LAMPIRAN

Lampiran-1 *Source code* app.py

```
from flask import Flask, render_template , request , jsonify , g , redirect ,
        request , session , url_for
from PIL import Image
import os , io , sys
import numpy as np
import cv2
import base64
from yolo_detection_images import runModel

class User:
    def __init__(self, id, username, password):
        self.id = id
        self.username = username
        self.password = password

    def __repr__(self):
        return f'<User: {self.username}>'

users = []
users.append(User(id=1, username='user', password='user'))

app = Flask(__name__)
app.secret_key = 'kuncirahasia'

@app.before_request
def before_request():
    g.user = None

    if 'user_id' in session:
        user = [x for x in users if x.id == session['user_id']][0]
        g.user = user


@app.route('/Login', methods=['GET', 'POST'])
def Login():
    if request.method == 'POST':
        session.pop('user_id', None)

        username = request.form['username']
        password = request.form['password']
```

```
        user = [x for x in users if x.username == username][0]
        if user and user.password == password:
            session['user_id'] = user.id
            return redirect(url_for('HalamanUtama'))

        return redirect(url_for('Login'))

    return render_template('Login.html')

@app.route('/HalamanUtama')
def HalamanUtama():
    if not g.user:
        return redirect(url_for('Login'))

    return render_template('HalamanUtama.html')
@app.route('/Deteksi')
def Deteksi():

    return render_template('Deteksi.html')
@app.route('/Perbandingan')
def Perbandingan():

    return render_template('Perbandingan.html')

@app.route('/detectObject' , methods=['POST'])
def mask_image():
        # print(request.files , file=sys.stderr)
        file = request.files['image'].read() ## byte file
        npimg = np.fromstring(file, np.uint8)
        img = cv2.imdecode(npimg,cv2.IMREAD_COLOR)

        img = runModel(img)

        img = Image.fromarray(img.astype("uint8"))
        rawBytes = io.BytesIO()
        img.save(rawBytes, "JPEG")
        rawBytes.seek(0)
        img_base64 = base64.b64encode(rawBytes.read())
        return jsonify({'status':str(img_base64)})


@app.route('/test' , methods=['GET','POST'])
def test():
        print("log: got at test" , file=sys.stderr)
        return jsonify({'status':'succces'})
```

```
@app.after_request
def after_request(response):
    print("log: setting cors" , file = sys.stderr)
    response.headers.add('Access-Control-Allow-Origin', '*')
    response.headers.add('Access-Control-Allow-Headers', 'Content-
        Type,Authorization')
    response.headers.add('Access-Control-Allow-Methods',
        'GET,PUT,POST,DELETE')
    return response


if __name__ == '__main__':
        app.run(debug = True)
```

Lampiran-2 *Source code* yolo_detection_images.py

```
import numpy as np
import argparse
import time
import cv2
import os

confthres=0.5
nmsthres=0.5
yolo_path="./"

def get_labels(labels_path):
    lpath=os.path.sep.join([yolo_path, labels_path])
    LABELS = open(lpath).read().strip().split("\n")
    return LABELS

def get_colors(LABELS):
    np.random.seed(42)
    COLORS = np.random.randint(0, 255, size=(len(LABELS),
3),dtype="uint8")
    return COLORS

def get_weights(weights_path):
    weightsPath = os.path.sep.join([yolo_path, weights_path])
    return weightsPath

def get_config(config_path):
```

```
    configPath = os.path.sep.join([yolo_path, config_path])
    return configPath

def load_model(configpath,weightspath):
    print("[INFO] load YOLO dari disk...")
    net = cv2.dnn.readNetFromDarknet(configpath, weightspath)
    return net


def get_predection(image,net,LABELS,COLORS):
    (H, W) = image.shape[:2]

    ln = net.getLayerNames()
    ln = [ln[i[0] - 1] for i in net.getUnconnectedOutLayers()]

    blob = cv2.dnn.blobFromImage(image, 1 / 255.0, (416, 416),
                                 swapRB=True, crop=False)
    net.setInput(blob)
    start = time.time()
    layerOutputs = net.forward(ln)
    print(layerOutputs)
    end = time.time()

    print("[INFO] YOLO took {:.6f} seconds".format(end - start))

    class_ids = []
    confidences = []
    boxes = []

    for output in layerOutputs:
        for detection in output:
            scores = detection[5:]
            class_id = np.argmax(scores)
            confidence = scores[class_id]
            if confidence > confthres:


                box = detection[0:4] * np.array([W, H, W, H])
                (center_x, center_y, width, height) = box.astype("int")

                x = int(center_x - (width / 2))
                y = int(center_y - (height / 2))

                boxes.append([x, y, int(width), int(height)])
                confidences.append(float(confidence))
                class_ids.append(class_id)
```

```
    indexes = cv2.dnn.NMSBoxes(boxes, confidences, confthres,
                           nmsthres)

    print(indexes)
    font = cv2.FONT_HERSHEY_PLAIN
    for i in range(len(boxes)):
        if i in indexes:
            x, y, w, h = boxes[i]
            label = str(LABELS[class_ids[i]])
            color = [int(c) for c in COLORS[class_ids[i]]]
            confidence = str(round(confidences[i]*100, 2))
            cv2.rectangle(image, (x, y), (x + w, y + h), color, 20)
            cv2.putText(image, label + ":" + confidence + "%", (x, y + -10),
font, 12, color, 15)
    return image

def runModel(image):
    # image = cv2.imread(img)
    labelsPath="./obj.names"
    cfgpath="cfg/yolov4-tiny-custom.cfg"
    wpath="yolov4-tiny-custom.weights"
    Lables=get_labels(labelsPath)
    CFG=get_config(cfgpath)
    Weights=get_weights(wpath)
    nets=load_model(CFG,Weights)
    Colors=get_colors(Lables)
    res=get_predection(image,nets,Lables,Colors)
    return res
```

Lampiran-3 *Source code* index.css

```
 html {
   text-align: center;
   background-color: #282c34;
   color: white;
 }

 .App-header {
   background-color: #282c34;
   min-height: 100vh;
```

```
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  font-size: calc(10px + 2vmin);
  color: white;
}

.App-link {
  color: #61dafb;
}



.pre_img{
  height: 600px;
  width: 600px;
  /* background-color: white; */
}

input[type=file]{
  background-color: #4CAF50;
  border: none;
  color: white;
  padding: 15px 32px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
  box-shadow: 0 8px 16px 0 rgba(0,0,0,0.2), 0 6px 20px 0 rgba(0,0,0,0.19);
  margin: 40px 40px;
  /* display: none; */

}

#pre{
  background-color: #4CAF50; /* Green */
  border: none;
  color: white;
  padding: 15px 32px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  font-size: 16px;
  box-shadow: 0 8px 16px 0 rgba(0,0,0,0.2), 0 6px 20px 0 rgba(0,0,0,0.19);
  margin: 10px 40px;
```

```
}

p{
  margin: 50px;
  font-size: 72px;
}
```

Lampiran-4 *Source code* index.js

```
window.onload = () => {
        $('#sendbutton').click(() => {
         imagebox = $('#imagebox')
         input = $('#imageinput')[0]
         if(input.files && input.files[0])
         {
                let formData = new FormData();
                formData.append('image' , input.files[0]);
                $.ajax({
                        url: "http://localhost:5000/detectObject",
                        type:"POST",
                        data: formData,
                        cache: false,
                        processData:false,
                        contentType:false,
                        error: function(data){
                                console.log("upload error" , data);

         console.log(data.getAllResponseHeaders());
                        },
                        success: function(data){
                                console.log(data);
                                bytestring = data['status']
                                image = bytestring.split('\'')[1]
                                imagebox.attr('src' ,
        ''+image)
                        }
                });
         }
        });
};
```

```
function readUrl(input){
        imagebox = $('#imagebox')
        console.log("evoked readUrl")
        if(input.files && input.files[0]){
         let reader = new FileReader();
         reader.onload = function(e){
                // console.log(e)

                imagebox.attr('src',e.target.result);
                imagebox.height(500);
                imagebox.width(800);
         }
         reader.readAsDataURL(input.files[0]);
        }
}
```

Lampiran-5 *Source code* Login.html

```
<!DOCTYPE html>
<html>

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Login</title>
  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bulma@0.8.0/css/bulma.min.css">
  <script defer
src="https://use.fontawesome.com/releases/v5.3.1/js/all.js"></script>
</head>

<body>
  <section class="hero is-info is-fullheight">
    <div class="hero-body">
      <div class="container">
        <div class="columns is-centered">
          <div class="column is-half">
            <form method="POST">
              <div class="field">
                <label class="label">Username</label>
                <div class="control has-icons-left has-icons-right">
                  <input class="input" type="text" name="username"
```

```
placeholder="Username">
              <span class="icon is-small is-left">
                <i class="fas fa-user"></i>
              </span>
            </div>
          </div>
          <div class="field">
            <label class="label">Password</label>
            <p class="control has-icons-left">
              <input class="input" type="password" name="password"
placeholder="Password">
              <span class="icon is-small is-left">
                <i class="fas fa-lock"></i>
              </span>
            </p>
          </div>
          <div class="field">
            <p class="control">
              <button type="submit" class="button is-success">
                Login
              </button>
            </p>
          </div>
        </form>
      </div>
    </div>
  </div>
 </section>
</body>

</html>
```

Lampiran-6 *Source code* Deteksi.html

```
<!DOCTYPE html>
<html>
<head>
 <title>Aplikasi Pengujian Deteksi Objek</title>
 <link rel="stylesheet" type="text/css" href="../static/index.css">
 <script
                src="https://code.jquery.com/jquery-3.3.1.min.js"
                integrity="sha256-
```

```
FgpCb/KJQlLNfOu91ta32o/NMZxltwRo8QtmkMRdAu8="
                        crossorigin="anonymous"></script>
  <script type="text/javascript" src="../static/index.js"></script>
  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bulma@0.8.0/css/bulma.min.css">
  <script defer
src="https://use.fontawesome.com/releases/v5.3.1/js/all.js"></script>
</head>

<div class='App'>
   <header class='App-header'>
      <p>
        Halaman Pengujian Deteksi Objek
      </u>
      <div className="pre_img">
        <span><img id="imagebox" src=""></span>
      </div>
      <form>
        <input id="imageinput" type="file" name="image"
onchange="readUrl(this)">
      </form>
          <div class="field">
            <p class="control">
                      <button type="submit" class="button is-success" button
name="send" id = "sendbutton">
                        Deteksi Objek
                      </button>
 
              <button type="submit" class="button is-success"><a
style="color:white" href="{{ url_for ('Login') }}">
                Logout
              </button>
            </p>
            </div)

          </div>
  </header>
</div>
</body>
</html>
```