Name : Yogesh D. Kamat **SPIT Batch E** **UID : 2018240066**

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

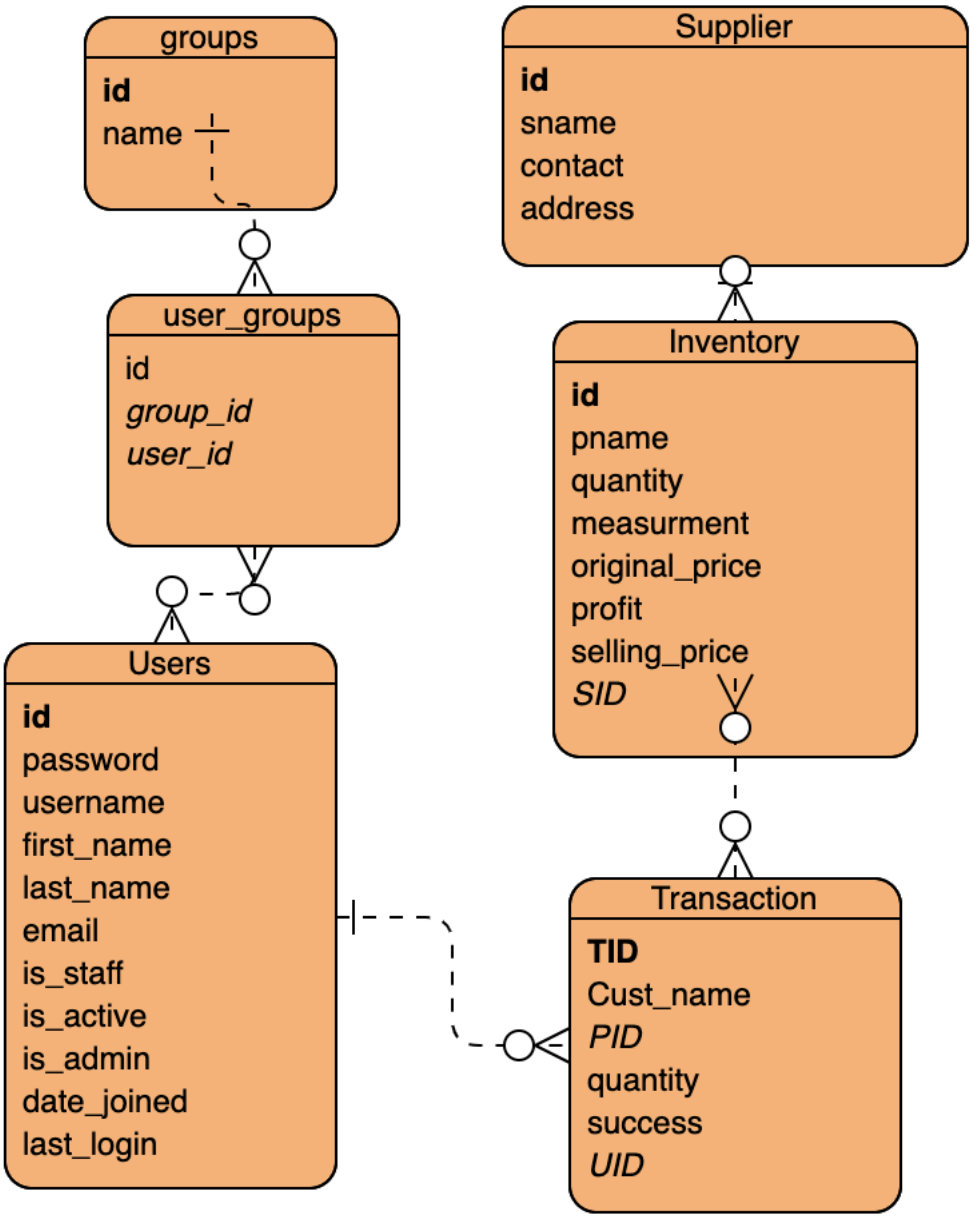
PURPOSE Inventory is the amount of tangible goods, products or services you offer your customers. These goods are itemized and catalogued into an inventory management system.

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 it's 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

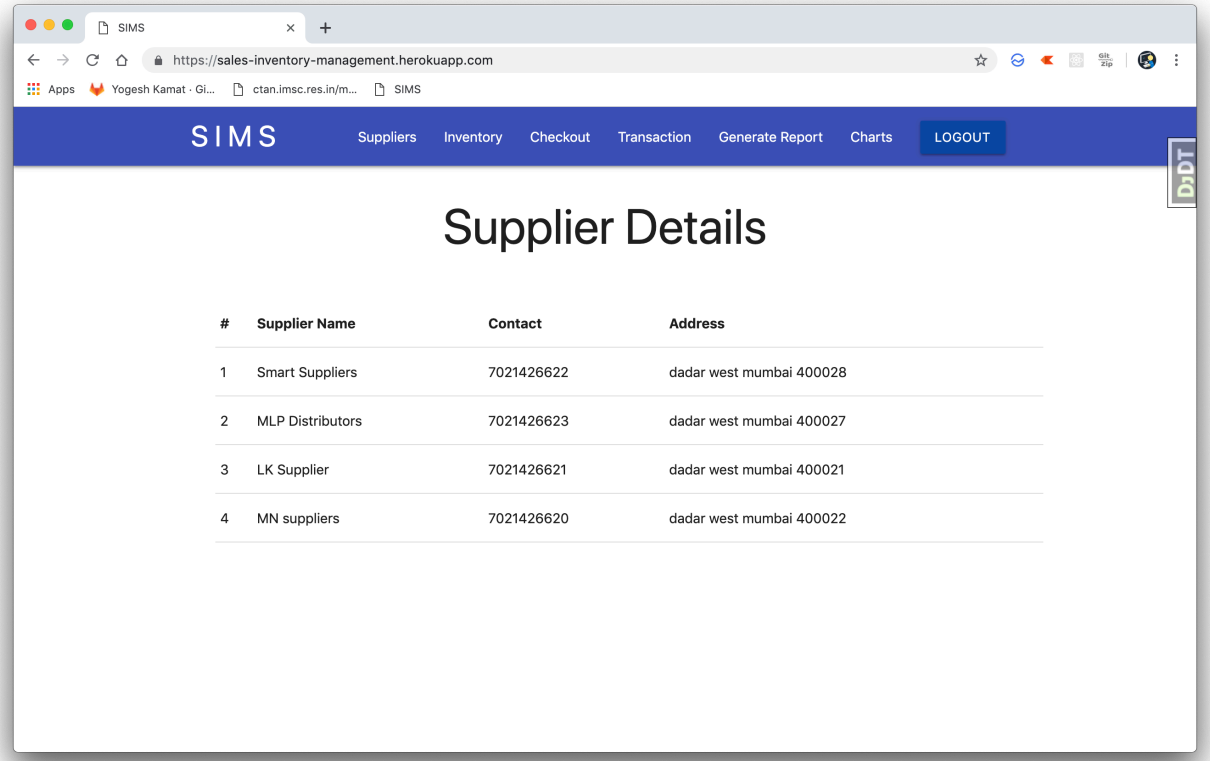
Name : Yogesh D. Kamat **SPIT Batch E** **UID : 2018240066**



OUTPUT SCREENSHOTS WITH EXPLANATION

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

Name : Yogesh D. Kamat **SPIT Batch E** **UID : 2018240066**

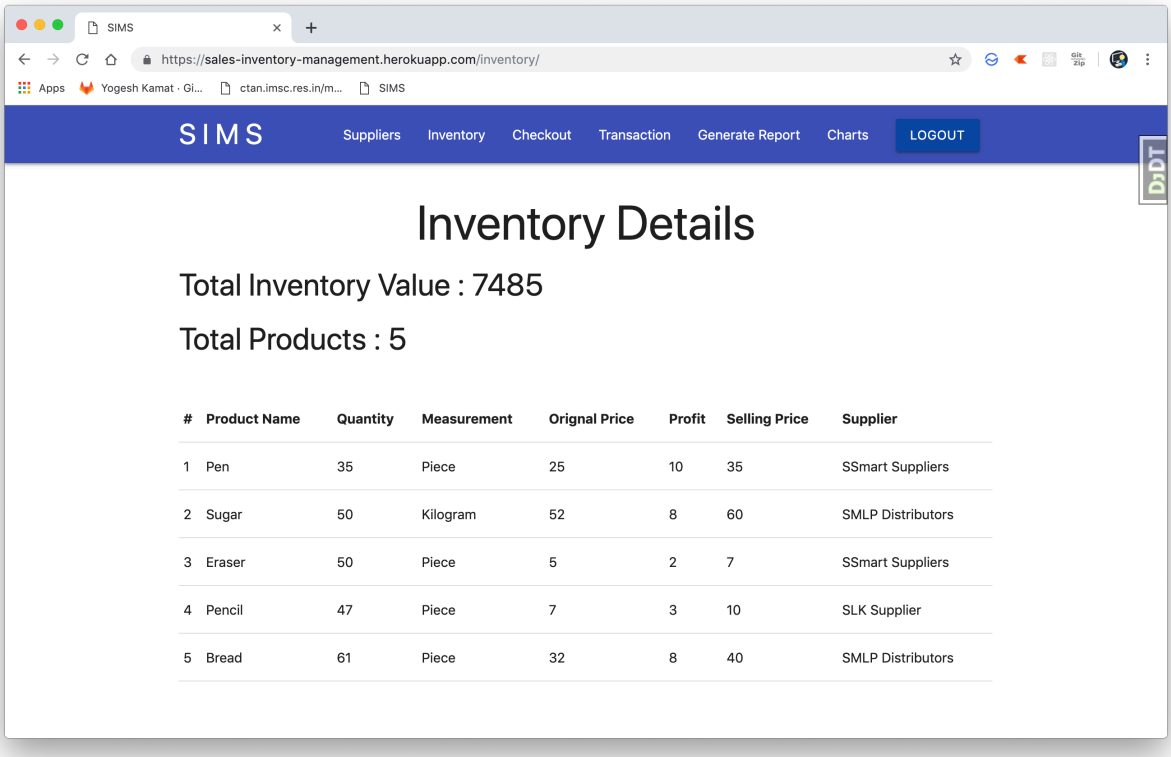


<u>Home Page</u> : (Shows supplier details)

<u>Home Page contains following tabs as well :</u>

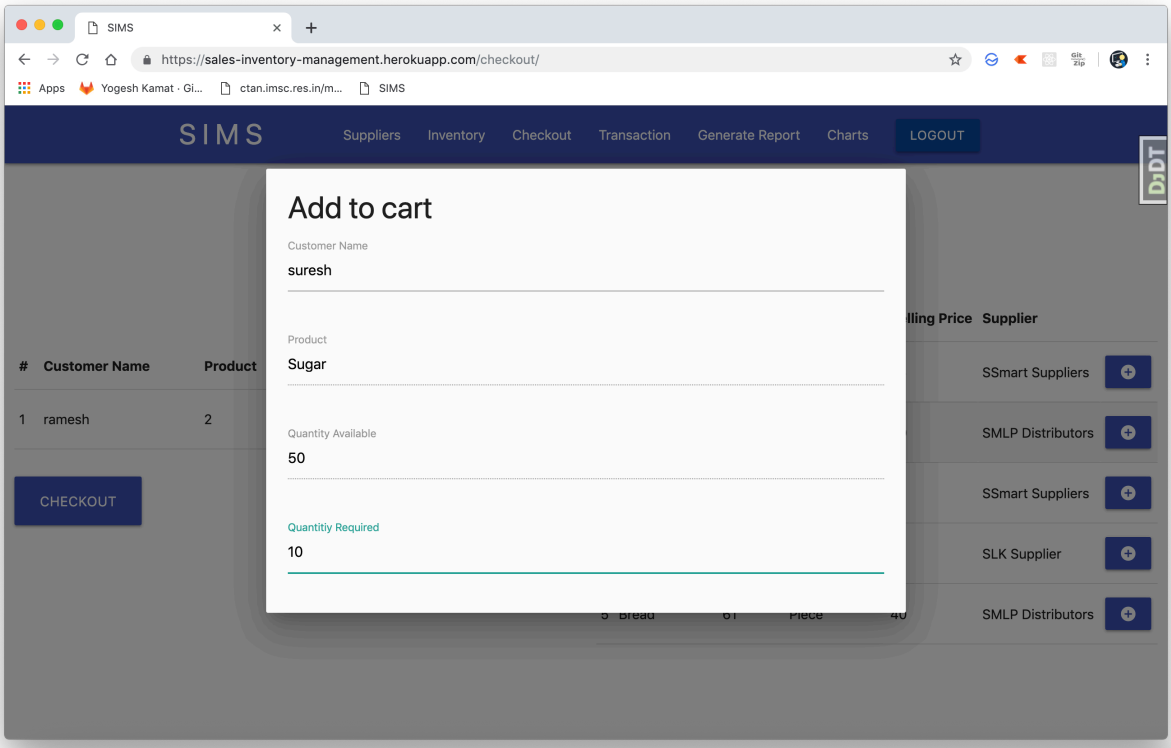
1. **Inventory (shows inventory details)**
2. **Checkout page for sales.**
3. **Transaction page for successful transactions.**
4. **Report generating page**
5. **Graph analytics**
6. **Debug tool to show row sql queries**

1.

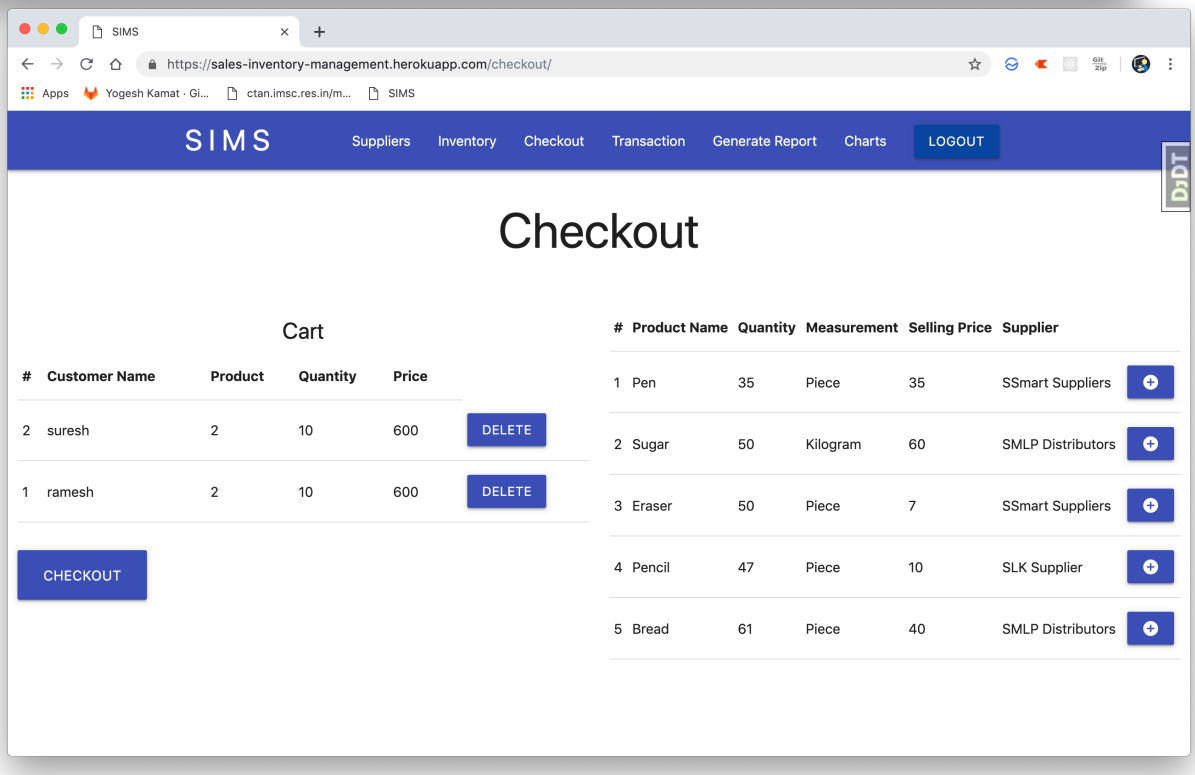


Name : Yogesh D. Kamat **SPIT Batch E** **UID : 2018240066**

2.

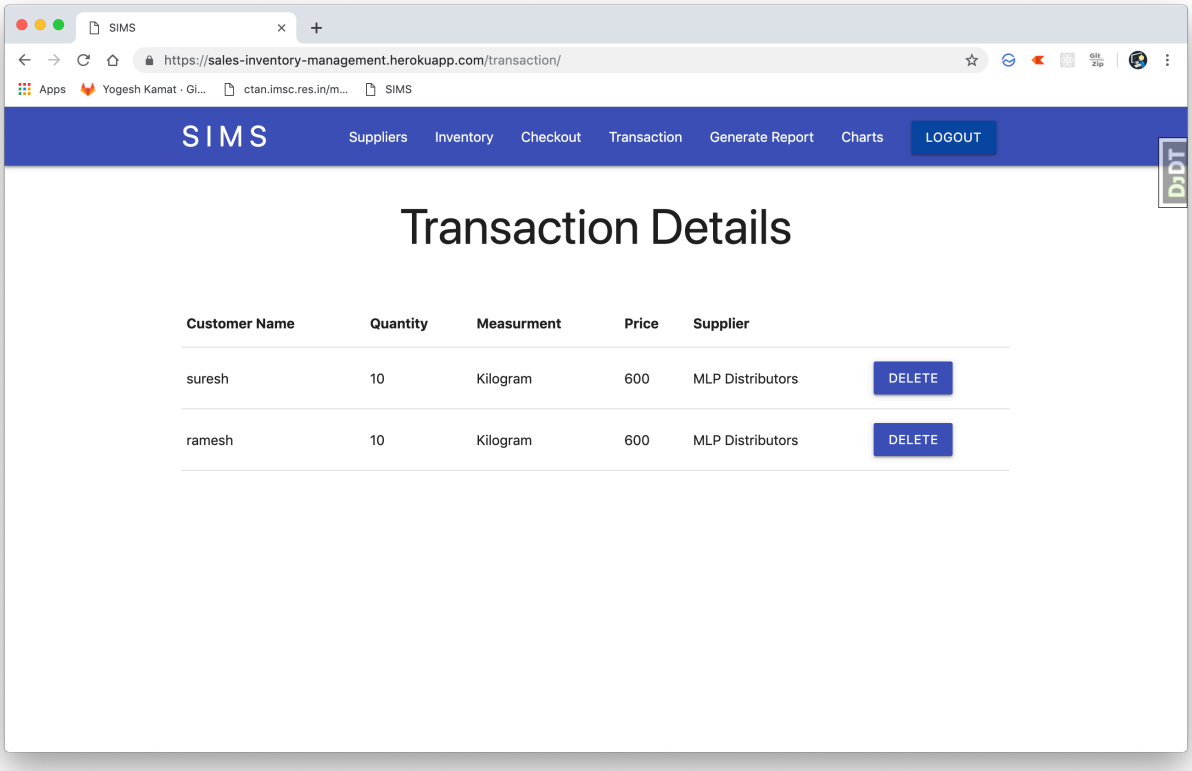


2.

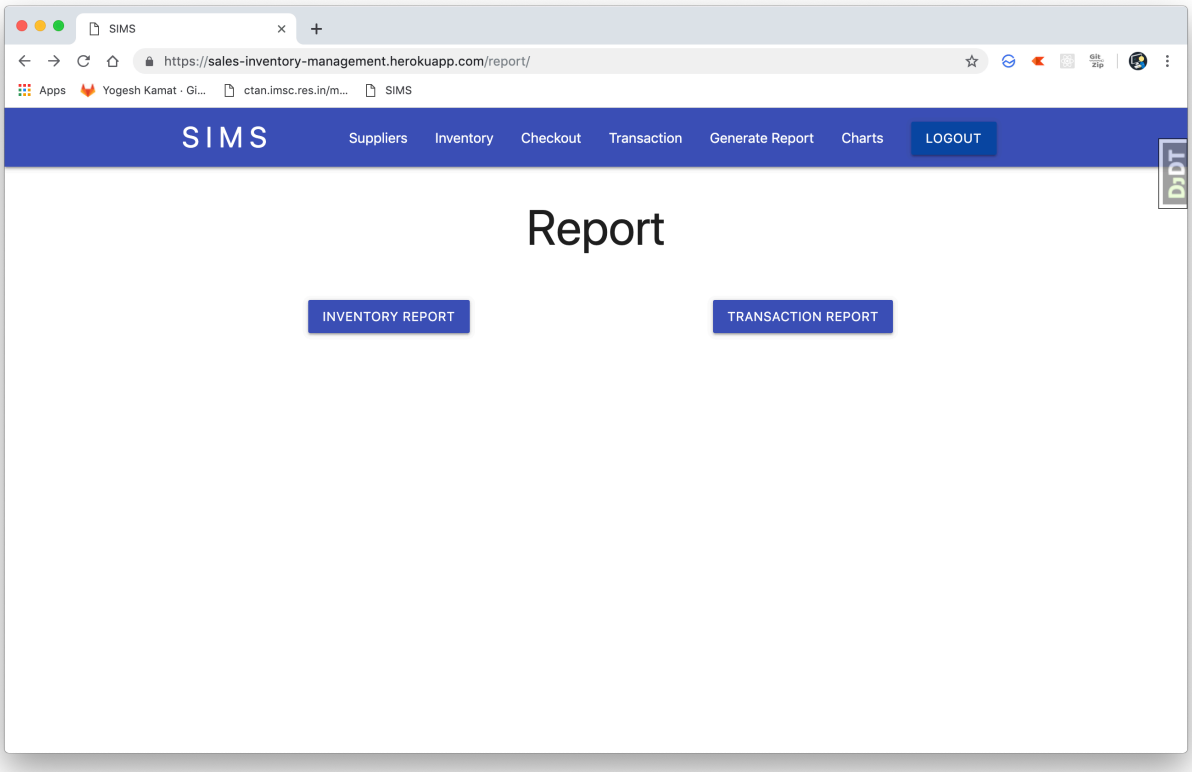


3.

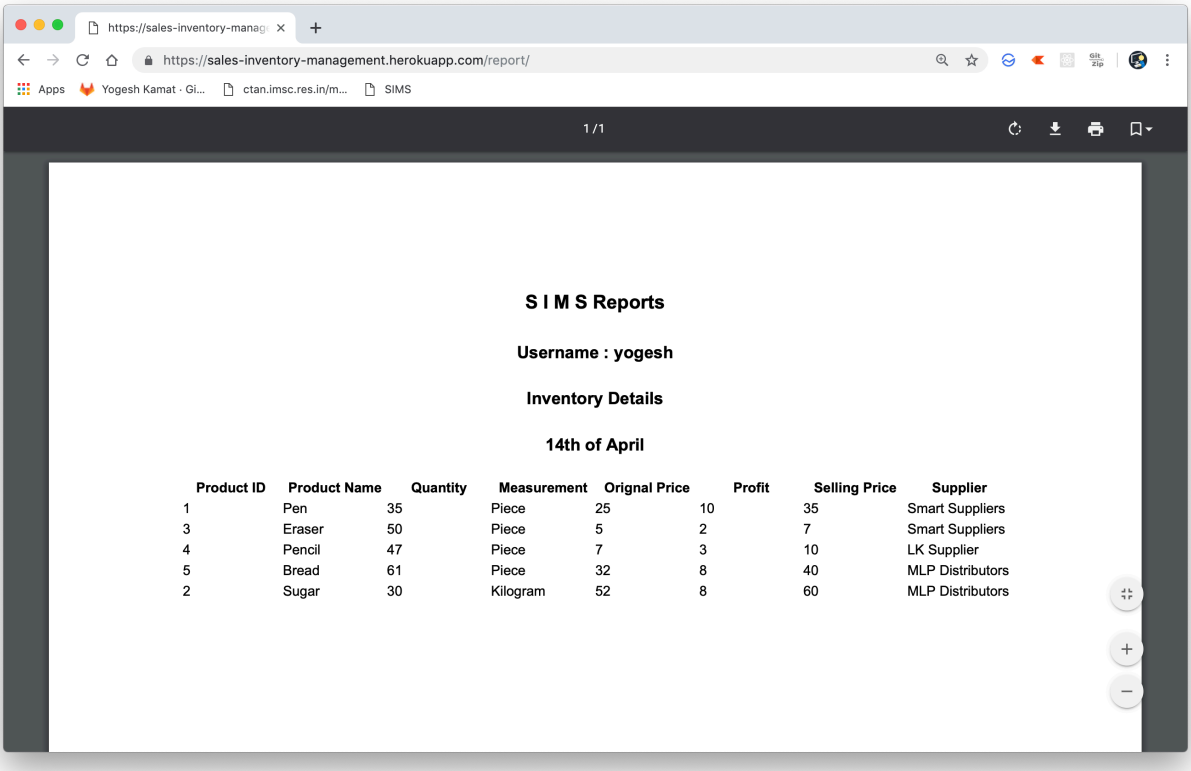
Name : Yogesh D. Kamat **SPIT Batch E** **UID : 2018240066**



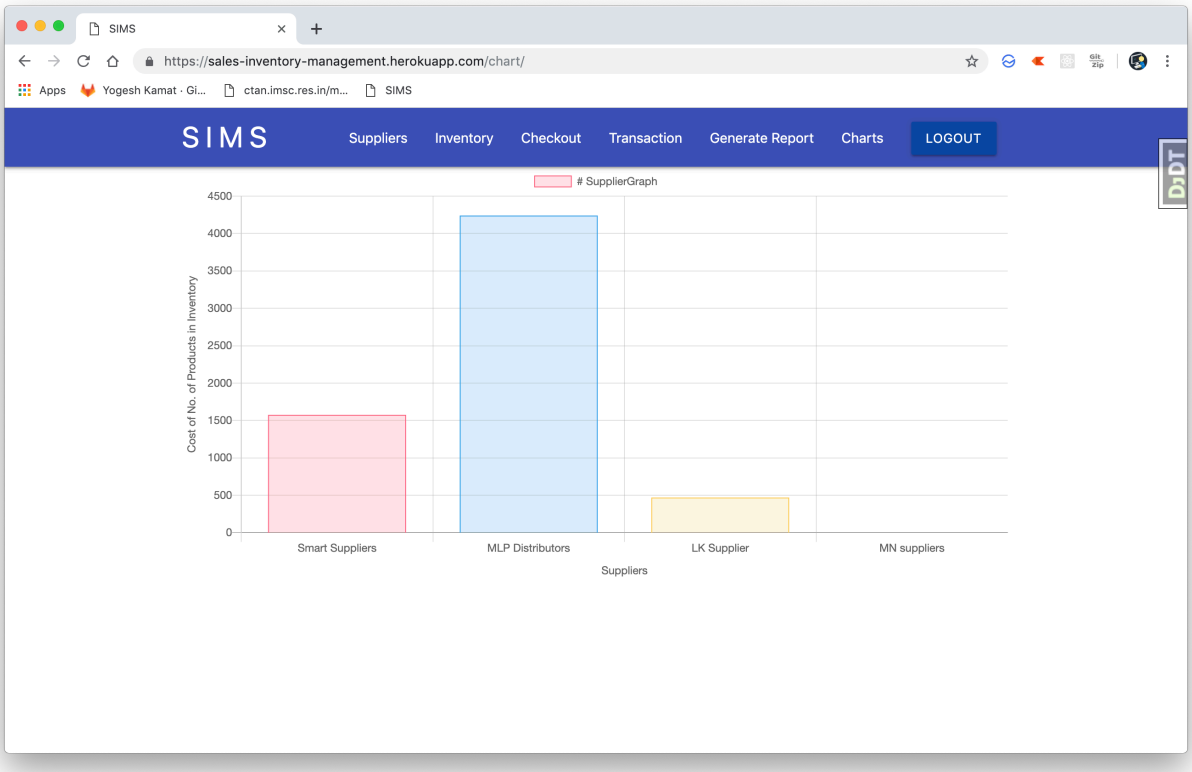
4.



Name : Yogesh D. Kamat **SPIT Batch E** **UID : 2018240066**

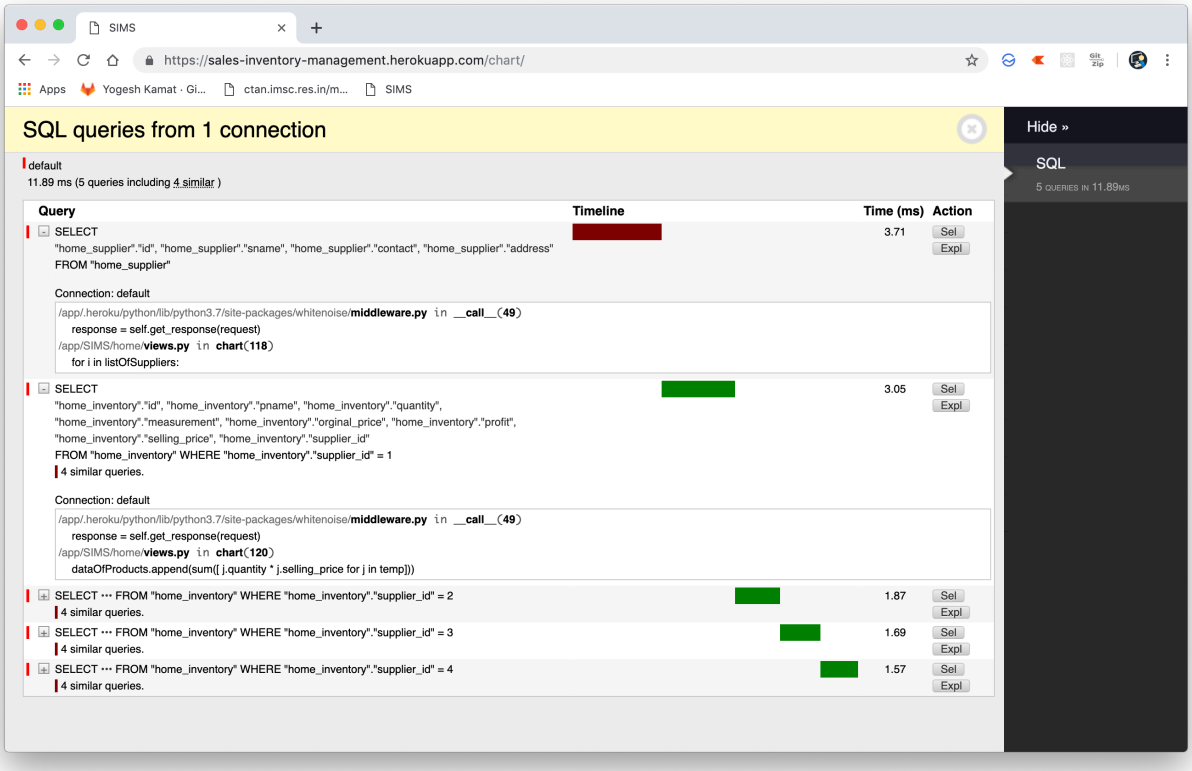


5.



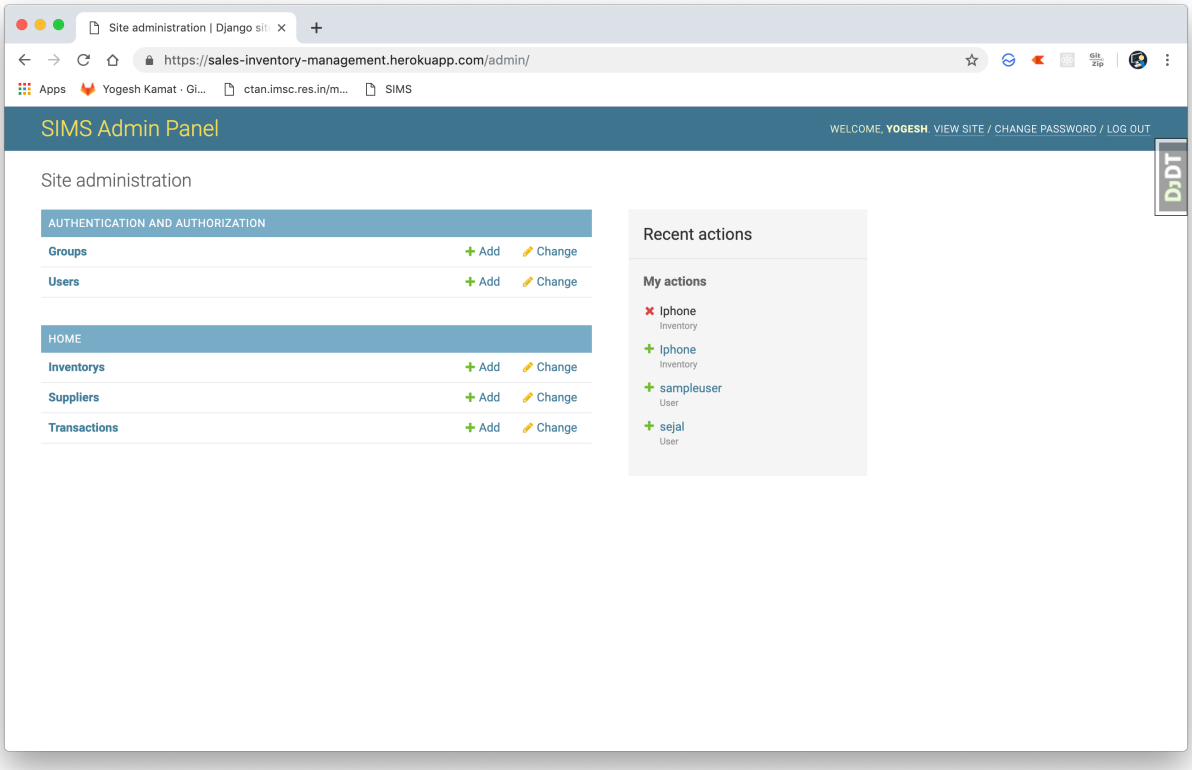
6.

Name : Yogesh D. Kamat **SPIT Batch E** **UID : 2018240066**



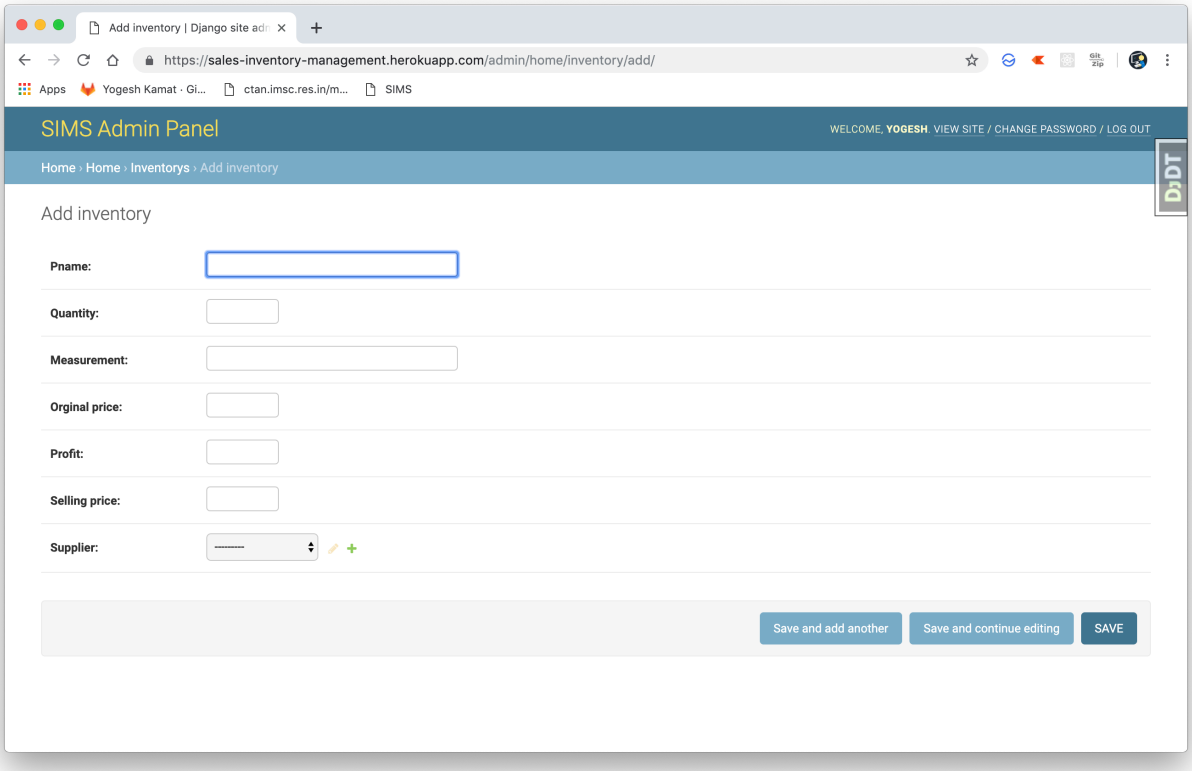
<u>Admin Panel</u> to add or delete inventories or suppliers or transactions. In Admin Panel Only admin user can login.

<u>HomePage for admin panel :</u>

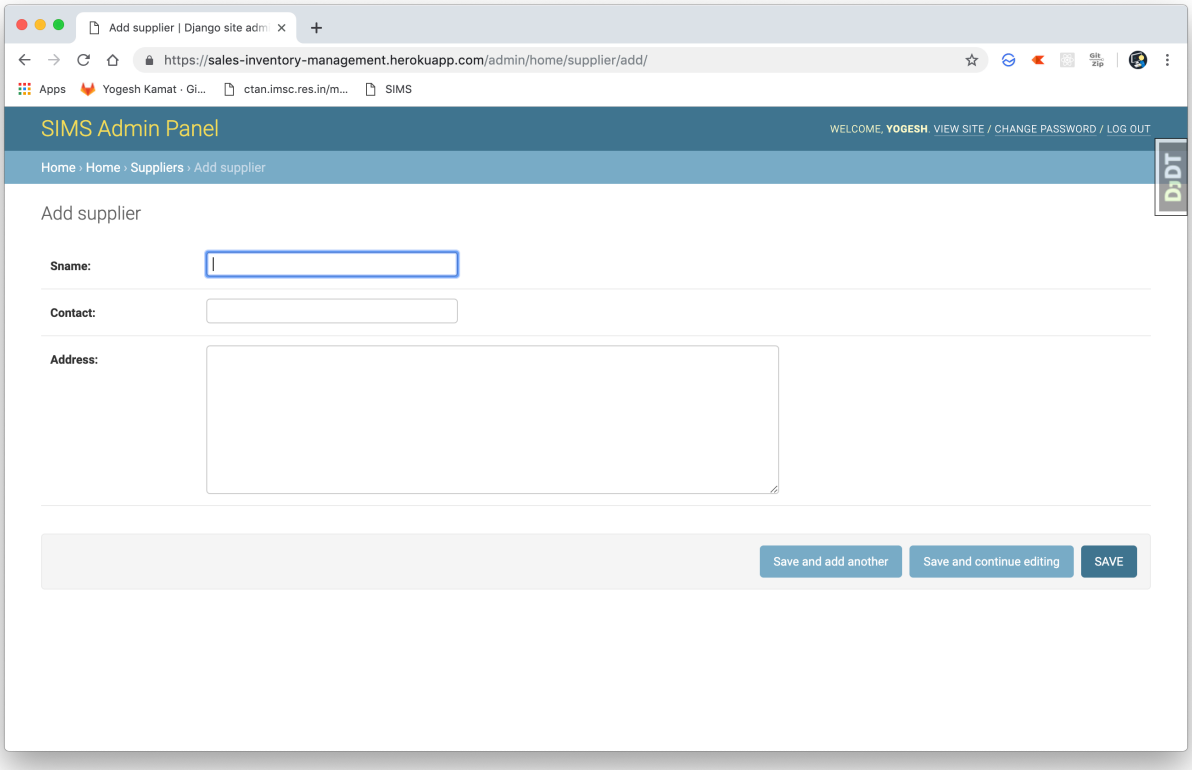


Name : Yogesh D. Kamat **SPIT Batch E** **UID : 2018240066**

Add Inventory form :

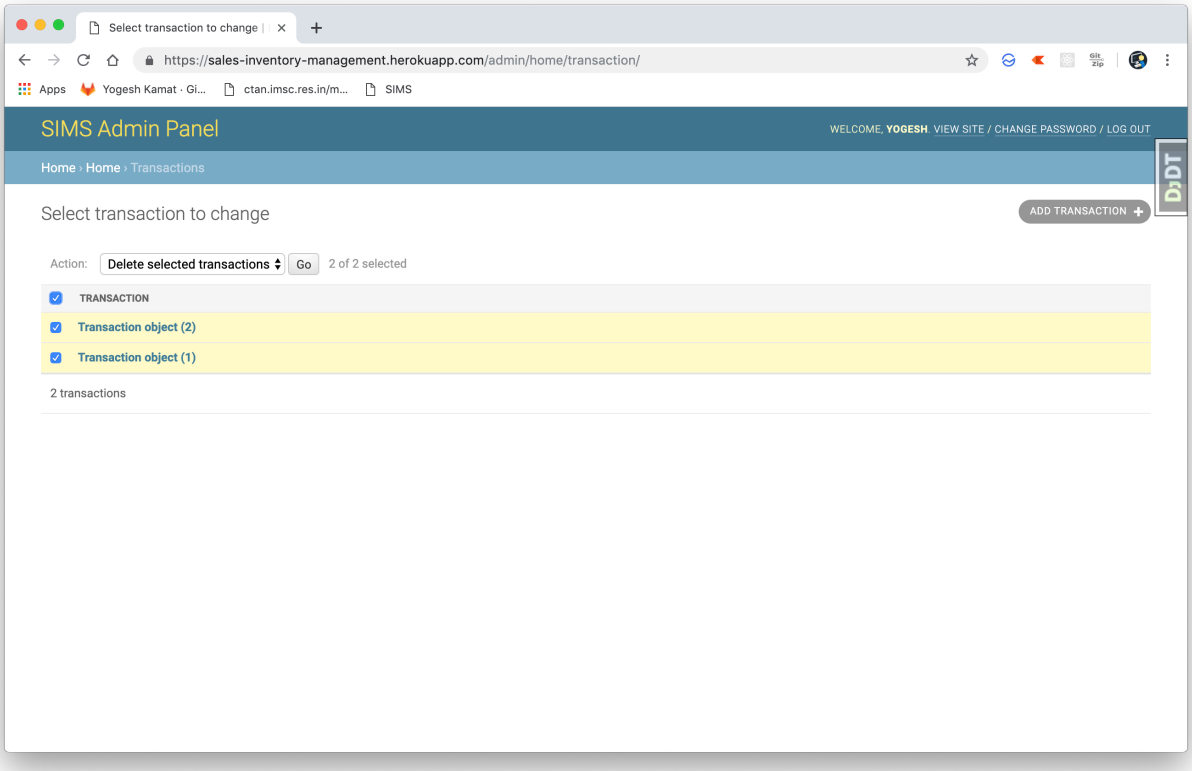


Supplier add form :



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

Name : Yogesh D. Kamat **SPIT Batch E** **UID : 2018240066**



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, HttpResponse
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**