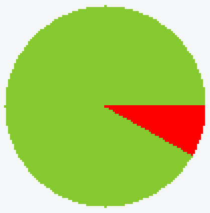
PLAGIARISM SCAN REPORT

Date	April 20, 2022
Exclude URL:	NO



Unique Content	92%
Plagiarized Content	8%
Paraphrased Plagiarism	0

Word Count	965
Records Found	10

CONTENT CHECKED FOR PLAGIARISM:

The web application aims at generating complaints or reports, where a specific user (citizen) can capture the pic of an area, which will be fed to a Deep learning model that has the ability to geocode, validate and track the potholes in that specific region captured. The feature has been achieved by training a YOLO v2 model for object tracking on multiple images as well as videos. The model uses a convolutional neural network. Users would have the amenity to view the damage on the roads using this application. A dynamic complaint to the authorities in the form of a report is generated which is shared with the closest authority(within 5000m radius) who can view and update this report. They can also view the reports generated by users within their municipal locality. An additional feature of filtering based on filters like search bar,etc are also enabled.

With this project one can now expect citizens to engage and take an active part in helping maintain the city infrastructure by using their mobiles to capture a pic of the nearby potholes and report it to the nearest municipality.
An easy to use and sustainable solution is now available for government officials as well to manage and act on citizen issues with transparency and a streamlined process when it comes to reports related to potholes and maintaining city streets.
For a person travelling in a highway, this is an extremely useful solution to avoid the upcoming potholes on his way by having a look at the google maps on where are the potholes present and what are their respective status.
Citizen features:

Create new report for creating a new report for the users to start reporting

My complaints dashboard to view the status of his complaint.

Route navigation to help users detect potholes in the respective route for which he wishes to travel.

Profile for viewing basic user details such as name, email,etc.

Sign in screen which uses the oauth google api for necessary authentications for security purposes.

Authority features

A Dashboard for displaying an analytics of user reports based on the status of severity. An option for also viewing reports based on filters.

An interactive Map region view to see the potholes in their region.

Manage users to view user details on the reported potholes and provide necessary updates.

Profile section to view basic details and number of reports approved, in progress or submitted.

Sign in screen which again uses google oauth apis and stand alone database to authorize authority logins for security purposes.

7. MYSQL DATABASE SERVER CONFIGURATION

Step 1 : Before the Database configuration, we build a schema of the actual database to be used for storing data from Authorities and Citizens.

Step 2: Follow the below commands to setup the above database schema in the remote instance configured in aws. Here we have already configured the above schema from local computer and dumped them into the remote instance

create database onspot;

sudo mysql -u root -p onspot < onspot>sql

use database onspot;

show tables;

Step 3: Now to test the above configuration, we add some dummy data using the following commands

insert into values ;

Step 4: After insertion of dummy data lets query the table for testing using the below command

select * from ;

9. APACHE2 WEB SERVER CONFIGURATION

Step 1: Install Apache server using the below command

Step 2: Make a directory under /var/www/html

cd /var/www/html

```
mkdir flask_api
cd flask_api
```

Step 3: Now copy the flask_api folder(in local machine) to written locally for api endpoints configuration to the remote instance

```
scp -r -i /https://github.com/kuluruvineeth/onspot. backend/flask_api/*
ubuntu@:/var/www/html/flask_api
```

Step 4: Now copy the darkflow_object_detection_model folder by creating the below repository tree under /var/www/html by following the steps 2 and 3 as shown above

Folder tree to construct : onspot/onspot_backend/backend/

Step 5: Now we need to setup the flask_api.wsgi file (Web Server gateway interface) for web server (Apache) - database interface configuration

Here are the contents of the wsgi file

```
import sys
sys.path.insert(0,'/var/www/html/flask_api')
from flask_api import app as application
```

Step 6: Enable the flask_api application using the 000-default.conf file. Refer to the command below

```
sudo vi /etc/apache2/sites-enabled/000-default.conf
ServerName www.onspot.click
ServerAdmin webmaster@localhost
DocumentRoot /var/www/html
WSGIDaemonProcess flask_api threads=5
WSGIScriptAlias / /var/www/html/flask_api/flask_api.wsgi
WSGIApplicationGroup %{GLOBAL}

WSGIProcessGroup flask_api
WSGIApplicationGroup %{GLOBAL}
Order deny,allow
Allow from all

ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined
```

Step 7: Now create the utility.py file under the flask_api folder using the below constants

```
# constants
DARKFLOW_PATH = '/var/www/html/onspot/onspot_backend/backend/darkflow/'
MODEL_CFG_PATH = '/var/www/html/onspot/onspot_backend/backend/darkflow_object_detection_model/cfg/yolov2-tiny-voc-1c.cfg'
PROTOBUF_PATH = '/var/www/html/onspot/onspot_backend/backend/darkflow_object_detection_model/built_graph/yolov2-tiny-voc-1c.pb'
META_FILE_PATH = '/var/www/html/onspot/onspot_backend/backend/darkflow_object_detection_model/built_graph/yolov2-tiny-voc-1c.meta'
DEFAULT_THRESHOLD = 0.2
LABEL = 'label'
LABEL_X = 'x'
LABEL_Y = 'y'
LABEL_TOP_LEFT = 'topleft'
LABEL_BOTTOM_RIGHT = 'bottomright'
BOX_OFFSET = 10
DOMAIN_NAME = 'https://www.onspot.click' # YOUR HTTPS DOMAIN NAME HERE
WEB_DIR_PATH = '/var/www/html'
IMG_UPLOADS_DIRECTORY = '/var/www/html/flask_api/static/'
IMG_UPLOADS_DISPLAY_URL = DOMAIN_NAME + '/flask_api/static/'
DEFAULT_FILE_TYPE = '.png'
# sample paths
SAMPLE_IMG_PATH = '/var/www/html/onspot/onspot_backend/backend/darkflow_object_detection_model/sample_img/000008.png'
SAMPLE_IMG_OUTPUT_PATH = '/var/www/html/onspot/onspot_backend/backend/darkflow_object_detection_model/sample_img/a.jpg'
SAMPLE_IMG_OUTPUT_DISPLAY_URL = DOMAIN_NAME + '/onspot/onspot_backend/backend/darkflow_object_detection_model/sample_img/a.jpg'
```

Step 8: Let us now configure the app_controller.wsgi file under the /onspot/onspot_backend/backend which acts as an interface between the Apache server and the flask api for rendering the endpoints.

```
#! /usr/bin/python
import sys
import logging
logging.basicConfig(stream=sys.stderr)
sys.path.insert(0,"/var/www/html/spothole/spothole_backend/backend/flask_api/")
# home points to the home.py file
from app_controller import app as application
application.secret_key = "somesecretsessionkey"
```

Step 9: Now, let us create the onspot.conf file for the api endpoints configuration from the flask_api folder

```
sudo vi /etc/apache2/sites-enabled/onspot.conf
ServerName www.onspot.click
ServerAdmin admin@www.onspot.click
WSGIScriptAlias / /var/www/html/onspot/onspot_backend/backend/app_controller.wsgi

Order allow,deny
Allow from all

ErrorLog ${APACHE_LOG_DIR}/error.log
LogLevel warn
CustomLog ${APACHE_LOG_DIR}/access.log combined
RewriteEngine on
RewriteCond %{SERVER_NAME} =www.onspot.click
RewriteRule ^ https://%{SERVER_NAME}%{REQUEST_URI} [END,NE,R=permanent]
```

Step 10: As we have configured only a development server , we need to use certbot in order to enhance it into a production server by creating an ssl certificate. Refer the below commands for the same.

```
--------certbot-------
sudo apt-get update
sudo apt-get install software-properties-common
sudo add-apt-repository ppa:certbot/certbot

sudo apt-get update

sudo apt-get install python3-certbot-apache

sudo certbot --apache -d www.onspot.click
```

MATCHED SOURCES:

stackoverflow.com - *2% Similar*Compare

https://stackoverflow.com/questions/24251898/flask-app-updat....

---

dzone.com - *2% Similar*Compare

https://dzone.com/articles/how-to-install-and-configure-apac....

---

www.digitalocean.com - *1% Similar*Compare

https://www.digitalocean.com/community/tutorials/how-to-depl....

---

apireference.groupdocs.com - *1% Similar*Compare

https://apireference.groupdocs.com/signature/net/groupdocs.s....

---

kelvinxu.github.io - *<1>Compare*

http://kelvinxu.github.io/projects/capgen.html

---

www.digitalocean.com - *<1>Compare*

https://www.digitalocean.com/community/questions/document-ro....

pastebin.com - *<1>Compare*

https://pastebin.com/yHXcusvT

github.com - *<1>Compare*

https://github.com/GrahamDumpleton/mod_wsgi/issues/737

github.com - *<1>Compare*

https://github.com/anieo/flask-blog/blob/master/flask.conf

docs.djangoproject.com - *<1>Compare*

https://docs.djangoproject.com/en/4.0/howto/deployment/wsgi/....