**LAMPIRAN 2**

**BARIS PROGRAM**

1. **File ahp\_procject/ahp\_project/settings.py**

"""

Django settings for ahp\_project project.

"""

BASE\_DIR = os.path.dirname(os.path.dirname(\_\_file\_\_))

SECRET\_KEY = 'uc%cs#pv2+ock$70y-%n&^0xx94-+q##o5aychz1$7)t=5x1r\_'

DEBUG = True

TEMPLATE\_DEBUG = True

ALLOWED\_HOSTS = []

# Application definition

INSTALLED\_APPS = (

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

'lovelips'

)

MIDDLEWARE\_CLASSES = (

'django.contrib.sessions.middleware.SessionMiddleware',

'django.middleware.common.CommonMiddleware',

'django.middleware.csrf.CsrfViewMiddleware',

'django.contrib.auth.middleware.AuthenticationMiddleware',

'django.contrib.auth.middleware.SessionAuthenticationMiddleware',

'django.contrib.messages.middleware.MessageMiddleware',

'django.middleware.clickjacking.XFrameOptionsMiddleware',

)

ROOT\_URLCONF = 'ahp\_project.urls'

WSGI\_APPLICATION = 'ahp\_project.wsgi.application'

# Database

# https://docs.djangoproject.com/en/1.7/ref/settings/#databases

EMAIL\_HOST = 'smtp.gmail.com'

EMAIL\_PORT = 587

EMAIL\_HOST\_USER = 'lovelips.ahp@gmail.com'

EMAIL\_HOST\_PASSWORD = 'lovelips\*\*\*\*\*\*\*\*'

EMAIL\_USE\_TLS = True

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.mysql',

'NAME': 'sukmadyu$ahp\_project',

'USER': 'sukmadyu',

'PASSWORD':'mysqlpass\*\*\*\*',

'HOST':'mysql.server'

}

}

# Internationalization

# https://docs.djangoproject.com/en/1.7/topics/i18n/

LANGUAGE\_CODE = 'en-us'

TIME\_ZONE = 'UTC'

USE\_I18N = True

USE\_L10N = True

USE\_TZ = True

# Static files (CSS, JavaScript, Images)

# https://docs.djangoproject.com/en/1.7/howto/static-files/

STATIC\_PATH = os.path.join(BASE\_DIR, 'static')

STATICFILES\_DIRS = (

STATIC\_PATH,

)

STATIC\_URL = '/static/'

TEMPLATE\_DIRS = (

os.path.join(BASE\_DIR, 'templates'),

)

1. **File ahp\_procject/ahp\_project/urls.py**

from django.conf.urls import patterns, include, url

from django.contrib import admin

from lovelips.views import HomeView

urlpatterns = patterns('',

url(r'^$', HomeView.as\_view(), name='index'),

url(r'^admin/', include(admin.site.urls)),

url(r'^ahp/',include('lovelips.urls')),

)

1. **File ahp\_procject/lovelips/admin.py**

from django.contrib import admin

from lovelips.models import Komentar,Alternatif

admin.site.register(Komentar)

admin.site.register(Alternatif)

1. **File ahp\_procject/lovelips/forms.py**

from django import forms

from .models import \*

class KontakForm(forms.ModelForm):

class Meta:

model = Komentar

fields = ('nama','email','no\_telepon','pesan',)

widgets = {

'nama': forms.TextInput(attrs={'type':'text','class':'form-control','placeholder':'Your Name \*','data-validation-required-message':'Please enter your name.'}),

'email': forms.EmailInput(attrs={'type':'text','class':'form-control','placeholder':'Your Email \*','data-validation-required-message':'Please enter your email.'}),

'no\_telepon': forms.TextInput(attrs={'type':'text','class':'form-control','placeholder':'Your Phone Number \*','data-validation-required-message':'Please enter your phone number.'}),

'pesan': forms.Textarea(attrs={'type':'text','class':'form-control','placeholder':'Your Message \*','data-validation-required-message':'Please enter your message.'}),

}

class NamaProdukForm(forms.ModelForm):

produk\_a=forms.CharField(widget=forms.TextInput(attrs={'class':'form-control '}))

produk\_b=forms.CharField(widget=forms.TextInput(attrs={'class':'form-control'}))

produk\_c=forms.CharField(widget=forms.TextInput(attrs={'class':'form-control'}))

class Meta:

model = Alternatif

fields = ('nama\_product',)

widgets = {

'nama\_product':forms.HiddenInput(),

}

class AHPForm(forms.ModelForm):

harga\_a=forms.CharField(widget=forms.TextInput(attrs={'class':'form-control '}))

harga\_b=forms.CharField(widget=forms.TextInput(attrs={'class':'form-control'}))

harga\_c=forms.CharField(widget=forms.TextInput(attrs={'class':'form-control'}))

isi\_a=forms.CharField(widget=forms.TextInput(attrs={'class':'form-control'}))

isi\_b=forms.CharField(widget=forms.TextInput(attrs={'class':'form-control'}))

isi\_c=forms.CharField(widget=forms.TextInput(attrs={'class':'form-control'}))

pao\_a=forms.CharField(widget=forms.TextInput(attrs={'class':'form-control'}))

pao\_b=forms.CharField(widget=forms.TextInput(attrs={'class':'form-control'}))

pao\_c=forms.CharField(widget=forms.TextInput(attrs={'class':'form-control'}))

time\_a=forms.CharField(widget=forms.TextInput(attrs={'class':'form-control'}))

time\_b=forms.CharField(widget=forms.TextInput(attrs={'class':'form-control'}))

time\_c=forms.CharField(widget=forms.TextInput(attrs={'class':'form-control'}))

cruelty\_free\_a=forms.CharField(widget=forms.TextInput(attrs={'class':'form-control'}))

cruelty\_free\_b=forms.CharField(widget=forms.TextInput(attrs={'class':'form-control'}))

cruelty\_free\_c=forms.CharField(widget=forms.TextInput(attrs={'class':'form-control'}))

class Meta:

model = Alternatif

fields = ('harga','isi','pao','time','cruelty\_free')

widgets = {

'harga':forms.HiddenInput(),

'isi':forms.HiddenInput(),

'pao':forms.HiddenInput(),

'time':forms.HiddenInput(),

'cruelty\_free':forms.HiddenInput(),}

class AHPKriteriaForm(forms.ModelForm):

kri\_harga\_isi=forms.FloatField()

kri\_harga\_pao=forms.FloatField()

kri\_harga\_time=forms.FloatField()

kri\_harga\_cf=forms.FloatField()

kri\_isi\_pao=forms.FloatField()

kri\_isi\_time=forms.FloatField()

kri\_isi\_cf=forms.FloatField()

kri\_pao\_time=forms.FloatField()

kri\_pao\_cf=forms.FloatField()

kri\_time\_cf=forms.FloatField()

class Meta:

model = Alternatif

fields = ('kri',)

widgets = {

'kri':forms.HiddenInput(),}

1. **File ahp\_procject/lovelips/models.py**

from django.db import models

from django.utils import timezone

class Alternatif(models.Model):

kri=models.TextField(null=True,blank=True)

harga=models.TextField(null=True,blank=True)

isi=models.TextField(null=True,blank=True)

pao=models.TextField(null=True,blank=True)

time=models.TextField(null=True,blank=True)

cruelty\_free=models.TextField(null=True,blank=True)

nama\_product=models.TextField(null=True,blank=True)

class Komentar(models.Model):

nama=models.CharField(max\_length=50)

email=models.EmailField()

no\_telepon=models.CharField(max\_length=12)

pesan=models.TextField()

created\_date = models.DateTimeField(default=timezone.now)

1. **File ahp\_procject/lovelips/urls.py**

from django.conf.urls import patterns, include, url

from django.contrib import admin

from . import views

urlpatterns = patterns('',

url(r'^admin/', include(admin.site.urls)),

url(r'^input-ahp/(?P<id>\d+)$', views.InputAHPView.as\_view(id=None),name='input-ahp'),

url(r'^input-nama/(?P<id>\d+)$', views.InputNamaView.as\_view(id=None),name='input-nama'),

url(r'^input-kri/$', views.InputKriteriaAHPView.as\_view(),name='input-kri'),

url(r'^detail-ahp/(?P<id>\d+)$', views.DetailAHPView.as\_view(id=None),name='detail-ahp'),

url(r'^konsistensi-ahp/(?P<id>\d+)$', views.KonsistensiAHPView.as\_view(id=None),name='konsistensi-ahp'),

)

1. **File ahp\_procject/lovelips/views.py**

import simplejson as json

import numpy as np

from numpy import linalg as LA

from decimal import Decimal as D

from django.conf import settings

from django.contrib import messages

from django.core.urlresolvers import reverse

from django.core.mail import send\_mail

from django.http import HttpResponseRedirect

from django.shortcuts import render, get\_object\_or\_404

from django.views.generic import View, UpdateView

from .forms import \*

from .models import \*

def home(request):

return render(request,'home.html')

def index(request):

return render(request,'index.html')

class HomeView(View):

template\_name = 'index.html'

def get(self,request):

form = KontakForm()

return render(request,self.template\_name,{'form':form})

def post(self,request):

form = KontakForm(request.POST or None)

if form.is\_valid():

kontak=form.save(commit=False)

kontak.save()

subject= 'Pesan pada website lovelips dari {}'.format(kontak.nama)

message= 'nama: {} \nemail: {} \nno telepon: {} \nmessage: {}'.format(kontak.nama,kontak.email,kontak.no\_telepon,kontak.pesan)

email\_from= settings.EMAIL\_HOST\_USER

email\_to= 'sukmadyu@gmail.com'

send\_mail(subject, message, email\_from, [email\_to], fail\_silently=False)

messages.success(request,'Your comment has been successfully saved')

return HttpResponseRedirect(reverse('index'))

else:

messages.error(request,'An error has occured, please try again')

return render(request,self.template\_name,{'form':form})

class InputKriteriaAHPView(View):

template\_name = 'ahp/input-kri.html'

def get(self,request):

form = AHPKriteriaForm()

return render(request,self.template\_name,{'form':form})

def post(self,request):

form = AHPKriteriaForm(request.POST or None)

if form.is\_valid():

ahp=form.save(commit=False)

kri\_harga\_isi=form.cleaned\_data['kri\_harga\_isi']

kri\_harga\_pao=form.cleaned\_data['kri\_harga\_pao']

kri\_harga\_time=form.cleaned\_data['kri\_harga\_time']

kri\_harga\_cf=form.cleaned\_data['kri\_harga\_cf']

kri\_isi\_pao=form.cleaned\_data['kri\_isi\_pao']

kri\_isi\_time=form.cleaned\_data['kri\_isi\_time']

kri\_isi\_cf=form.cleaned\_data['kri\_isi\_cf']

kri\_pao\_time=form.cleaned\_data['kri\_pao\_time']

kri\_pao\_cf=form.cleaned\_data['kri\_pao\_cf']

kri\_time\_cf=form.cleaned\_data['kri\_time\_cf']

kriteria=[]

kriteria.append(kri\_harga\_isi),kriteria.append(kri\_harga\_pao),kriteria.append(kri\_harga\_time)

kriteria.append(kri\_harga\_cf),kriteria.append(kri\_isi\_pao),kriteria.append(kri\_isi\_time)

kriteria.append(kri\_isi\_cf),kriteria.append(kri\_pao\_time),kriteria.append(kri\_pao\_cf),kriteria.append(kri\_time\_cf)

ahp.kri= json.dumps(kriteria)

ahp.save()

messages.success(request,'Input nilai perbandingan kriteria telah berhasil')

return HttpResponseRedirect(reverse('konsistensi-ahp', kwargs={'id':ahp.id}))

else:

messages.error(request,'Ada kesalahan yang terjadi, mohon ulangi lagi')

return render(request,self.template\_name,{'form':form})

class InputAHPView(View):

template\_name = 'ahp/input-ahp.html'

id = None

def get(self, request, id):

ahp = Alternatif.objects.get(id=id)

form = AHPForm(instance=ahp)

jsonDec = json.decoder.JSONDecoder()

nama = jsonDec.decode(ahp.nama\_product)

nama\_a=nama[0]

nama\_b=nama[1]

nama\_c=nama[2]

return render(request,self.template\_name,{'form':form,'ahp':ahp,

'nama\_a':nama\_a,'nama\_b':nama\_b,'nama\_c':nama\_c,

})

def post(self, request, id=None):

ahp\_ob = Alternatif.objects.get(id=id)

form = AHPForm(request.POST, instance=ahp\_ob)

if form.is\_valid():

ahp=form.save(commit=False)

harga\_a=form.cleaned\_data['harga\_a']

harga\_b=form.cleaned\_data['harga\_b']

harga\_c=form.cleaned\_data['harga\_c']

harga=[]

harga.append(harga\_a),harga.append(harga\_b)

harga.append(harga\_c)

ahp.harga= json.dumps(harga)

isi\_a=form.cleaned\_data['isi\_a']

isi\_b=form.cleaned\_data['isi\_b']

isi\_c=form.cleaned\_data['isi\_c']

isi=[]

isi.append(isi\_a),isi.append(isi\_b),isi.append(isi\_c)

ahp.isi= json.dumps(isi)

pao\_a=form.cleaned\_data['pao\_a']

pao\_b=form.cleaned\_data['pao\_b']

pao\_c=form.cleaned\_data['pao\_c']

pao=[]

pao.append(pao\_a),pao.append(pao\_b),pao.append(pao\_c)

ahp.pao= json.dumps(pao)

time\_a=form.cleaned\_data['time\_a']

time\_b=form.cleaned\_data['time\_b']

time\_c=form.cleaned\_data['time\_c']

time=[]

time.append(time\_a),time.append(time\_b),time.append(time\_c)

ahp.time= json.dumps(time)

cruelty\_free\_a=form.cleaned\_data['cruelty\_free\_a']

cruelty\_free\_b=form.cleaned\_data['cruelty\_free\_b']

cruelty\_free\_c=form.cleaned\_data['cruelty\_free\_c']

cruelty\_free=[]

cruelty\_free.append(cruelty\_free\_a)

cruelty\_free.append(cruelty\_free\_b)

cruelty\_free.append(cruelty\_free\_c)

ahp.cruelty\_free= json.dumps(cruelty\_free)

ahp.save()

messages.success(request,'Input data produk berhasil')

return HttpResponseRedirect(reverse('detail-ahp', kwargs={'id':ahp.id}))

else:

messages.error(request,'Kesalahan dalam input data produk, mohon ulangi lagi')

return render(request,self.template\_name,{'form':form})

RI = (0, 0, D('0.58'), D('0.9'), D('1.12'),

D('1.24'), D('1.32'), D('1.41'), D('1.45'), D('1.49'),

D('1.51'), D('1.48'), D('1.56'), D('1.57'), D('1.59')

)

def calculateConsistency(arr):

eva = max(LA.eig(arr)[0]).real

n = len(arr)

CI = (eva-n) / (n-1)

return CI / float(RI[n])

def calculateWeights(arr, rounding=4):

PLACES = D(10) \*\* -(rounding)

evas, eves = LA.eig(arr)

eva = max(evas)

eva\_idx = evas.tolist().index(eva)

eve = eves.take((eva\_idx,), axis=1)

normalized = eve / sum(eve)

vector = [abs(e[0]) for e in normalized]

arr= [ D( v ).quantize(PLACES) for v in vector ]

return arr

def calculateWeightsmin(arr, rounding=4):

PLACES = D(10) \*\* -(rounding)

evas, eves = LA.eig(arr)

eva = min(evas)

eva\_idx = evas.tolist().index(eva)

eve = eves.take((eva\_idx,), axis=1)

normalized = eve / sum(eve)

vector = [abs(e[0]) for e in normalized]

arr= [ D( v ).quantize(PLACES) for v in vector ]

return arr

class DetailAHPView(View):

template\_name = 'ahp/hasil-ahp.html'

id = None

def get(self,request,id=None):

alternatif = Alternatif.objects.get(id=id)

jsonDec = json.decoder.JSONDecoder()

nama = jsonDec.decode(alternatif.nama\_product)

nama\_a=nama[0]

nama\_b=nama[1]

nama\_c=nama[2]

harga = jsonDec.decode(alternatif.harga)

harga\_a=float(harga[0])

harga\_b=float(harga[1])

harga\_c=float(harga[2])

harga\_arr=np.array([harga\_a/harga\_a,harga\_b/harga\_a,harga\_c/harga\_a,

harga\_a/harga\_b,harga\_b/harga\_b,harga\_c/harga\_b,

harga\_a/harga\_c,harga\_b/harga\_c,harga\_c/harga\_c])

harga\_arr=harga\_arr.reshape(3,3)

harga\_arr=calculateWeights(harga\_arr)

harga\_arr\_a=harga\_arr[0]

harga\_arr\_b=harga\_arr[1]

harga\_arr\_c=harga\_arr[2]

harga\_arr=np.array(harga\_arr)

harga\_arr=harga\_arr.reshape(3,1)

isi = jsonDec.decode(alternatif.isi)

isi\_a=float(isi[0])

isi\_b=float(isi[1])

isi\_c=float(isi[2])

isi\_arr=np.array([isi\_a/isi\_a,isi\_b/isi\_a,isi\_c/isi\_a,

isi\_a/isi\_b,isi\_b/isi\_b,isi\_c/isi\_b,

isi\_a/isi\_c,isi\_b/isi\_c,isi\_c/isi\_c])

isi\_arr=isi\_arr.reshape(3,3)

isi\_arr=calculateWeights(isi\_arr)

isi\_arr\_a=isi\_arr[0]

isi\_arr\_b=isi\_arr[1]

isi\_arr\_c=isi\_arr[2]

isi\_arr=np.array(isi\_arr)

isi\_arr=isi\_arr.reshape(3,1)

pao = jsonDec.decode(alternatif.pao)

pao\_a=float(pao[0])

pao\_b=float(pao[1])

pao\_c=float(pao[2])

pao\_arr=np.array([pao\_a/pao\_a,pao\_b/pao\_a,pao\_c/pao\_a,

pao\_a/pao\_b,pao\_b/pao\_b,pao\_c/pao\_b,

pao\_a/pao\_c,pao\_b/pao\_c,pao\_c/pao\_c])

pao\_arr=pao\_arr.reshape(3,3)

pao\_arr=calculateWeights(pao\_arr)

pao\_arr\_a=pao\_arr[0]

pao\_arr\_b=pao\_arr[1]

pao\_arr\_c=pao\_arr[2]

pao\_arr=np.array(pao\_arr)

pao\_arr=pao\_arr.reshape(3,1)

time = jsonDec.decode(alternatif.time)

time\_a=float(time[0])

time\_b=float(time[1])

time\_c=float(time[2])

time\_arr=np.array([time\_a/time\_a,time\_b/time\_a,time\_c/time\_a,

time\_a/time\_b,time\_b/time\_b,time\_c/time\_b,

time\_a/time\_c,time\_b/time\_c,time\_c/time\_c])

time\_arr=time\_arr.reshape(3,3)

time\_arr=calculateWeights(time\_arr)

time\_arr\_a=time\_arr[0]

time\_arr\_b=time\_arr[1]

time\_arr\_c=time\_arr[2]

time\_arr=np.array(time\_arr)

time\_arr=time\_arr.reshape(3,1)

cruelty\_free = jsonDec.decode(alternatif.cruelty\_free)

cruelty\_free\_a=float(cruelty\_free[0])

cruelty\_free\_b=float(cruelty\_free[1])

cruelty\_free\_c=float(cruelty\_free[2])

cruelty\_free\_arr=np.array([cruelty\_free\_a/cruelty\_free\_a,cruelty\_free\_b/cruelty\_free\_a,cruelty\_free\_c/cruelty\_free\_a,

cruelty\_free\_a/cruelty\_free\_b,cruelty\_free\_b/cruelty\_free\_b,cruelty\_free\_c/cruelty\_free\_b,

cruelty\_free\_a/cruelty\_free\_c,cruelty\_free\_b/cruelty\_free\_c,cruelty\_free\_c/cruelty\_free\_c])

cruelty\_free\_arr=cruelty\_free\_arr.reshape(3,3)

cruelty\_free\_arr=calculateWeights(cruelty\_free\_arr)

cruelty\_free\_arr\_a=cruelty\_free\_arr[0]

cruelty\_free\_arr\_b=cruelty\_free\_arr[1]

cruelty\_free\_arr\_c=cruelty\_free\_arr[2]

cruelty\_free\_arr=np.array(cruelty\_free\_arr)

cruelty\_free\_arr=cruelty\_free\_arr.reshape(3,1)

kri = jsonDec.decode(alternatif.kri)

kri\_a=float(kri[0])

kri\_b=float(kri[1])

kri\_c=float(kri[2])

kri\_d=float(kri[3])

kri\_e=float(kri[4])

kri\_f=float(kri[5])

kri\_g=float(kri[6])

kri\_h=float(kri[7])

kri\_i=float(kri[8])

kri\_j=float(kri[9])

kri\_k=1/kri\_a

kri\_l=1/kri\_b

kri\_m=1/kri\_e

kri\_n=1/kri\_c

kri\_o=1/kri\_f

kri\_p=1/kri\_h

kri\_q=1/kri\_d

kri\_r=1/kri\_g

kri\_s=1/kri\_i

kri\_t=1/kri\_j

kri\_arr=np.array([1,kri\_a,kri\_b,kri\_c,kri\_d,

1/kri\_a,1,kri\_e,kri\_f,kri\_g,

1/kri\_b,1/kri\_e,1,kri\_h,kri\_i,

1/kri\_c,1/kri\_f,1/kri\_h,1,kri\_j,

1/kri\_d,1/kri\_g,1/kri\_i,1/kri\_j,1])

kri\_arr=kri\_arr.reshape(5,5)

cons=calculateConsistency(kri\_arr)

kri\_arr=calculateWeights(kri\_arr)

kri\_harga=kri\_arr[0]

kri\_isi=kri\_arr[1]

kri\_pao=kri\_arr[2]

kri\_time=kri\_arr[3]

kri\_cruelty\_free=kri\_arr[4]

kri\_arr=np.array(kri\_arr)

kri\_arr=kri\_arr.reshape(5,1)

alt=np.concatenate((harga\_arr, isi\_arr), axis=1)

alt=np.concatenate((alt, pao\_arr), axis=1)

alt=np.concatenate((alt, time\_arr), axis=1)

alt=np.concatenate((alt, cruelty\_free\_arr), axis=1)

alt=alt.reshape(3,5)

hasil=np.dot(alt,kri\_arr)

produk\_a=float(hasil[0])

produk\_b=float(hasil[1])

produk\_c=float(hasil[2])

maks=np.amax(hasil)

maks=np.array(maks)

hasil=hasil.tolist()

inde=hasil.index(maks)

produk=[nama\_a,nama\_b, nama\_c]

return render(request,self.template\_name,{'harga\_a':int(harga\_a),'harga\_b':int(harga\_b),'harga\_c':int(harga\_c),

'isi\_a':int(isi\_a),'isi\_b':int(isi\_b),'isi\_c':int(isi\_c),

'pao\_a':int(pao\_a),'pao\_b':int(pao\_b),'pao\_c':int(pao\_c),

'time\_a':int(time\_a),'time\_b':int(time\_b),'time\_c':int(time\_c),

'cruelty\_free\_a':int(cruelty\_free\_a),'cruelty\_free\_b':int(cruelty\_free\_b),

'cruelty\_free\_c':int(cruelty\_free\_c),

'bobot\_harga\_a':harga\_arr\_a,'bobot\_harga\_b':harga\_arr\_b,

'bobot\_harga\_c':harga\_arr\_c,

'bobot\_isi\_a':isi\_arr\_a,'bobot\_isi\_b':isi\_arr\_b,'bobot\_isi\_c':isi\_arr\_c,

'bobot\_pao\_a':pao\_arr\_a,'bobot\_pao\_b':pao\_arr\_b,'bobot\_pao\_c':pao\_arr\_c,

'bobot\_time\_a':time\_arr\_a,'bobot\_time\_b':time\_arr\_b,

'bobot\_time\_c':time\_arr\_c,

'bobot\_cruelty\_free\_a':cruelty\_free\_arr\_a,

'bobot\_cruelty\_free\_b':cruelty\_free\_arr\_b,

'bobot\_cruelty\_free\_c':cruelty\_free\_arr\_c,

'kri\_a':kri\_a,'kri\_b':kri\_b,'kri\_c':kri\_c,

'kri\_d':kri\_d,'kri\_e':kri\_e,'kri\_e':kri\_e,'kri\_f':kri\_f,

'kri\_g':kri\_g,'kri\_h':kri\_h,'kri\_i':kri\_i,'kri\_j':kri\_j,'kri\_k':kri\_k,'kri\_l':kri\_l,

'kri\_m':kri\_m,'kri\_n':kri\_n,'kri\_o':kri\_o,'kri\_p':kri\_p,'kri\_q':kri\_q,'kri\_r':kri\_r,

'kri\_s':kri\_s,'kri\_t':kri\_t,

'bobot\_harga':kri\_harga,'bobot\_isi':kri\_isi,'bobot\_pao':kri\_pao,

'bobot\_time':kri\_time,'bobot\_cruelty\_free':kri\_cruelty\_free,

'produk\_a':produk\_a,'produk\_b':produk\_b,'produk\_c':produk\_c,

'produk\_max':produk[inde],'cons':cons,'cons\_pre':cons\*100,

'nama\_a':nama\_a,'nama\_b':nama\_b,'nama\_c':nama\_c,

})

class KonsistensiAHPView(View):

template\_name = 'ahp/konsistensi-ahp.html'

id = None

def get(self,request,id=None):

alternatif = Alternatif.objects.get(id=id)

jsonDec = json.decoder.JSONDecoder()

kri = jsonDec.decode(alternatif.kri)

kri\_a=float(kri[0])

kri\_b=float(kri[1])

kri\_c=float(kri[2])

kri\_d=float(kri[3])

kri\_e=float(kri[4])

kri\_f=float(kri[5])

kri\_g=float(kri[6])

kri\_h=float(kri[7])

kri\_i=float(kri[8])

kri\_j=float(kri[9])

kri\_k=1/kri\_a

kri\_l=1/kri\_b

kri\_m=1/kri\_e

kri\_n=1/kri\_c

kri\_o=1/kri\_f

kri\_p=1/kri\_h

kri\_q=1/kri\_d

kri\_r=1/kri\_g

kri\_s=1/kri\_i

kri\_t=1/kri\_j

kri\_arr=np.array([1,kri\_a,kri\_b,kri\_c,kri\_d,

1/kri\_a,1,kri\_e,kri\_f,kri\_g,

1/kri\_b,1/kri\_e,1,kri\_h,kri\_i,

1/kri\_c,1/kri\_f,1/kri\_h,1,kri\_j,

1/kri\_d,1/kri\_g,1/kri\_i,1/kri\_j,1])

kri\_arr=kri\_arr.reshape(5,5)

cons=calculateConsistency(kri\_arr)

kri\_arr=calculateWeights(kri\_arr)

kri\_harga=kri\_arr[0]

kri\_isi=kri\_arr[1]

kri\_pao=kri\_arr[2]

kri\_time=kri\_arr[3]

kri\_cruelty\_free=kri\_arr[4]

kri\_arr=np.array(kri\_arr)

kri\_arr=kri\_arr.reshape(5,1)

return render(request,self.template\_name,{'kri\_a':kri\_a,'kri\_b':kri\_b,'kri\_c':kri\_c,

'kri\_d':kri\_d,'kri\_e':kri\_e,'kri\_e':kri\_e,'kri\_f':kri\_f,

'kri\_g':kri\_g,'kri\_h':kri\_h,'kri\_i':kri\_i,

'kri\_j':kri\_j,'kri\_k':kri\_k,'kri\_l':kri\_l,

'kri\_m':kri\_m,'kri\_n':kri\_n,'kri\_o':kri\_o,

'kri\_p':kri\_p,'kri\_q':kri\_q,'kri\_r':kri\_r,

'kri\_s':kri\_s,'kri\_t':kri\_t,

'bobot\_harga':kri\_harga,'bobot\_isi':kri\_isi,'bobot\_pao':kri\_pao,

'bobot\_time':kri\_time,'bobot\_cruelty\_free':kri\_cruelty\_free,

'cons':cons,'cons\_pre':cons\*100,'alternatif':alternatif

})

class InputNamaView(View):

template\_name = 'ahp/input-nama.html'

id = None

def get(self, request, id):

ahp = Alternatif.objects.get(id=id)

form = NamaProdukForm(instance=ahp)

return render(request,self.template\_name,{'form':form,'ahp':ahp})

def post(self, request, id=None):

ahp\_ob = Alternatif.objects.get(id=id)

form = NamaProdukForm(request.POST, instance=ahp\_ob)

if form.is\_valid():

ahp=form.save(commit=False)

produk\_a=form.cleaned\_data['produk\_a']

produk\_b=form.cleaned\_data['produk\_b']

produk\_c=form.cleaned\_data['produk\_c']

produk=[]

produk.append(produk\_a),produk.append(produk\_b)

produk.append(produk\_c)

ahp.nama\_product= json.dumps(produk)

ahp.save()

messages.success(request,'Input nama produk telah berhasil')

return HttpResponseRedirect(reverse('input-ahp', kwargs={'id':ahp.id}))

else:

messages.error(request,'Ada kesalahan saaat input nama produk, mohon ulangi lagi')

return render(request,self.template\_name,{'form':form})