Introduction

The United Kingdom's DVLA Vehicle Enquiry Service (VES) API[1] plays a crucial role in supporting traffic and parking solutions by offering real-time access to vehicle details based on registration numbers. Traffic management systems can utilize this API to track vehicles entering and exiting specific zones, facilitating efficient traffic flow management. Additionally, parking solutions can leverage the API to streamline parking enforcement processes by verifying vehicle ownership and registration details, thereby reducing fraudulent parking activities and enhancing revenue collection for municipalities. Furthermore, integration of this API into parking apps or systems can provide users with accurate information about available parking spaces and payment options, contributing to a smoother parking experience for residents and visitors alike.

In the realm of smart city security reporting and response solutions, the DVLA Vehicle Enquiry Service API offers valuable insights into vehicle ownership and registration status, aiding law enforcement agencies and security personnel in identifying vehicles involved in criminal activities or security threats. By integrating this API into surveillance systems or security checkpoints, authorities can quickly access vehicle details and cross-reference them with databases of stolen or suspicious vehicles, enabling rapid response to potential security incidents. Moreover, smart city initiatives can benefit from this API by incorporating it into intelligent monitoring systems that automatically flag vehicles of interest, helping to enhance overall safety and security within urban environments.

While no comparable API is publicly available in Kenya today, the purpose of this Hackathon is to explore what systems in the realms of traffic & parking solutions and smart city security reporting and response solutions, could be created should such an API be made available in the future. To support this, Liquid Intelligent Technologies has developed a mock API which resembles the VES API can be used by hackathon participants to showcase possible solutions. Both the mock API as well as this documentation has been inspired by the official VES API and documentation and it is hoped that hackathon participants will be able to leverage VES documentation as well as this document to better understand the system.

Hackathon RESTful API

The following API has been made available for the purposes of this hackathon.

URL	http://vesapi.liquid.tech
PORT	8010
AUTHENTICATION KEY	Hr*ugf(N*&YH

The following additional resources have been made available.

License Plates loaded into system	http://huggingface.co/spaces/dbhenriques/license-plate-reader/raw/main/license-plates.txt
API Postman Collection	http://huggingface.co/spaces/dbhenriques/license-plate-reader/raw/main/VES Hackathon Collection.json
Demo Application	https://huggingface.co/spaces/dbhenriques/license-plate-reader
UK DVLA Developer Portal	https://developer-portal.driver-vehicle-licensing.api.gov.uk/apis/vehicle-enquiry-service/vehicle-enquiry-service-description.html#vehicle-enquiry-service-ves-api-guide
UK's Vehicle Enquiry API v1.2.0	https://developer-portal.driver-vehicle-licensing.api.gov.uk/apis/vehicle-enquiry-service/v1.2.0-vehicle-enquiry-service.html#schemavehicle

API Requests

Authentication

Every request to the API must provide a valid API authentication key. This key must be provided as a key value pair in the header of the request. The key to be used is x-api-key with a value of the API authentication key.

The API authentication key for this hackathon is: Hr*ugf(N*&YH

Body

A valid vehicle registration number must be provided as part of every API query. The vehicle registration number should be included in the body of an HTTP POST request sent to the

system. The content should be in JSON format. An example of such a body is provided below.

```
{
  "registrationNumber": "KCG900V"
}
```

Registration numbers should only include the following characters:

- a-z
- A-Z
- 0-9

The mock API has been prepopulated with over 50 vehicle entries. The vehicle registration numbers for these entries can be found HERE .

Example

A sample request using cURL is provided below:

```
curl -L -X POST '
http://vesapi.liquid.tech:8010/vehicle-enquiry/v1/vehicles'
\
-H 'x-api-key: Hr*ugf(N*&YH' \
-H 'Content-Type: application/json' \
-d '{"registrationNumber": "KCG900V"}
```

Response

An example of a successful API query response is provided below:


```
{
    "registrationNumber": "KCG900V",
    "artEndDate": "2025-03-30",
    "co2Emissions": 300,
    "engineCapacity": 2000,
    "euroStatus": "EURO1",
    "markedForExport": false,
    "fuelType": "PETROL",
    "motStatus": "No details held by DVLA",
    "revenueWeight": 0,
    "colour": "WHITE",
    "make": "TOYOTA",
    "typeApproval": "M1",
```

```
    "yearOfManufacture": 2019,
    "taxDueDate": "2025-03-11",
    "taxStatus": "Taxed",
    "dateOfLastV5CIssued": "2019-05-20",
    "wheelplan": "2 AXLE RIGID BODY",
    "monthOfFirstDvlaRegistration": "2019-03",
    "monthOfFirstRegistration": "2019-03",
    "realDrivingEmissions": "1"
}
```


A detailed description of each field can be found on the UK's Vehicle Enquiry API v1.2.0[3] page.

Error Responses

Here is a list of the possible error responses you may receive using this API:

Status	Meaning	Description
200	OK	A valid request has been received, an existing vehicle entry has been located and data has been returned to requester
401	Unauthorized	Invalid x-api-key received
404	Not Found - Vehicle not found	No entry for the requested vehicle registration number exists in the VES database
404	Not Found - Invalid Registration Number	The requested vehicle registration number is not valid and contains invalid characters

Examples

Code Examples

cURL

```
curl -L -X POST '
http://vesapi.liquid.tech:8010/vehicle-enquiry/v1/vehicles'
\
-H 'x-api-key: Hr*ugf(N*&YH' \
-H 'Content-Type: application/json' \
-d '{"registrationNumber": "KDL194E"}
```

NodeJS (Using Axios)

```
var axios = require('axios');

var data = JSON.stringify({ registrationNumber: 'KDL194E' });

var config = {
  method: 'post',
  url:
    'http://vesapi.liquid.tech:8010/vehicle-enquiry/v1/vehicles',
  headers: {
    'x-api-key': 'Hr*ugf(N*&YH',
    'Content-Type': 'application/json',
  },
  data: data,
};

axios(config)
  .then(function(response) {
    console.log(JSON.stringify(response.data));
  })
  .catch(function(error) {
    console.log(error);
  });
```

Python (Using Requests)

```
import requests

url = "http://vesapi.liquid.tech:8010/vehicle-enquiry/v1/vehicles"

payload = "{\n\t\"registrationNumber\": \"KDL194E\"\n}"
headers = {
  'x-api-key': 'Hr*ugf(N*&YH',
  'Content-Type': 'application/json'
}

response = requests.request("POST", url, headers=headers, data = payload)

print(response.text.encode('utf8'))
```

Postman Collection

The postman collection can be downloaded HERE .

(https://huggingface.co/spaces/dbhenriques/license-plate-reader/blob/main/VES%20Hackathon%20Collection.json)

Please be sure to update the x-api-key in the downloaded postman collection to match the API authentication key provided in the API requests section of this document.

Demo Application - Automatic license-plate recognition (ALPR)

By leveraging pre-trained computer vision models[4], this application allows a user to input an image of a vehicle and the system will isolate the license plate in the image, identify the characters on the license plate, call the VES API and then finally provide a user with information related to that particular vehicle.

An example of this application can be found HERE.

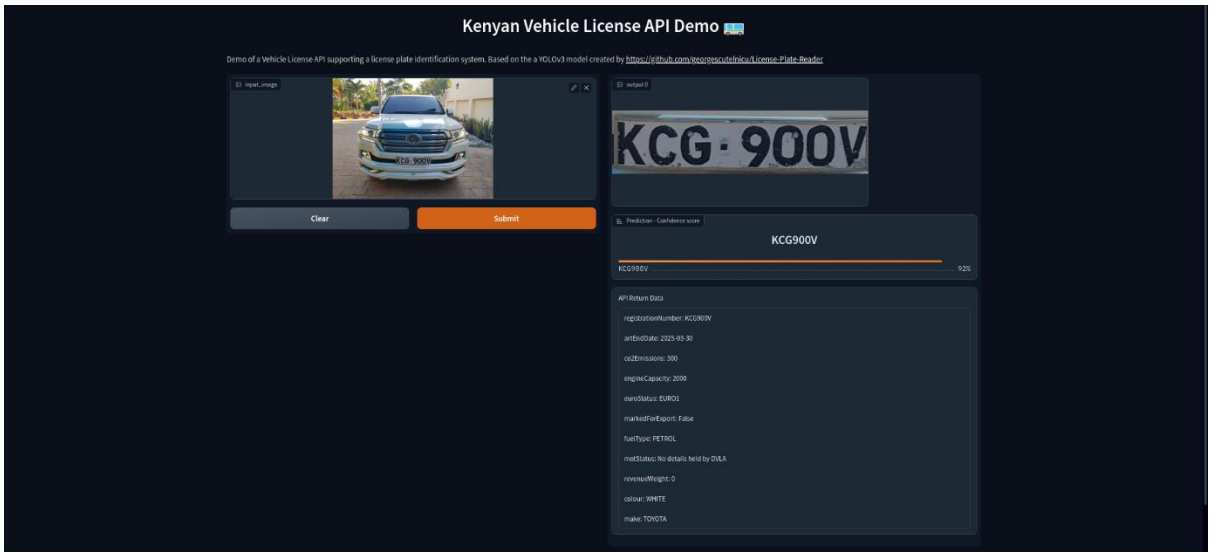

Figure 1 Screenshot of the demo application

Sources

[1] *Dvla Vehicle Enquiry Service (VES) API guide* (no date) *DVLA API Developer Portal*. Available at: https://developer-portal.driver-vehicle-licensing.api.gov.uk/apis/vehicle-

enquiry-service/vehicle-enquiry-service-description.html#vehicle-enquiry-service-ves-api-guide (Accessed: 25 March 2024).

[2] Henriques, D. (no date) ‘License Plates, *Hugging Face*. Available at: https://huggingface.co/spaces/dbhenriques/license-plate-reader/raw/main/license-plates.txt (Accessed: 25 March 2024).

[3] *Vehicle Enquiry API v1.2.0* (no date) *DVLA API Developer Portal*. Available at: https://developer-portal.driver-vehicle-licensing.api.gov.uk/apis/vehicle-enquiry-service/v1.2.0-vehicle-enquiry-service.html#schemavehicle (Accessed: 25 March 2024).

[4] Scutelnicu, G. (no date) ‘Georgescutelnicu License Plate Reader’, *Github*. Available at: https://github.com/georgescutelnicu/License-Plate-Reader (Accessed: 25 March 2024).