Google Dorking Findings Write-Up

I. Executive Summary

This investigation used targeted Google Dorking queries to identify publicly exposed directories on cognisun.net containing highly sensitive information. Discovered assets included API keys, plaintext database credentials, vulnerable dependencies, and exploitable code. These exposures present a critical (9–10/10) security risk, enabling potential unauthorized access, SQL injection, API abuse, and further exploitation. Immediate remediation is required—restrict directory access, rotate all credentials, and patch vulnerabilities to prevent compromise.

II. Introduction

Google Dorking is a method of using advanced Google search operators to locate information unintentionally exposed online. In this task, the goal is to find publicly exposed documents or directories that could reveal sensitive information. Queries such as `site:example.com filetype:pdf` or `intitle:"index of"` are used, and each finding is documented with the dork used and the link to the resource.

III. Dorking Methodology

The following details the Google dorks and techniques applied during this investigation.

- **Target Scope:**
  - Specific keywords/phrases: `API_KEY.txt`, `.env`
- **Dork Categories:**
  - intitle:`"index of" "API_KEY.txt"`, `"index of" ".env"`

IV. Findings

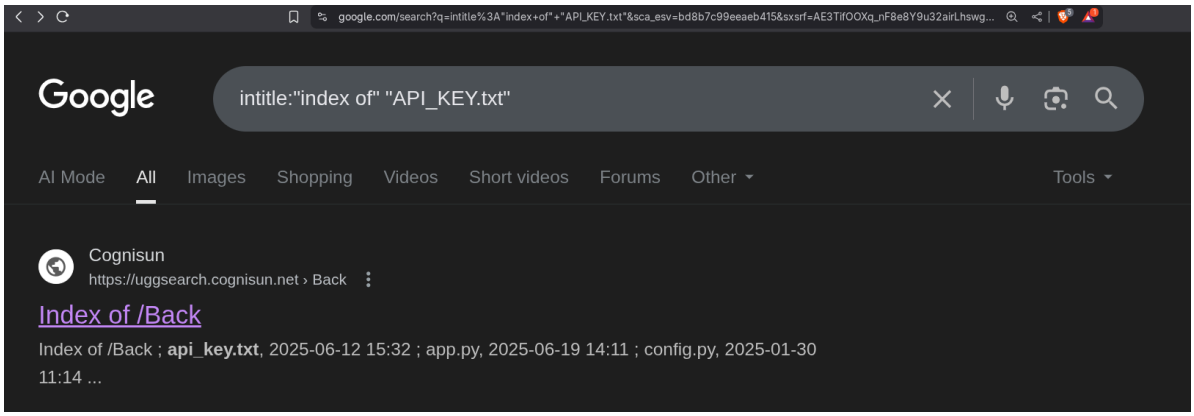
This section presents the detailed findings from the Google dorking investigation. For each significant finding, include the following information:

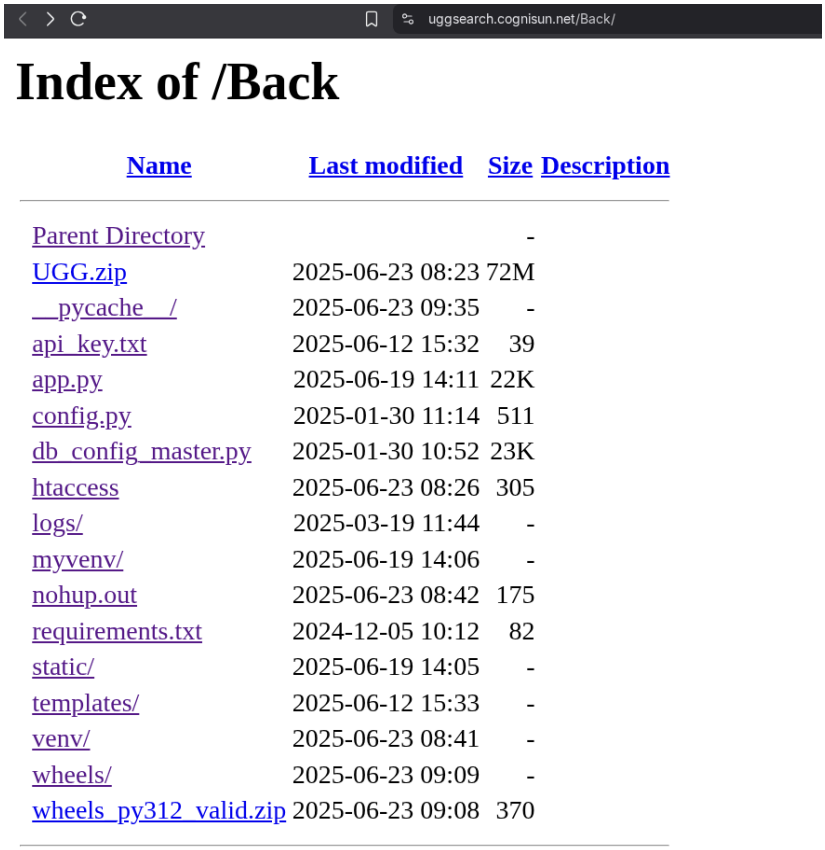
- **Dork Used:** `intitle:”index of" "API_KEY.txt"`
- **Description of Finding:** I’ve found /Back on cognisun.net and `found multiple directories with possible threat`
- **Example URL:** `https://uggsearch.cognisun.net/Back/`
- **Further findings:** I looked into the directories a bit deeper:
  - UGG.zip - I didn’t care to unzip it as I’m incapable of containing a potential malware(just incase)

    - __pychache__ - this directory contains 2 pycharm files from them i found that they are using mysql as their database
    - api_key.txt - this was the easiest found since the api key was in plain text
    - app.py - it is a Flask-based Python web application that acts as a Google Maps Places API data collection and reporting tool with user authentication, data storage, and Excel report generation
    - config.py - did not get much info from it but everything is related to os.getenv() which is a python function which get data from home environment variable
    - db_config_master.py - this was a gold mine, I got a lot of juicy data from it like the host ip, userid, password, name of database, etc
      There is also the risk of storing passwords as plain text and also has the sql injection vulnerability, and much more
    - htaccess - couldn't find any vulnerability
    - logs - empty dir
    - myenv/Lib/site-packages - contains all the packages used and found some outdated vulnerable packages
    - nohup - nohangup is a linux command used to keep a process running even after exiting the shell (but i thought the host is uing windows but this is a linux command so my best guess is the user is using windows OS with wsl)
    - requirements.txt - they are using these
        - Flask==2.0.1 → CVE-2023-30861
        - Werkzeug==2.0.1 → CVE-2024-34069
        - googlemaps==4.5.3 → no vulnerability found
        - mysql-connector-python==8.0.28 → CVE-2024-21090(ddos)
    - static - contains all javascripts and css and static/index_scripts.js has no input sanitization and potential XSS threat
    - template - basically a template
    - venv - couldn't find any vulnerability
    - wheels - empty dir
    - wheels_py312_valid.zip - didn't downloaded it

- **Severity/Impact:** critical 9/10 or 10/10 since this is a publicly exposed directory with direct access to authentication keys, database credentials, exploitable code, and vulnerable packages.

V. Figures/Screenshots



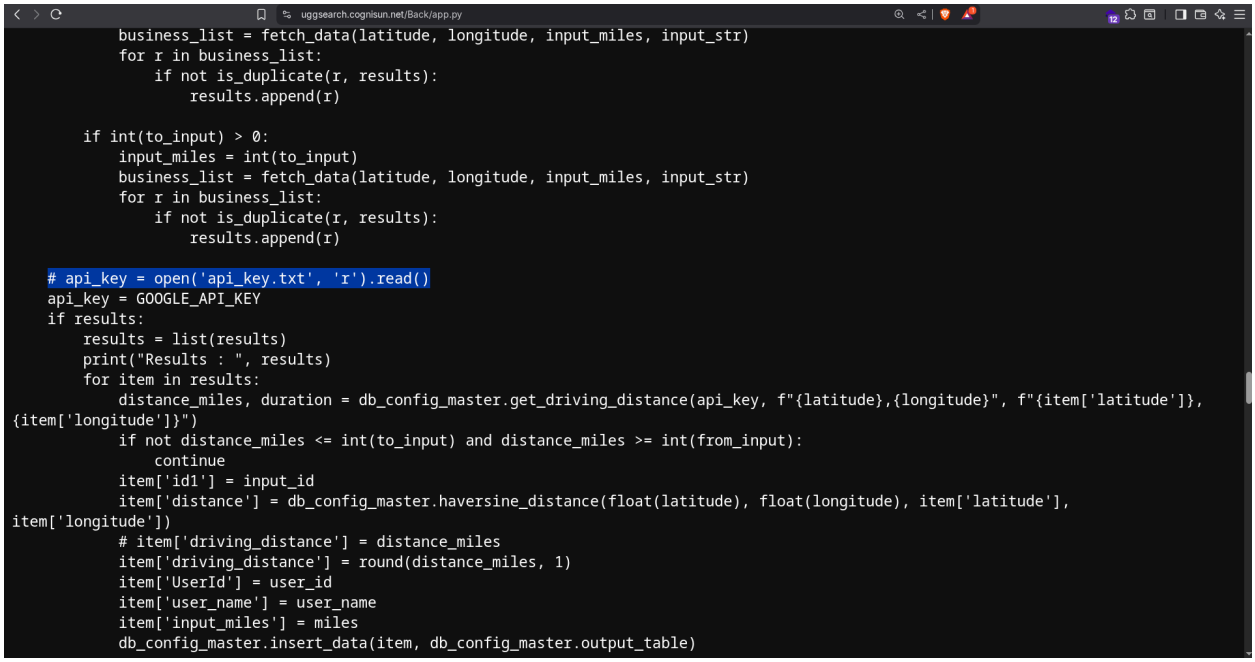


uggsearch.cognisun.net/Back/api_key.txt

```
AIzaSyA0rSf88TVQe8jv0E7ZXMhRahTvz-566KU
```

uggsearch.cognisun.net/Back/app.py

```
from datetime import date
import pandas as pd
from io import BytesIO
from flask import Flask, request, jsonify, render_template, redirect, url_for, session, send_file
from flask import Response
from werkzeug.wsgi import FileWrapper
import time
import googlemaps
import requests
import os
import sys
from config import *

sys.path.insert(0, os.path.dirname(__file__))

import db_config_master
import io
from functools import wraps
sys.stdout = io.TextIOWrapper(sys.stdout.buffer, encoding='utf-8')


app = Flask(__name__)

app.secret_key = 'abcd'
application = app


def is_logged_in():
    return 'username' in session
```


uggsearch.cognisun.net/Back/app.py
business_list = fetch_data(latitude, longitude, input_miles, input_str)
for r in business_list:
if not is_duplicate(r, results):
results.append(r)
if int(to_input) > 0:
input_miles = int(to_input)
business_list = fetch_data(latitude, longitude, input_miles, input_str)
for r in business_list:
if not is_duplicate(r, results):
results.append(r)
# api_key = open('api_key.txt', 'r').read()
api_key = GOOGLE_API_KEY
if results:
results = list(results)
print("Results : ", results)
for item in results:
distance_miles, duration = db_config_master.get_driving_distance(api_key, f"{latitude},{longitude}", f"{item['latitude']},
{item['longitude']}")
if not distance_miles <= int(to_input) and distance_miles >= int(from_input):
continue
item['id1'] = input_id
item['distance'] = db_config_master.haversine_distance(float(latitude), float(longitude), item['latitude'],
item['longitude'])
item['driving_distance'] = distance_miles
item['driving_distance'] = round(distance_miles, 1)
item['UserId'] = user_id
item['user_name'] = user_name
item['input_miles'] = miles
db_config_master.insert_data(item, db_config_master.output_table)

uggsearch.cognisun.net/Back/config.py

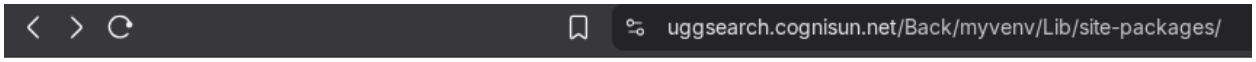
```
from dotenv import load_dotenv
import os

load_dotenv(override=True)

ENV = os.getenv('ENV')

DB_HOST = os.getenv('DB_HOST')
DB_USER = os.getenv('DB_USER')
DB_PASSWORD = os.getenv('DB_PASSWORD')

if ENV == 'UAT':
    DB_NAME = os.getenv('UAT_DB_NAME')
    PORT=os.getenv('UAT_PORT')
elif ENV == 'PROD':
    DB_NAME = os.getenv('PROD_DB_NAME')
    PORT=os.getenv('PROD_PORT')
else:
    print("Invalid Environment Variable")

GOOGLE_API_KEY = os.getenv('GOOGLE_API_KEY')

print("Port", PORT)
```

uggsearch.cognisun.net/Back/db_config_master.py

```
import math
from datetime import datetime
import requests
from scrapy import Selector
import mysql.connector
from mysql.connector import pooling
import time
import itertools
from config import *

# Database configuration
# dbconfig = {
#     "host": '103.211.218.87',
#     "user": 'root',
#     "password": 'bx9=UE;5I7WU',
#     "database": "UGGGoogleMapsUAT"
#     # "database": "UGGGoogleMapsProd"
# }


dbconfig = {
    "host":DB_HOST,
    "user":DB_USER,
    "password":DB_PASSWORD,
    "database":DB_NAME
}


# Connection pooling setup
pool = pooling.MySQLConnectionPool(pool_name="mypool", pool_size=30, **dbconfig)

input_table = "input_text"
output_table = "google_data"
google_data_buffer = "google_data_buffer"
```

uggsearch.cognisun.net/Back/db_config_master.py

```
            return True
        else:
            print("No IDs provided for deletion.")
            return False
    except Exception as err:
        print(f"Error deleting selected data from buffer: {err}")
        return False


def authenticate_user(username, password):
    try:
        conn = db_connect()
        cursor = conn.cursor(dictionary=True)
        cursor.execute(f"SELECT * FROM Users WHERE Name = '{username}' AND Password = '{password}'")
        user = cursor.fetchone()
        cursor.close()
        conn.close()
        return user
    except Exception as err:
        print(f"Error authenticating user: {err}")
        return None


def fetch_highways():
    try:
        conn = db_connect()
        cursor = conn.cursor()

        cursor.execute("SELECT DISTINCT highway FROM google_data WHERE highway IS NOT NULL;")
        highways = cursor.fetchall()

        cursor.close()
        conn.close()

        # Format the states for the response
```



Index of /Back/myvenv/Lib/site-packages

Name	Last modified	Size	Description
Parent Directory		-	
Flask-2.0.1.dist-info/	2025-06-19 14:07	-	
MarkupSafe-3.0.2.dis..>	2025-06-19 14:07	-	
OpenSSL/	2025-06-19 14:10	-	
Protego-0.4.0.dist-i..>	2025-06-19 14:09	-	
PyDispatcher-2.0.7.d..>	2025-06-19 14:09	-	
Werkzeug-2.0.1.dist-..>	2025-06-19 14:07	-	
__pycache__/	2025-06-19 14:09	-	
_cffi_backend.cp313-..>	2025-06-19 14:09	175K	
_distutils_hack/	2025-06-19 14:09	-	
attr/	2025-06-19 14:09	-	
attrs-25.3.0.dist-info/	2025-06-19 14:09	-	
attrs/	2025-06-19 14:09	-	
automat-25.4.16.dist..>	2025-06-19 14:09	-	
automat/	2025-06-19 14:09	-	
certifi-2025.6.15.di..>	2025-06-19 14:07	-	
certifi/	2025-06-19 14:07	-	
cffi-1.17.1.dist-info/	2025-06-19 14:09	-	
cffi/	2025-06-19 14:09	-	
charset_normalizer-3..>	2025-06-19 14:07	-	
charset_normalizer/	2025-06-19 14:07	-	
click-8.2.1.dist-info/	2025-06-19 14:07	-	
click/	2025-06-19 14:07	-	
colorama-0.4.6.dist-..>	2025-06-19 14:07	-	
colorama/	2025-06-19 14:07	-	
constantly-23.10.4.d..>	2025-06-19 14:09	-	
constantly/	2025-06-19 14:09	-	
cryptography-45.0.4...>	2025-06-19 14:10	-	
cryptography/	2025-06-19 14:10	-	
cssselect-1.3.0.dist..>	2025-06-19 14:09	-	

uggsearch.cognisun.net/Back/myvenv/pyvenv.cfg

```
home = C:\Python313
include-system-site-packages = false
version = 3.13.3
executable = C:\Python313\python.exe
command = C:\Python313\python.exe -m venv D:\Sagar\GitLab Projects\UGG\myvenv
```

uggsearch.cognisun.net/Back/requirements.txt

```
Flask==2.0.1
Werkzeug==2.0.1
googlemaps==4.5.3
mysql-connector-python==8.0.28
```

uggsearch.cognisun.net/Back/static/index_scripts.js

```
console.log("Index Script Loaded Started")
$(document).ready(function() {
    $('[data-toggle="tooltip"]').tooltip();

    // Initialize Google Maps
    initMap();

    // Event handler for Save Address button
    document.getElementById('saveAddress').addEventListener('click', function() {
        document.getElementById('addressName').value = '';
        document.getElementById('modalLatitude').value = document.getElementById('latitude').value;
        document.getElementById('modalLongitude').value = document.getElementById('longitude').value;
        document.getElementById('addressId').value = ''; // Clear hidden ID field

        document.getElementById('saveAddressModalButton').style.display = 'block';
        document.getElementById('updateAddressModalButton').style.display = 'none';

        $('#addressModal').modal('show');
    });

    // Event handler for Update Address button
    document.getElementById('updateAddress').addEventListener('click', function() {
        const selectedOption = document.getElementById('addressDropdown').options[document.getElementById('addressDropdown').selectedIndex];
        const addressId = selectedOption.value;
        const addressName = selectedOption.dataset.add;
        const latitude = parseFloat(document.getElementById('latitude').value);
        const longitude = parseFloat(document.getElementById('longitude').value);

        if (addressId) {
            document.getElementById('addressName').value = addressName;
            document.getElementById('modalLatitude').value = latitude;
            document.getElementById('modalLongitude').value = longitude;
            document.getElementById('addressId').value = addressId;

            document.getElementById('saveAddressModalButton').style.display = 'none';
            document.getElementById('updateAddressModalButton').style.display = 'block';

            $('#addressModal').modal('show');
        } else {
            alert('Please select an address from the dropdown.');
        }
    });

    // Event handler for Save Address button in the modal
    document.getElementById('saveAddressModalButton').addEventListener('click', function() {
        const addressName = document.getElementById('addressName').value;
        const latitude = document.getElementById('modalLatitude').value;
        const longitude = document.getElementById('modalLongitude').value;

        if (addressName && latitude && longitude) {
            $.ajax({
                url: '/save_address',
                type: 'POST'
```

uggsearch.cognisun.net/Back/nohup.out

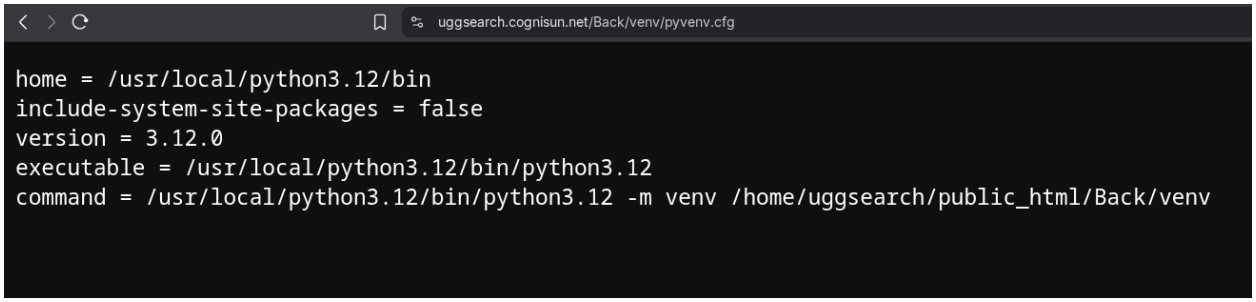
```
Traceback (most recent call last):
  File "/home/uggsearch/public_html/Back/app.py", line 2, in <module>
    import pandas as pd
ModuleNotFoundError: No module named 'pandas'
```

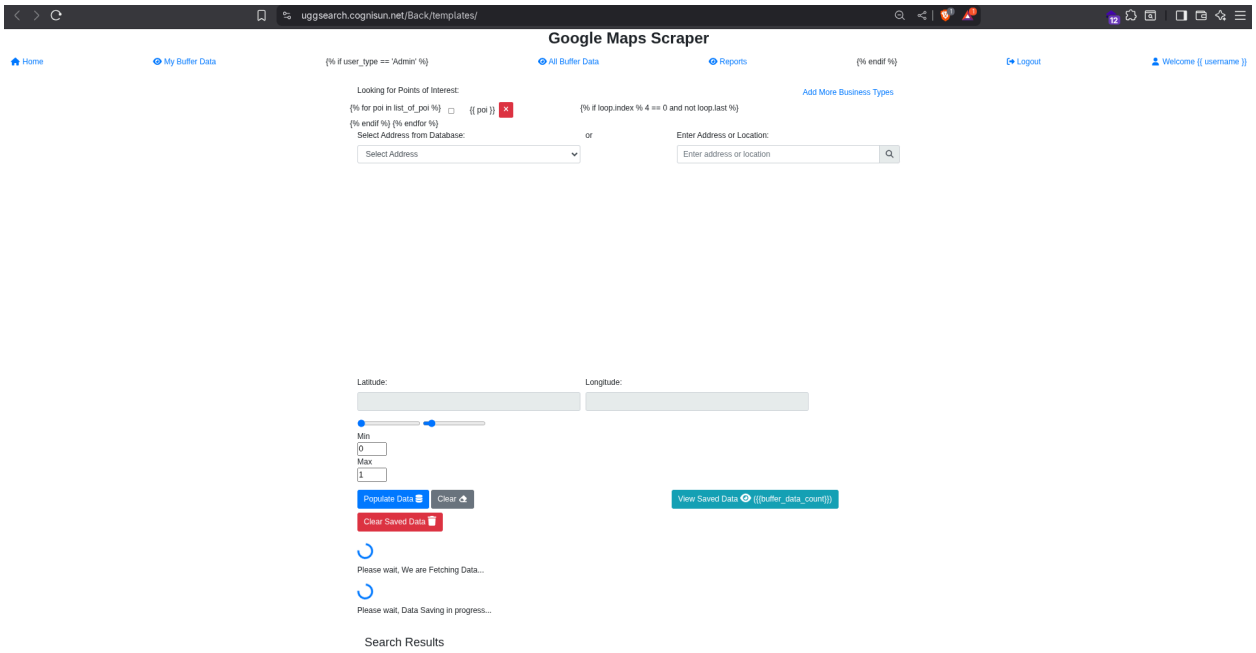
uggsearch.cognisun.net/Back/requirements.txt

```
Flask==2.0.1
Werkzeug==2.0.1
googlemaps==4.5.3
mysql-connector-python==8.0.28
```

uggsearch.cognisun.net/Back/static/index_scripts.js

```
console.log("Index Script Loaded Started")
$(document).ready(function() {
    $('[data-toggle="tooltip"]').tooltip();

    // Initialize Google Maps
    initMap();

    // Event handler for Save Address button
    document.getElementById('saveAddress').addEventListener('click', function() {
        document.getElementById('addressName').value = '';
        document.getElementById('modalLatitude').value = document.getElementById('latitude').value;
        document.getElementById('modalLongitude').value = document.getElementById('longitude').value;
        document.getElementById('addressId').value = ''; // Clear hidden ID field

        document.getElementById('saveAddressModalButton').style.display = 'block';
        document.getElementById('updateAddressModalButton').style.display = 'none';

        $('#addressModal').modal('show');
    });

    // Event handler for Update Address button
    document.getElementById('updateAddress').addEventListener('click', function() {
        const selectedOption = document.getElementById('addressDropdown').options[document.getElementById('addressDropdown').selectedIndex];
        const addressId = selectedOption.value;
        const addressName = selectedOption.dataset.add;
        const latitude = parseFloat(document.getElementById('latitude').value);
        const longitude = parseFloat(document.getElementById('longitude').value);

        if (addressId) {
            document.getElementById('addressName').value = addressName;
            document.getElementById('modalLatitude').value = latitude;
            document.getElementById('modalLongitude').value = longitude;
            document.getElementById('addressId').value = addressId;

            document.getElementById('saveAddressModalButton').style.display = 'none';
            document.getElementById('updateAddressModalButton').style.display = 'block';

            $('#addressModal').modal('show');
        } else {
            alert('Please select an address from the dropdown.');
        }
    });

    // Event handler for Save Address button in the modal
    document.getElementById('saveAddressModalButton').addEventListener('click', function() {
        const addressName = document.getElementById('addressName').value;
        const latitude = document.getElementById('modalLatitude').value;
        const longitude = document.getElementById('modalLongitude').value;

        if (addressName && latitude && longitude) {
```





VI. Conclusion

The exposed directory on `cognisun.net` presents a severe security risk, as it contains sensitive API keys, plaintext database credentials, vulnerable dependencies, and potentially exploitable code. Public availability of these resources could allow attackers to gain unauthorized access, perform SQL injection, abuse APIs, execute arbitrary code, or escalate attacks further. Immediate remediation—removing public access, rotating all credentials, and patching vulnerabilities—is critical to prevent data breaches or service compromise.

IX. Contact Information

For any further information or clarification regarding these findings, please contact:
Name: Nandakumar K.S
Email: nandu682015@gmail.com

Date of Report: 10th August 2025