
    suppliers      = Supplier.objects.all()
    context        = {
        'suppliers':suppliers,
    }
    return render(request, 'home/index.html', context)

@login_required
def inventory(request):
    inventory      = Inventory.objects.all()
    total_products = Inventory.objects.all().count()
    total_value    = [ i.quantity * i.selling_price for i in inventory]
    context        = {
        'inventory':inventory,
        'total_products':total_products,
        'total_value': sum(total_value)
    }
    return render(request, 'home/index.html', context)

@login_required
def checkout(request, *args, **kwargs):
    user          = request.user
    inventory     = Inventory.objects.all()
    cart          = Transaction.objects.all().filter(uid_id = user.id,success=0)
    context        = {
        'checkout':1,
        'inventory_checkout':inventory,
        'cart':cart,
    }

if request.method == 'POST' and request.POST.get('tiddel'):
    tid = int(request.POST.get('tiddel'))
    Transaction.objects.get(pk=tid).delete()
    cart      = Transaction.objects.all().filter(uid_id = user.id,success=0)
    context['cart'] = cart
    return render(request,'home/index.html', context)

if request.method == 'POST' and request.POST.get('checkoutrequest'):
    tobj       = Transaction.objects.all().filter(uid_id=user.id,success=False)
    for i in tobj:
        iobj = Inventory.objects.get(pk=i.pid_id)
        print(type(iobj.quantity),iobj.quantity)
        if iobj.quantity > 0 and (iobj.quantity - i.quantity_r) > 0:
            i.success = True
            i.save()
            iobj.quantity = iobj.quantity - i.quantity_r
            iobj.save()

    cart      = Transaction.objects.all().filter(uid_id = user.id,success=False)
```

```

context['cart'] = cart
return render(request,'home/index.html', context)

if request.method == 'POST':
    pid      = request.POST['pid']
    print("PID : ",pid)
    iobj     = Inventory.objects.get(pk=pid)
    cname   = request.POST['cname']
    qreq    = request.POST['qreq']
    if int(qreq) <= iobj.quantity:
        tobj   =
        Transaction.objects.create(cust_name=cname,pid_id=iobj.id,uid_id=user.id,quantity_r=qre
q)
        tobj.save()

    return render(request,'home/index.html', context)

```

```

@login_required
def transaction(request):
    user = request.user
    tobj = Transaction.objects.all().filter(uid_id = user.id, success=True)
    context = {
        'transactions':tobj,
    }

    if request.method == 'POST' and request.POST.get('tiddel'):
        tid = int(request.POST.get('tiddel'))
        Transaction.objects.get(pk=tid).delete()
        tobj      = Transaction.objects.all().filter(uid_id = user.id, success=True)
        context['transactions'] = tobj
        return render(request,'home/index.html', context)

    return render(request,'home/index.html', context)

```

```

@login_required
def report(request):
    user = request.user
    inventory = Inventory.objects.all()
    tobj = Transaction.objects.all().filter(uid_id = user.id, success=True)
    context = {
        'report':1,
        'inventory_report':inventory,
        'transaction_report':tobj,
        'user':request.user
    }
    if request.method == 'POST' and request.POST.get('ireport'):
        return rendering.render_to_pdf_response(request, 'home/ipdf.html', context,
using=None, download_filename=None, content_type='application/pdf',
response_class=HttpResponse)

    if request.method == 'POST' and request.POST.get('treport'):

```

```
    return rendering.render_to_pdf_response(request, 'home/tpdf.html', context,
using=None, download_filename=None, content_type='application/pdf',
response_class=HttpResponse)
    return render(request,'home/index.html', context)
```

```
def chart(request):
    listOfSuppliers = Supplier.objects.all()
    dataOfProducts = []
    for i in listOfSuppliers:
        temp = Inventory.objects.all().filter(supplier_id = i.id)
        dataOfProducts.append(sum([ j.quantity * j.selling_price for j in temp]))
    print(dataOfProducts)
    listOfSuppliers = [i.sname for i in listOfSuppliers]
    context = {
        'listOfSuppliers':listOfSuppliers,
        'dataOfProducts':dataOfProducts,
    }
    return render(request, 'home/chart.html',context)
```

Users App :**Views :**

```
from django.shortcuts import render, redirect, reverse
from django.contrib.auth import authenticate, login, logout
# Create your views here.
from users.forms import LoginForm
```

```
def index(request):
    if request.method == 'POST':
        loginform = LoginForm(request.POST)
        if loginform.is_valid():
            username = loginform.cleaned_data['username']
            password = loginform.cleaned_data['password']
            user = authenticate(username = username, password = password)
            if user is not None:
                login(request, user)
                return redirect(reverse('home'))
            else:
                context = {
                    'form':loginform,
                    'error':'Could not login, Please try again...',
                }
                return render(request, 'users/index.html', context)
    loginform = LoginForm()
    context = {
        'form' : loginform,
    }
    return render(request,'users/index.html', context)
```

```
def logout_user(request):
    logout(request)
```

```
return redirect(reverse('login'))
```

Url for routing :

```
from django.urls import path  
from users import views
```

```
urlpatterns = [  
    path('login/',views.index,name='login'),  
    path('logout/', views.logout_user, name = "logout"),  
]
```

CONCLUSION :

Hence, we successfully created sales and inventory management system using various things learned during the entire dbms course such as DDL, DML, DCL commands as well as View, Triggers etc. Our system can be used by Inventory manager to add or delete inventory items as well as supplier details.

And it can also be used by Sales persons to do various transactions.

Through chart section we can view graph analytics.

Name : Yogesh D. Kamat

SPIT Batch E

UID : 2018240066