**UAS**

**APLIKASI CLICKER,SOCKET.IO**

****

**Disusun Oleh :**

**IRMA NUUR ROCHMAH 18720251001**

**PENDIDIKAN TEKNIK ELEKTRONIKA DAN INFORMATIKA**

**PASCASARJANA**

**UNIVERSITAS NEGERI YOGYAKARTA**

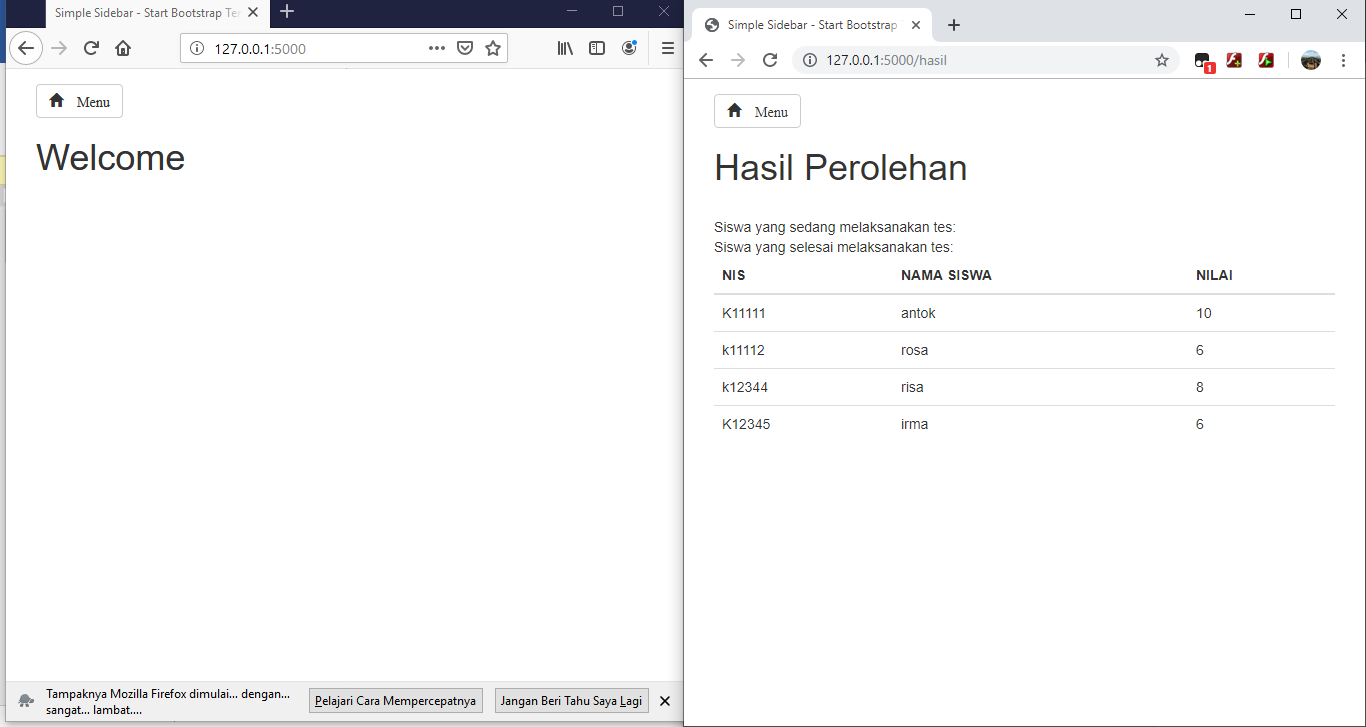
**Th. 2019**

1. **PEMBUATAN APLIKASI KUIS SEDERHANA**

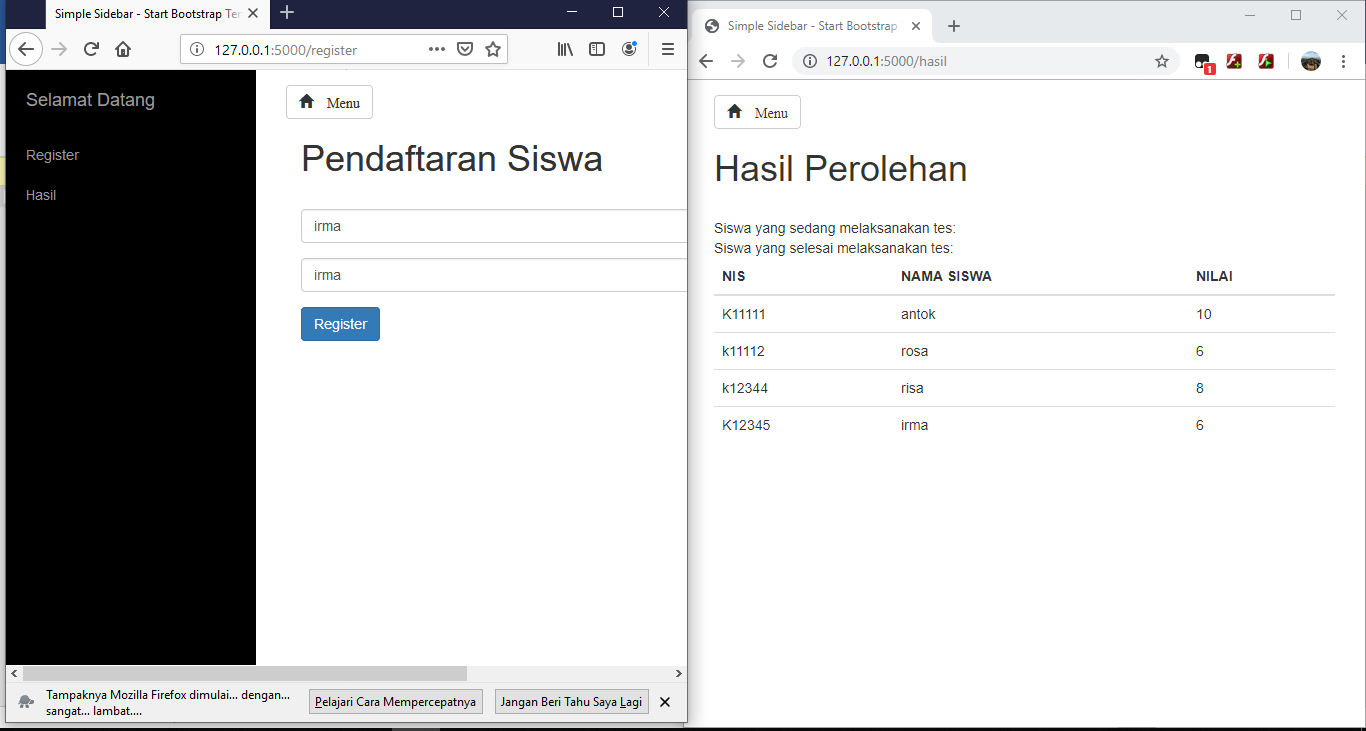
Clicker merupakan suatu cara untuk mempermudah kita dalam melakukan click. Clicker biasanya dibuat secara otomatis. Clicker menghasilkan input yang dapat direkam sebelumnya dan dapat diatur penggunaannya. Salah satu auto clicker yaitu ketika kita mengunjungi website kemudian muncul iklan. Kita dapat juga mensetting auto click ketika kita sedang flash sale. Karena saat flash sale membutuhkan kecepatan dalam mengklik, tentu saja jaringan dan keberuntungan juga berpengaruh.

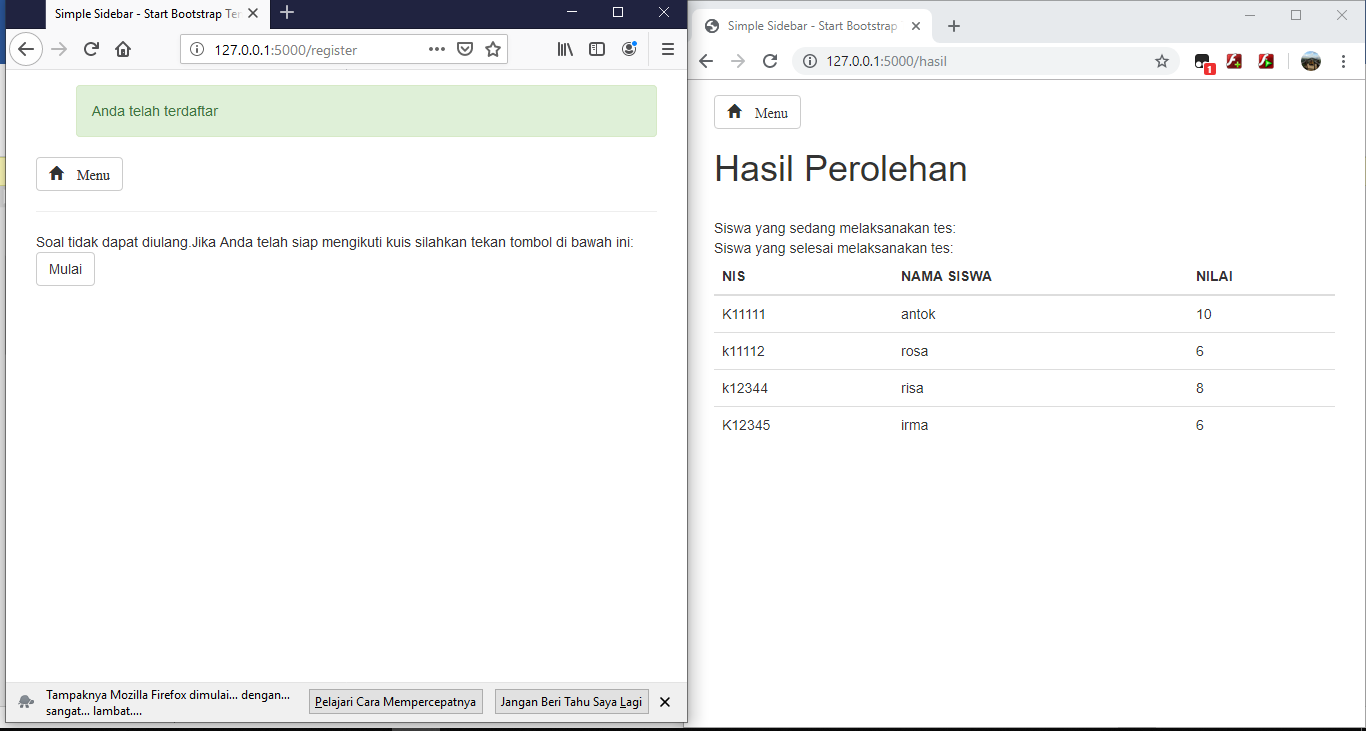
Aplikasi yang saya buat merupakan sebuah kuis sederhana yang terdiri dari 5 soal untuk siswa sekolah dasar. Siswa sekolah dasar dapat mengikuti kuis secara berulang. Nantinya nilai akan muncul di halaman hasil. Guru dapat memonitor berapa banyak siswa yang sudah mulai ikut tes, yang sudah selesai dan skor yang didapatkan oleh siswa. Bukan hanya hanya guru namun siswa juga dapat langsung melihatnya.

1. **ALUR**

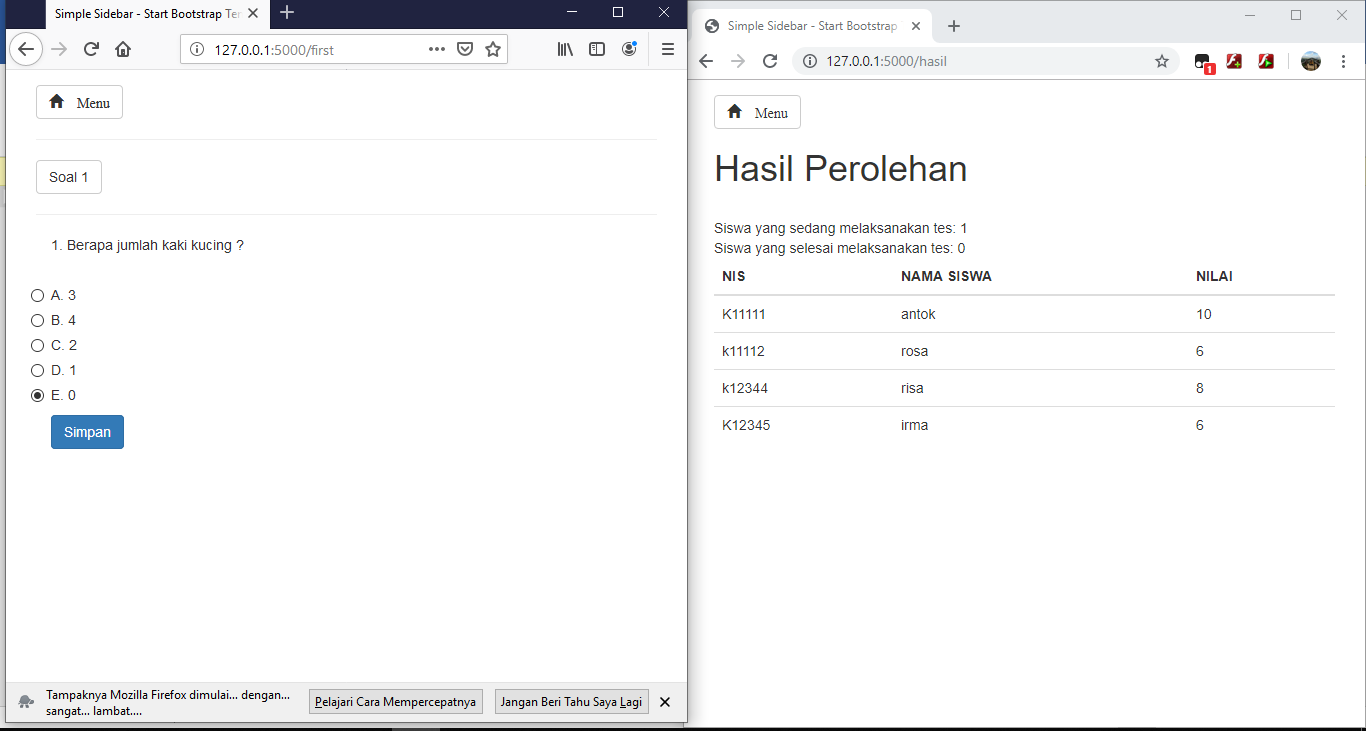


Klik tombol menu untuk menampilkan menu, kemudian pilih register. Baik sudah terdaftar atau belum tetap melakukan register untuk dapat masuk mengikuti kuis.

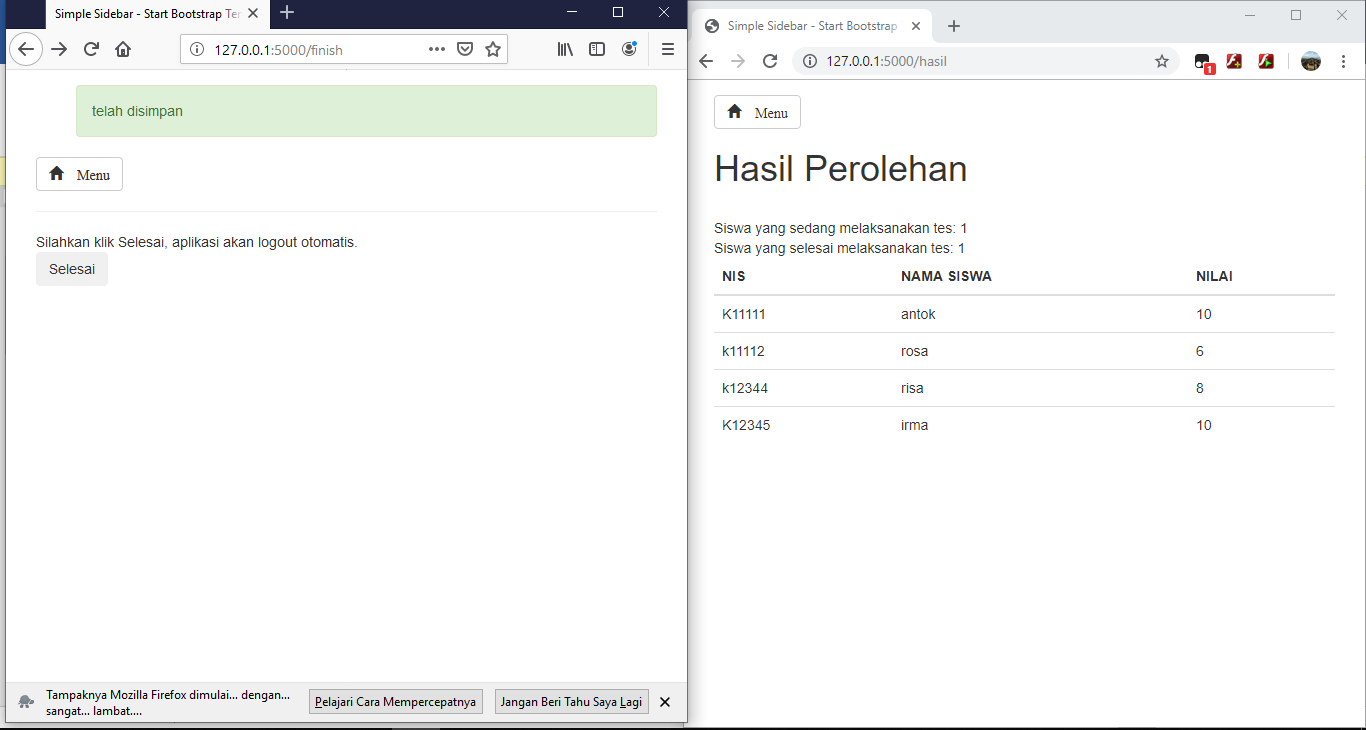




Setelah di klik tombol mulai akan muncul pada browser sebelahnya (ini berguna untuk guru yang ingin memantau) berapa siswa yang telah memulai kuis.



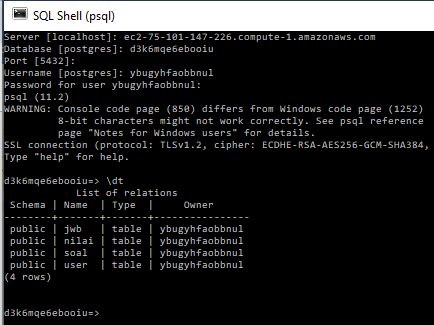
Setelah selesai mengikuti kuis, siswa menekan tombo selesai.

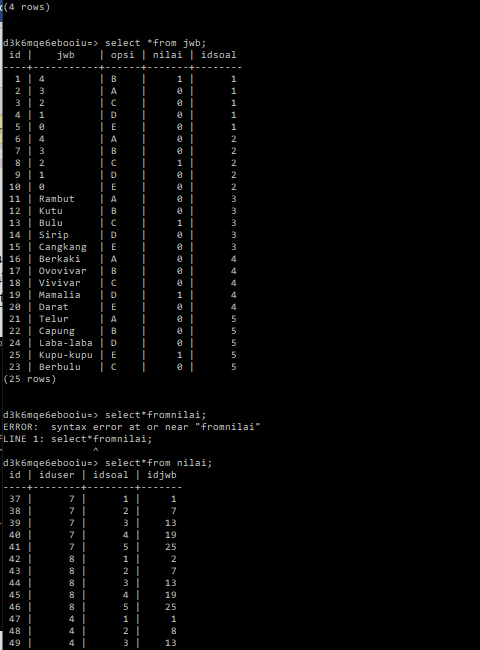


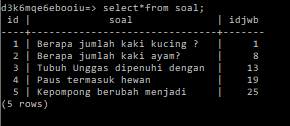
Maka browser sebelah akan menampilkan bahwa ada satu siswa yang telah selesai. Nilai diupdate terus selama beberapa detik. Setelah logout siswa dapat mengikuti kuis lagi dengan syarat harus melakukan tes sampai selesai. Agar nilai yang didapat akurat.

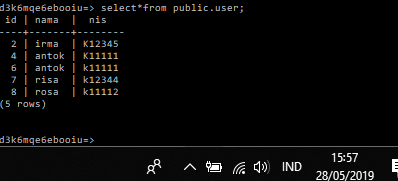
1. **SOURCE CODE**

Pada aplikasi ini saya membuat tabel user, nilai, jwb, dan soal. Masing-masing tabel saya relasikan. Sehingga ketika mengubah isi suatu baris harus disesuaikan dengan ketentuan pada tabel lainnya. Ini akibat penggunaan relasi foreign\_key.

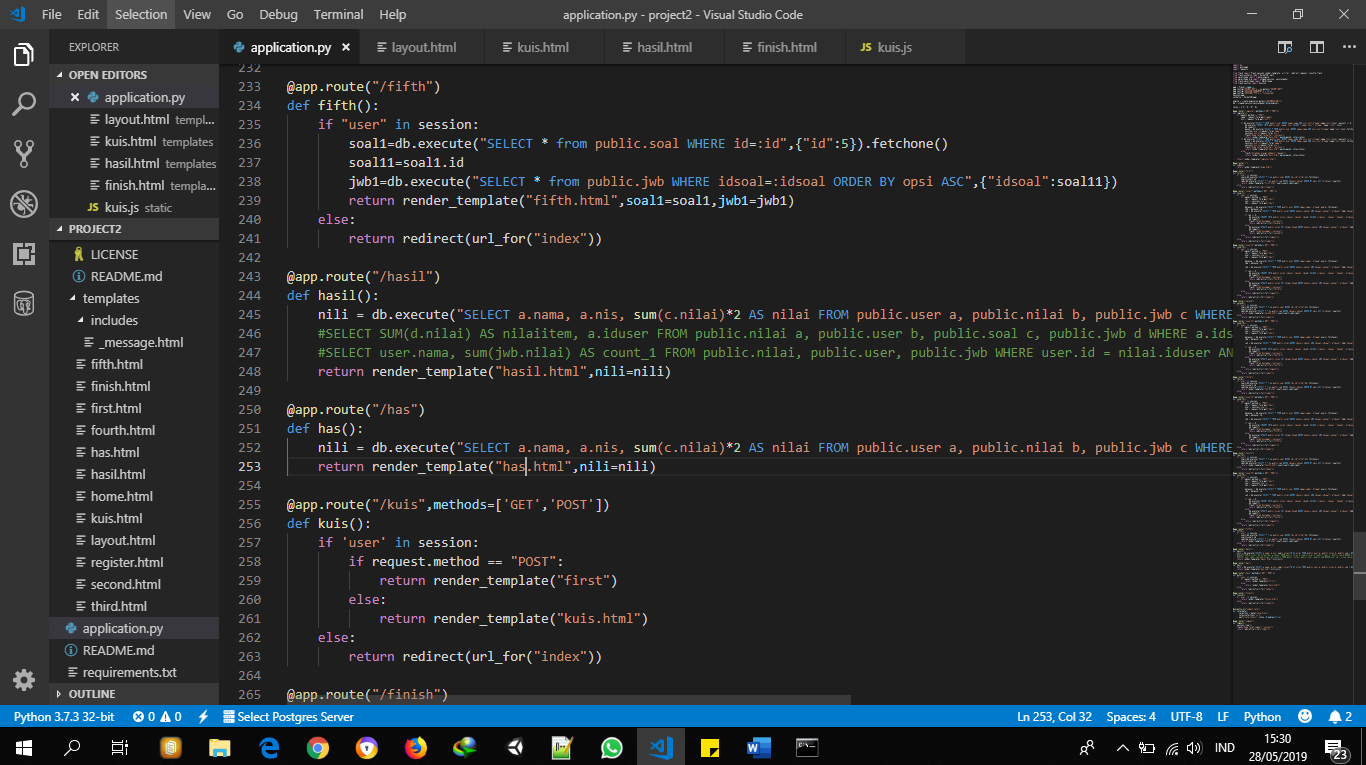








Berikut codingan pada python:



Pada gambar diatas dapat dilihat pada baris explorer, project2, template yang saya buat ada 12 html dan ada 1 html di dalam include, digunakan untuk memanggil message. Script utama yaitu application.py. Saya juga menyelipkan CSS, JS, dan sebagainya milik bootstrap di dalam folder static. Di dalam folder static juga terdapat kuis.js yang dalamnya ada socket.io.

Berikut script pada application.py:

import os

import psycopg2

import requests

from flask import Flask,session,render\_template, url\_for, redirect,request, jsonify,flash

from flask\_socketio import SocketIO, emit

from sqlalchemy import create\_engine

from sqlalchemy.orm import scoped\_session, sessionmaker

from flask\_sqlalchemy import SQLAlchemy

from flask\_session import Session

app = Flask(\_\_name\_\_)

app.config["SECRET\_KEY"] = os.getenv("SECRET\_KEY")

app.config["SESSION\_PERMANENT"] = False

app.config["SESSION\_TYPE"] = "filesystem"

Session(app)

socketio = SocketIO(app)

engine = create\_engine(os.getenv("DATABASE\_URL"))

db = scoped\_session(sessionmaker(bind=engine))

votes = {"A": 0, "B": 0}

@app.route("/register",methods=["GET","POST"])

def register():

if request.method == "POST":

nama = request.form.get("nama")

nis = request.form.get("nis")

if db.execute("SELECT \* FROM public.user WHERE nama=:nama AND nis=:nis",{"nama":nama,"nis":nis}).rowcount == 0:

db.execute("INSERT INTO public.user (nama, nis) VALUES (:nama,:nis)", {"nama": nama, "nis": nis})

db.commit()

masuk = db.execute("SELECT \* FROM public.user WHERE nama=:nama AND nis=:nis",{"nama":nama,"nis":nis}).fetchone()

session['user']=request.form['nama']

session['nis']=request.form['nis']

flash("Anda telah terdaftar","success")

return render\_template("kuis.html",masuk=masuk, votes=votes)

if db.execute("SELECT \* FROM public.user WHERE nama=:nama AND nis=:nis",{"nama":nama,"nis":nis}).rowcount == 1:

masuk = db.execute("SELECT \* FROM public.user WHERE nama=:nama AND nis=:nis",{"nama":nama,"nis":nis}).fetchone()

session['user']=request.form['nama']

session['nis']=request.form['nis']

flash("Anda telah terdaftar","success")

return render\_template("kuis.html",masuk=masuk, votes=votes)

else:

flash("Silahkan ulangi kembali","danger")

return render\_template("kuis.html",masuk=masuk, votes=votes)

return render\_template("register.html")

@app.route("/")

def index():

return render\_template("home.html")

@app.route("/first")

def first():

if "user" in session:

soal1=db.execute("SELECT \* from public.soal WHERE id=:id",{"id":1}).fetchone()

soal11=soal1.id

jwb1=db.execute("SELECT \* from public.jwb WHERE idsoal=:idsoal ORDER BY opsi ASC",{"idsoal":soal11})

return render\_template("first.html",soal1=soal1,jwb1=jwb1)

else:

return redirect(url\_for("index"))

@app.route("/insert",methods=['GET','POST'])

def insert():

if 'user' in session:

if request.method == "POST":

idj = request.form.get("idj")

user = session['user']

ids = request.form.get("ids")

datauser = db.execute("SELECT \* FROM public.user WHERE nama=:nama", {"nama":user}).fetchone()

idd = datauser.id

cek = db.execute("SELECT \* FROM public.nilai WHERE iduser=:iduser AND idsoal=:idsoal", {"iduser":idd,"idsoal":ids}).rowcount

if cek == 0:

db.execute("INSERT INTO public.nilai (iduser, idsoal, idjwb) VALUES (:iduser, :idsoal, :idjwb)", {"iduser":idd,"idsoal":ids, "idjwb":idj})

db.commit()

flash("telah disimpan","success")

return redirect(url\_for("second"))

else:

db.execute("UPDATE public.nilai SET idjwb=:idjwb WHERE iduser=:iduser AND idsoal=:idsoal", {"iduser":idd,"idsoal":ids, "idjwb":idj})

db.commit()

flash("telah disimpan","success")

return redirect(url\_for("second"))

else:

return redirect(url\_for("logout"))

else:

return redirect(url\_for("index"))

@app.route("/insert2",methods=['GET','POST'])

def insert2():

if 'user' in session:

if request.method == "POST":

idj = request.form.get("idj")

user = session['user']

ids = request.form.get("ids")

datauser = db.execute("SELECT \* FROM public.user WHERE nama=:nama", {"nama":user}).fetchone()

idd = datauser.id

cek = db.execute("SELECT \* FROM public.nilai WHERE iduser=:iduser AND idsoal=:idsoal", {"iduser":idd,"idsoal":ids}).rowcount

if cek == 0:

db.execute("INSERT INTO public.nilai (iduser, idsoal, idjwb) VALUES (:iduser, :idsoal, :idjwb)", {"iduser":idd,"idsoal":ids, "idjwb":idj})

db.commit()

flash("telah disimpan","success")

return redirect(url\_for("third"))

else:

db.execute("UPDATE public.nilai SET idjwb=:idjwb WHERE iduser=:iduser AND idsoal=:idsoal", {"iduser":idd,"idsoal":ids, "idjwb":idj})

db.commit()

flash("telah disimpan","success")

return redirect(url\_for("third"))

else:

return redirect(url\_for("logout"))

else:

return redirect(url\_for("index"))

@app.route("/second")

def second():

if "user" in session:

soal1=db.execute("SELECT \* from public.soal WHERE id=:id",{"id":2}).fetchone()

soal11=soal1.id

jwb1=db.execute("SELECT \* from public.jwb WHERE idsoal=:idsoal ORDER BY opsi ASC",{"idsoal":soal11})

return render\_template("second.html",soal1=soal1,jwb1=jwb1)

else:

return redirect(url\_for("index"))

@app.route("/insert3",methods=['GET','POST'])

def insert3():

if 'user' in session:

if request.method == "POST":

idj = request.form.get("idj")

user = session['user']

ids = request.form.get("ids")

datauser = db.execute("SELECT \* FROM public.user WHERE nama=:nama", {"nama":user}).fetchone()

idd = datauser.id

cek = db.execute("SELECT \* FROM public.nilai WHERE iduser=:iduser AND idsoal=:idsoal", {"iduser":idd,"idsoal":ids}).rowcount

if cek == 0:

db.execute("INSERT INTO public.nilai (iduser, idsoal, idjwb) VALUES (:iduser, :idsoal, :idjwb)", {"iduser":idd,"idsoal":ids, "idjwb":idj})

db.commit()

flash("telah disimpan","success")

return redirect(url\_for("fourth"))

else:

db.execute("UPDATE public.nilai SET idjwb=:idjwb WHERE iduser=:iduser AND idsoal=:idsoal", {"iduser":idd,"idsoal":ids, "idjwb":idj})

db.commit()

flash("telah disimpan","success")

return redirect(url\_for("fourth"))

else:

return redirect(url\_for("logout"))

else:

return redirect(url\_for("index"))

@app.route("/third")

def third():

if "user" in session:

soal1=db.execute("SELECT \* from public.soal WHERE id=:id",{"id":3}).fetchone()

soal11=soal1.id

jwb1=db.execute("SELECT \* from public.jwb WHERE idsoal=:idsoal ORDER BY opsi ASC",{"idsoal":soal11})

return render\_template("third.html",soal1=soal1,jwb1=jwb1)

else:

return redirect(url\_for("index"))

@app.route("/insert4",methods=['GET','POST'])

def insert4():

if 'user' in session:

if request.method == "POST":

idj = request.form.get("idj")

user = session['user']

ids = request.form.get("ids")

datauser = db.execute("SELECT \* FROM public.user WHERE nama=:nama", {"nama":user}).fetchone()

idd = datauser.id

cek = db.execute("SELECT \* FROM public.nilai WHERE iduser=:iduser AND idsoal=:idsoal", {"iduser":idd,"idsoal":ids}).rowcount

if cek == 0:

db.execute("INSERT INTO public.nilai (iduser, idsoal, idjwb) VALUES (:iduser, :idsoal, :idjwb)", {"iduser":idd,"idsoal":ids, "idjwb":idj})

db.commit()

flash("telah disimpan","success")

return redirect(url\_for("fifth"))

else:

db.execute("UPDATE public.nilai SET idjwb=:idjwb WHERE iduser=:iduser AND idsoal=:idsoal", {"iduser":idd,"idsoal":ids, "idjwb":idj})

db.commit()

flash("telah disimpan","success")

return redirect(url\_for("fifth"))

else:

return redirect(url\_for("logout"))

else:

return redirect(url\_for("index"))

@app.route("/fourth")

def fourth():

if "user" in session:

soal1=db.execute("SELECT \* from public.soal WHERE id=:id",{"id":4}).fetchone()

soal11=soal1.id

jwb1=db.execute("SELECT \* from public.jwb WHERE idsoal=:idsoal ORDER BY opsi ASC",{"idsoal":soal11})

return render\_template("fourth.html",soal1=soal1,jwb1=jwb1)

else:

return redirect(url\_for("index"))

@app.route("/insert5",methods=['GET','POST'])

def insert5():

if 'user' in session:

if request.method == "POST":

idj = request.form.get("idj")

user = session['user']

ids = request.form.get("ids")

datauser = db.execute("SELECT \* FROM public.user WHERE nama=:nama", {"nama":user}).fetchone()

idd = datauser.id

cek = db.execute("SELECT \* FROM public.nilai WHERE iduser=:iduser AND idsoal=:idsoal", {"iduser":idd,"idsoal":ids}).rowcount

if cek == 0:

db.execute("INSERT INTO public.nilai (iduser, idsoal, idjwb) VALUES (:iduser, :idsoal, :idjwb)", {"iduser":idd,"idsoal":ids, "idjwb":idj})

db.commit()

flash("telah disimpan","success")

return redirect(url\_for("finish"))

else:

db.execute("UPDATE public.nilai SET idjwb=:idjwb WHERE iduser=:iduser AND idsoal=:idsoal", {"iduser":idd,"idsoal":ids, "idjwb":idj})

db.commit()

flash("telah disimpan","success")

return redirect(url\_for("finish"))

else:

return redirect(url\_for("logout"))

else:

return redirect(url\_for("index"))

@app.route("/fifth")

def fifth():

if "user" in session:

soal1=db.execute("SELECT \* from public.soal WHERE id=:id",{"id":5}).fetchone()

soal11=soal1.id

jwb1=db.execute("SELECT \* from public.jwb WHERE idsoal=:idsoal ORDER BY opsi ASC",{"idsoal":soal11})

return render\_template("fifth.html",soal1=soal1,jwb1=jwb1)

else:

return redirect(url\_for("index"))

@app.route("/hasil")

def hasil():

nili = db.execute("SELECT a.nama, a.nis, sum(c.nilai)\*2 AS nilai FROM public.user a, public.nilai b, public.jwb c WHERE b.iduser=a.id AND b.idjwb = c.id Group By a.id Order by a.nis ASC; ").fetchall()

#SELECT SUM(d.nilai) AS nilaiitem, a.iduser FROM public.nilai a, public.user b, public.soal c, public.jwb d WHERE a.idsoal=:c.id and a.idjwb=:d.id group by a.iduser

#SELECT user.nama, sum(jwb.nilai) AS count\_1 FROM public.nilai, public.user, public.jwb WHERE user.id = nilai.iduser AND nilai.idjwb = jwb.id GROUP BY nilai.iduser

return render\_template("hasil.html",nili=nili)

@app.route("/has")

def has():

nili = db.execute("SELECT a.nama, a.nis, sum(c.nilai)\*2 AS nilai FROM public.user a, public.nilai b, public.jwb c WHERE b.iduser=a.id AND b.idjwb = c.id Group By a.id Order by a.nis ASC; ").fetchall()

return render\_template("has.html",nili=nili)

@app.route("/kuis",methods=['GET','POST'])

def kuis():

if 'user' in session:

if request.method == "POST":

return render\_template("first")

else:

return render\_template("kuis.html")

else:

return redirect(url\_for("index"))

@app.route("/finish")

def finish():

if 'user' in session:

return render\_template("finish.html")

else:

return redirect(url\_for("index"))

@socketio.on("submit vote")

def vote(data):

selection = data["selection"]

votes[selection] += 1

emit("vote totals", votes, broadcast=True)

@app.route("/logout")

def logout():

session.clear()

flash("Anda telah logout","success")

return redirect(url\_for("index"))

kuis.js:

document.addEventListener('DOMContentLoaded', () => {

var socket = io.connect(location.protocol + '//' + document.domain + ':' + location.port);

socket.on('connect', () => {

document.querySelectorAll('button').forEach(button => {

button.onclick = () => {

const selection = button.dataset.vote;

socket.emit('submit vote', {'selection': selection});

};

});

});

socket.on('vote totals', data => {

document.querySelector('#A').innerHTML = data.A;

document.querySelector('#B').innerHTML = data.B;

});

});

Hasil.html:

{% extends "layout.html" %}

{% block head %}

<script type="text/javascript" src="//cdnjs.cloudflare.com/ajax/libs/socket.io/1.3.6/socket.io.min.js"></script>

<script src="{{ url\_for('static',filename='kuis.js') }}"></script>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.3.0/jquery.min.js" type="text/javascript"></script>

<script type="text/javascript">

//digunakan untuk merefresh hasil skor

var auto\_refresh = setInterval(

function () {

$(".table").load("/has ").fadeIn("slow");

}, 10000);

</script>

{% endblock%}

{% block body %}

<h1> Hasil Perolehan </h1>

<br/>

<div>Siswa yang sedang melaksanakan tes: <span id="A"></span></div>

<div>Siswa yang selesai melaksanakan tes: <span id="B"></span></div>

<table class="table">

<thead>

<tr>

<th scope="col">NIS</th>

<th scope="col">NAMA SISWA</th>

<th scope="col">NILAI</th>

</tr>

</thead>

<tbody>

{% for nilai in nili%}

<tr>

<td scope="col">{{ nilai.nis }}</td>

<td scope="col">{{ nilai.nama }}</td>

<td scope="col">{{ nilai.nilai }}</td>

</tr>

{% endfor %}

</tbody>

</table>

{% endblock%}