UKRAINIAN CATHOLIC UNIVERSITY

BACHELOR THESIS

Development of a car information retrieval system by licence plate

Author:
Borys TURII

Supervisor:
Oles DOBOSEVYCH

A thesis submitted in fulfillment of the requirements for the degree of Bachelor of Science

in the

Department of Computer Sciences
Faculty of Applied Sciences

Lviv 2020

Declaration of Authorship

I, Borys TURII, declare that this thesis titled, "Development of a car information retrieval system by licence plate" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

UKRAINIAN CATHOLIC UNIVERSITY

Faculty of Applied Sciences

Bachelor of Science

Development of a car information retrieval system by licence plate

by Borys TURII

Abstract

During four years of study, we were able to work on almost all available topics in Computer Science. Programming, Algorithms, Robotics, Operating Systems, Artificial Intelligence, Networks, Security, Databases, Cloud Computing, Web Development, and many more. This work is an essence of courses that interested me the most - my trial and final exam. Here Databases, Programming, Cloud Engineering, Web Development will combine into powerful architecture - that will be easy to reproduce and scale. There will be implemented - gathering, analyzing, storing, and providing car data.

Acknowledgements

First of all, I want to thank my parents, who helped me raise my talents and supported me in my objectives.

Many thanks to UCU - a top place for study, work, and communication, where every teacher is willing to teach you what he knows.Thanks, Oles Dobosevych and Yaroslav Prytula for management and Stepan Fedynyak, Oleg Farenyuk for being the best examples of a teacher.

Finally thanks to my best friend Yur Tepliukh, who always supported and motivated me.

Contents

Declaration of Authorship i

Abstract ii

Acknowledgements iii

1 Introduction 1
- 1.1 Motivation . . . 1
- 1.2 Awareness . . . 1
- 1.3 Data . . . 1
- 1.4 Solution . . . 2

2 Background Information 3
- 2.1 Ukrainian vehicle registration plate . . . 3
- 2.2 VIN Code . . . 5

3 Gathering Data: 7
- 3.1 Public databases . . . 7
  - 3.1.1 Registration data . . . 7
  - 3.1.2 Ukrainian Car Insurance - MTIBU . . . 7
  - 3.1.3 "Openness is the key" . . . 7
- 3.2 Online car markets: . . . 9
  - 3.2.1 Auto Ria . . . 9
  - 3.2.2 Rst UA . . . 12

4 Plate recognition: 13
- 4.1 Rekognition . . . 13
- 4.2 Open ALPR . . . 13
- 4.3 ALPR in Unscontrained Scenarios . . . 14
- 4.4 Nomeroff-net . . . 14
- 4.5 Optimization . . . 15
- 4.6 Tests . . . 16

5 Infrastructure development 17

6 Other solutions 22

7 Conclusion 24

Bibliography 25

List of Figures

2.1 Special plates . . . 3
2.2 Old standards . . . 3
2.3 Regular plate numbers . . . 4
2.4 Ukrainian Documents . . . 5
2.5 Checkvinfree (*Using VIN to DECODE where your Vehicle is made*) . . . 5
2.6 autoevolution (*Where can I find the VIN?*) . . . 6
2.7 Door jamb . . . 6

3.1 MTIBU Interface . . . 8
3.2 Opendatabot Interface . . . 8
3.3 Opendatabot API Result . . . 9
3.4 Search API Result . . . 10
3.6 Photos API Result . . . 10
3.5 Info API Result . . . 11
3.7 Relevant data from sources . . . 12

4.1 *OpenALPR CarCheck API - Automatic License Plate Recognition* (2020) . . . 14

5.1 *Build Computer Vision Applications with Amazon Rekognition* 2020 . . . 17
5.2 Platform infrastructure model . . . 18
5.3 Response structure . . . 18
5.4 Examples of response data . . . 19
5.5 Plate number recognition system . . . 20
5.6 Output API creation . . . 21

6.1 *UNDA website* . . . 22
6.2 Exciting examples . . . 23

List of Tables

4.1 Nomeroff net speed test 15
4.2 Speed test 16
4.3 Quality test 16

List of Abbreviations

VIN	**V**ehicle-**I**dentification **N**umber
SaaS	**S**oftware **a**s **a** **S**ervice
API	**A**pplication **P**rogramming **I**nterface
RNN	**R**ecurrent **N**eural **N**etwork
CNN	**C**onvolutional **N**eural **N**etwork
R-CNN	**R**egion-based **CNN**
CV	**C**omputer **V**ision
GPU	**G**raphics **P**rocessing **U**nit
IaC	**I**nfrastructure **a**s **C**ode
GRU	**G**ated **R**ecurrent **U**nit
AWS	**A**mazon **W**eb **S**ervices
ALPR	**A**utomatic **L**icence **P**late **R**ecognition
JSON	**J**ava**S**cript **O**bject **N**otation

Chapter 1

Introduction

1.1 Motivation

Today the Ukrainian government takes measures for more and more data to become public. I was inspired by this trend and searched for its applications in people's everyday life.

Very interesting for me was the used car market. I was holding an idea about a platform where only honest people will trade good cars. Sadly, it is almost impossible in the present world, so the buyer is the only one responsible for buying a reliable vehicle.

1.2 Awareness

There seems to be no other option than just know how people can trick you, and how to protect yourself from it. Apart from car visual and technical condition, the buyer must check documents, registration number, VIN code.

This is public data used to identify and check the vehicle, so the honest seller would not hide it without a reason or focus on it much. The seller must be the owner of the vehicle. Documents, registration plate number must be original. VIN code stamp can be neither damaged nor freshly painted or stickered. Though, the legal procedure of changing license plate data is cheap - appx 20$, VIN is unchangeable, so all of the car data will be linked with it.

1.3 Data

There is a lot of data that will help you clearly understand the history of the vehicle and it is problem parts.

- Number of people, how long they owned (which will characterize the type of usage)
- Car crashes, fines
- Dealership data
- Country of import data
- Previous sales
- ...

Surely, a common person would not have access to all databases and search for the car he is interested in. But, as data is becoming more popular, there are some public data sources and some solutions that analyze this data. I want to develop own one, which will be publicly available. Moreover, I had no practical experience for such big projects, so it will a great challenge for me.

1.4 Solution

I decided to develop a platform that will gather and store as much as possible data for cars from the internet. Their history of sales, photos, specifications is already available to be gathered today. With access to other databases - this platform can be easily extended to all data listed above in 1.3. It can be both systems for people to check cars, and for markets to integrate it.
My work will consist of two vectors:

1. Gather, classify and store cars data
2. Make the solution versatile and scalable using cloud infrastructure

Chapter 2

Background Information

2.1 Ukrainian vehicle registration plate

(A) 2004 Police plate

(B) 2015 Police plate

(C) Military plate

(D) Diplomatic plate

(E) 2004 Public transport plate

(F) 2015 Public transport plate

(G) 2004 Temporary plate

(H) 2015 Temporary plate

FIGURE 2.1: Special plates

A vehicle registration plate in Ukraine is a metal plate attached to a motor vehicle or trailer for official identification purposes.
The size for the actual single line license plate is 520 mm x 112 mm.
Two-line plate for cars with square mount place is 300mm x 150mm.

(A) 1995 Regular plate

(B) 2004 Unstandard plate

(C) 2004 Regular Plate

FIGURE 2.2: Old standards

While old 2.2 are legal to use, during any operation with the vehicle: ownership change, re-registering, technical inspection - it gets new 2015 standard plates with a country code and flag on a blue background 2.3.

(A) 2015 Unstandard plate

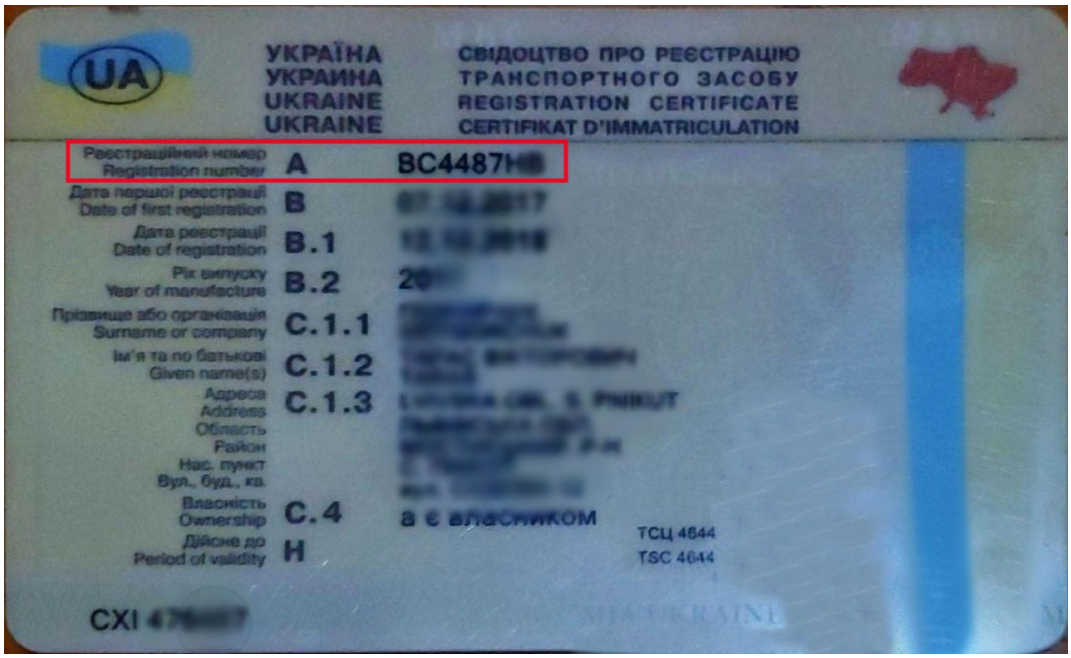
(B) 2015 Regular plate

FIGURE 2.3: Regular plate numbers

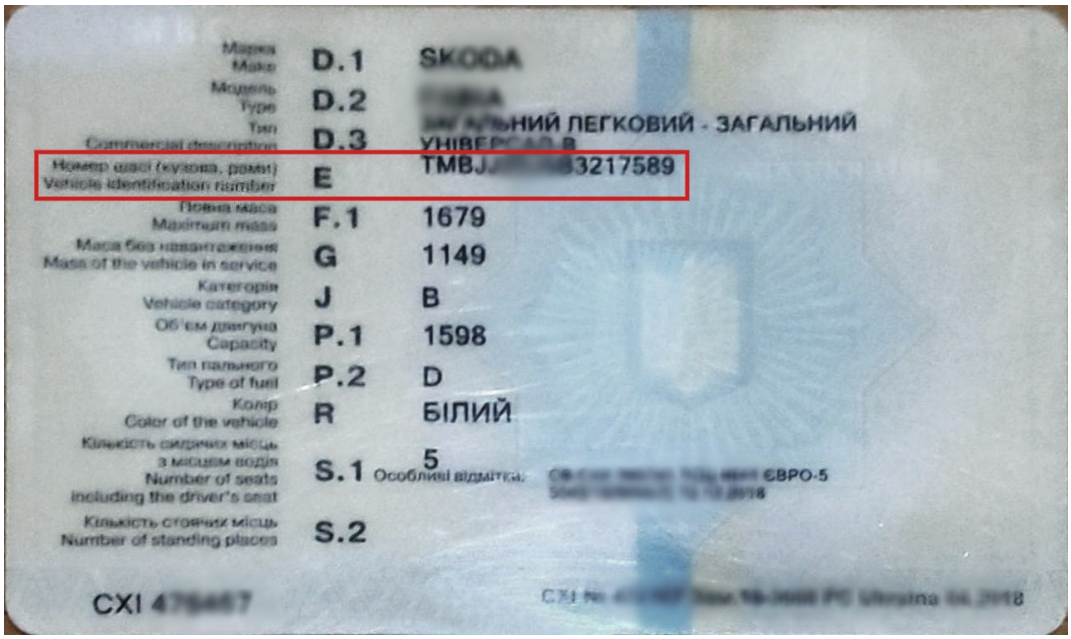
Wikipedia (2020)

> To enable drivers using their vehicle abroad, and to adhere to the Vienna Convention on Road Traffic, Ukrainian regular license plates use only those Cyrillic characters where the glyph resembles a letter from the Roman alphabet; a total of 12 characters: A, B, E, I, K, M, H, O, P, C, T, X).

So, the format of a regular car plate number is AA 0000 BB where AA is a regional prefix, 0 - digit (0-9) then a BB, serial suffix. It must be same on the license plate and in the documents 2.4a
But, as mentioned before, the most accurate identification is the VIN of the car 2.4b.



(A) Number Plate



(B) VIN Code

FIGURE 2.4: Ukrainian Documents

2.2 VIN Code

Autocheck (*What is a vehicle identification number (VIN)?*)

> A **VIN - Vehicle Identification Number** is composed of 17 characters (digits and capital letters) that act as a **unique** identifier for the vehicle. A **VIN** displays the car's unique **features, specifications, and manufacturer**. It can be used to track recalls, registrations, warranty claims, thefts, and insurance coverage.

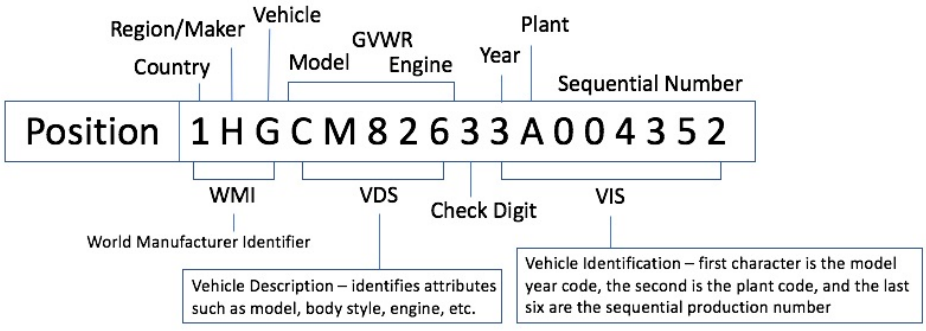


FIGURE 2.5: Checkvinfree (*Using VIN to DECODE where your Vehicle is made*)

These are only general locations of car VIN, nowadays manufacturers have stickers and identifications on many more. That is specific for the manufacturer itself, but all of them have at least one from areas marked red 2.6.

Real photos of most used locations:

- 2.7a On the dashboard, under the windshield, visible from outside
- 2.7b Top left, under the hood, on the suspension top mount.
- 2.7c Bottom of the driver's door jamb

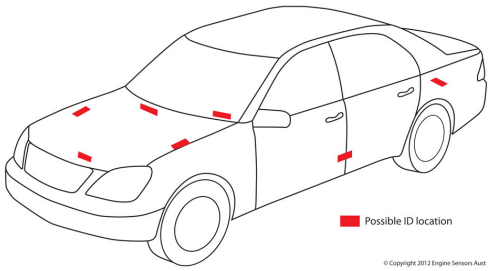


FIGURE 2.6: autoevolution (*Where can I find the VIN?*)

(A) Windshield

(B) Suspension mount

(C) Number Plate

FIGURE 2.7: Door jamb

Chapter 3

Gathering Data:

3.1 Public databases

3.1.1 Registration data

On 28.08.2018 car registration data from 2013 was opened and is currently available. HSC.gov.ua (*A service based on open data on the first registration of cars in Ukraine was opened*)
This dataset provides registration data and technical info about a car by it is plate number: Make, Model, Year of Production, Engine, Weight. Sadly, the VIN is not there, so this data has to be combined with other sources. Data.gov.ua (*Information about vehicles and their owners*) To find info about the desired plate number I have written a simple script @github.
Sample result:

```
"P";"6310400000";"308";"ПЕРЕРЕЄСТРАЦІЯ НА НОВОГО ВЛАСНИКА
ЗА ДОГ. КУПІВЛІ-ПРОДАЖУ (СГ)";03.03.2020;12321;
"ТСЦ 5141";"VOLKSWAGEN";"TIGUAN";2011;"СИНІЙ";"ЛЕГКОВИЙ";
"УНІВЕРСАЛ-В";"ЗАГАЛЬНИЙ";"БЕНЗИН";1984;1629;2180;"АХ5870НХ"
```

3.1.2 Ukrainian Car Insurance - MTIBU

MTIBU (*General Information*):

> Motor Transport Insurance Bureau of Ukraine is the only association of the Insurers which execute the obligatory civil liability insurance of the owners of the ground vehicles against damage made to third persons. Membership in MTIBU is obligatory for the activity of an Insurer in the sphere of obligatory civil liability insurance of the owners of the ground vehicles.

There is a service (*Insurance Policy validation*) to check insurance and technical data, VIN 3.1. But, the *MTIBU API* is available only for companies, so it cannot be integrated into the system implicitly.

3.1.3 "Openness is the key"

Initially, *Opendatabot* is both a company-developer and platform designed to work with government open data. On 28.08.2018, they have released additional functionality to the platform: *Using bot for Car data* In addition to 3.1.1 it provides VIN code. Moreover, sometimes this is the only place I found info about a car during testing. For commerce, they have *Opendatabot API* which they kindly provided for my project. This API 3.3 will be used to identify the car and gather the data.

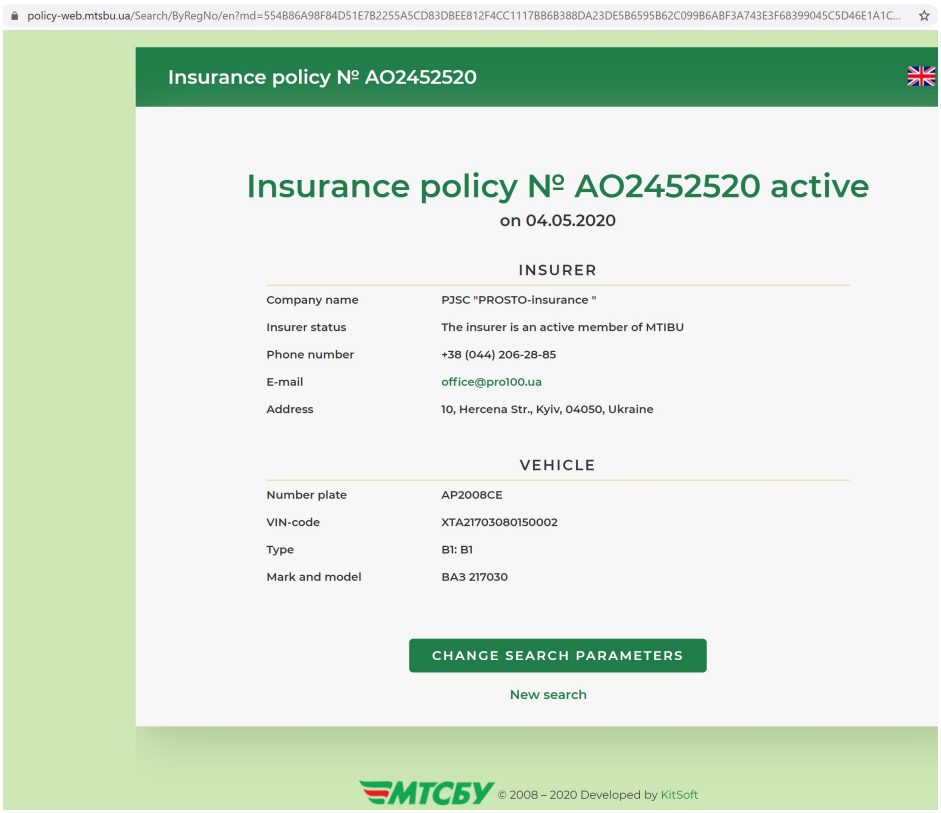


FIGURE 3.1: MTIBU Interface

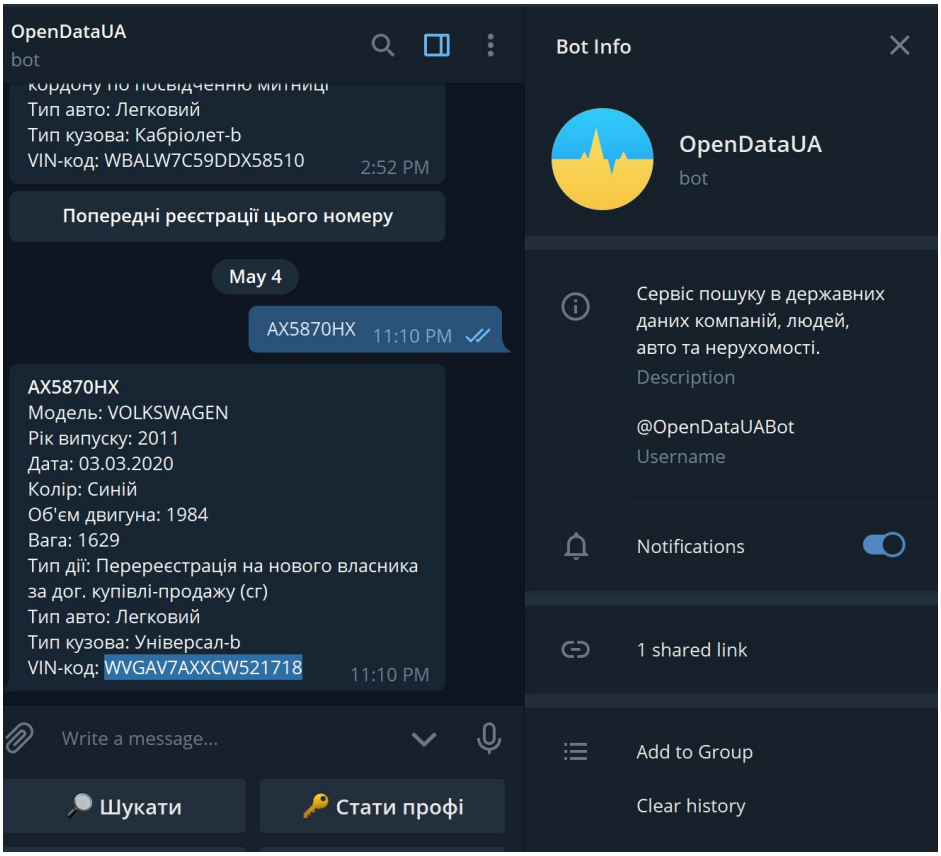


FIGURE 3.2: Opendatabot Interface

```
{
   "status":"ok",
   "data":{
      "count":1,
      "items":[
         {
            "nRegNew":"AX5870HX",
            "brand":"VOLKSWAGEN",
            "capacity":"1984",
            "color":"СИНІЙ",
            "fuel":"БЕНЗИН",
            "kind":"ЛЕГКОВИЙ УНІВЕРСАЛ-В",
            "model":"TIGUAN",
            "nSeating":"5",
            "ownWeight":"1629",
            "makeYear":"2011",
            "totalWeight":"2180",
            "dFirstReg":"2020-03-03",
            "dReg":"2020-03-03",
            "vin":"WVGAV7AXXCW521718"
         }
      ]
    }
}
```


FIGURE 3.3: Opendatabot API Result

3.2 Online car markets:

At the moment of writing the approximate number of active ads is:
Auto.ria.com main page ~181 000
Olx UA main page ~142 000
RST main page ~33 000 (Number of ads for last month)

3.2.1 Auto Ria

Auto.ria.com main page

This platform has its structured system 3.5 for describing the vehicle listing and is opened for other developers through *Ria.com API Portal*. I have got the increase for the API requests limit right in the day I have asked for it. Massive amount of data is accessible by API and each listing has its ID.

IDs

- Starting id is **100001**;
- Last one on the moment of writing is **26905501**;
- Number of actual offers is **180970** 3.4

In sample requests results, some fields were removed to optimize the size. Actual API examples are available in extensive *Auto Ria API Documentation*.
https://developers.ria.com/auto/search?api_key=<API_KEY>&category_id=1&countpage=3&page=1

```
"result":{
      "search_result":{
         "ids":[
            "26897548",
            "20327838",
            "26753282"
         ],
         "count":180970
      }
}
```

FIGURE 3.4: Search API Result

https://developers.ria.com/auto/search?api_key=<API_KEY>&category_id=1&plateNumber.length.gte=1&countpage=3&page=1
Using API we can get list of actual listings that have numberplate recognized or stated by holder with parameter **plateNumber.length.gte=1** ~92000 results.
https://developers.ria.com/auto/info?api_key=<API_KEY>&auto_id=100002
And get an detailed about a car with its listing ID 3.5.
https://developers.ria.com/auto/fotos/198101?api_key=<API_KEY>'
Photos for them will be available with Photos API

```
{
   "status":1,
   "data":{
      "198101":{
         "194991":{
            "photo_id":194991,
            "auto_id":198101,
            "status":0,
            "checked":1,
            "sortingIndex":0,
            "date_add":"0000-00-00 00:00:00",
            "description":null,
            "url":"auto/photo/19/1949/194991/194991.jpg"
         }
      }
   }
}}
```

FIGURE 3.6: Photos API Result

Using these APIs I have created both local and an cloud scripts for data gathering. **After processing, for the plate number we can get link and data.**

```
{
   "userId":26782,
   "locationCityName":"Полтава",
   "cityLocative":"Полтаве",
   "exchangeType":"Любой",
   "addDate":"2006-09-18 16:34:24",
   "soldDate":"2008-02-20 17:16:04",
   "userPhoneData":{
      "phoneId":"",
      "phone":""
   },
   "USD":5800,
   "autoData":{
      "active":false,
      "description":"Сигнализация,  центр. замок",
      "year":2001,
      "autoId":100002,
      "bodyId":3,
      "statusId":1,
      "withVideo":false,
      "race":"69 тыс. км",
      "raceInt":69,
      "fuelId":1,
      "fuelName":"Бензин, 1.5 л.",
      "fuelNameEng":"benzin",
      "gearBoxId":1,
      "gearboxName":"Ручная / Механика",
      "driveName":"Не указано",
      "isSold":true,
      "mainCurrency":"USD",
      "fromArchive":true,
      "categoryId":0,
      "categoryNameEng":"legkovie",
      "subCategoryNameEng":"sedan",
      "custom":0
   },
   "markName":"ВАЗ",
   "markNameEng":"vaz",
   "markId":88,
   "modelName":"21099",
   "modelNameEng":"21099",
   "modelId":855,
   "photoData":{
      "count":1,
      "seoLinkM":"https://cdn2.riastatic.com/photosnew/auto/photo/
      vaz_21099__71712m.jpg"
   },
   "linkToView":"/auto_vaz_21099_100002.html",
   "title":"ВАЗ 21099",
   "VIN":"",
   "haveInfotechReport":false
}
```


FIGURE 3.5: Info API Result

3.2.2 Rst UA

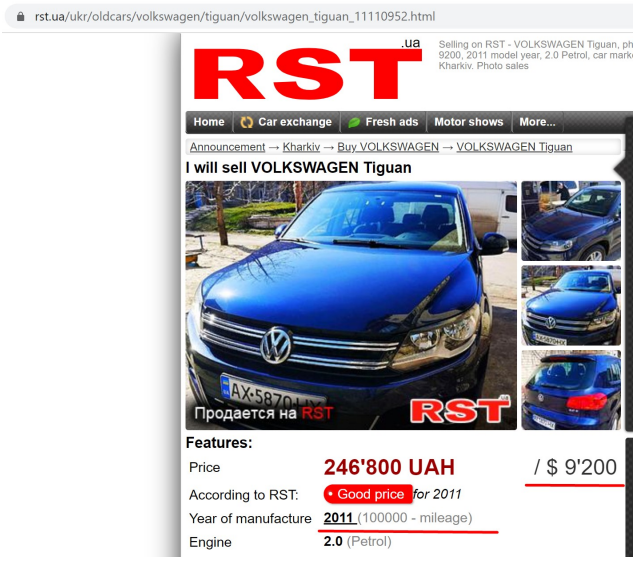
RST main page
This resource is third by the number of cars, has no API, no plate number / VIN code field in structure. Even though it is a marketplace-competitor so the audience intersects a lot, it has different ads that are not published anywhere else. Moreover, it stores all images by <ID>-<Photo Number>
`http://i1.rst.ua/oldcars/volkswagen/tiguan/big/11110952-1.jpg`

IDs

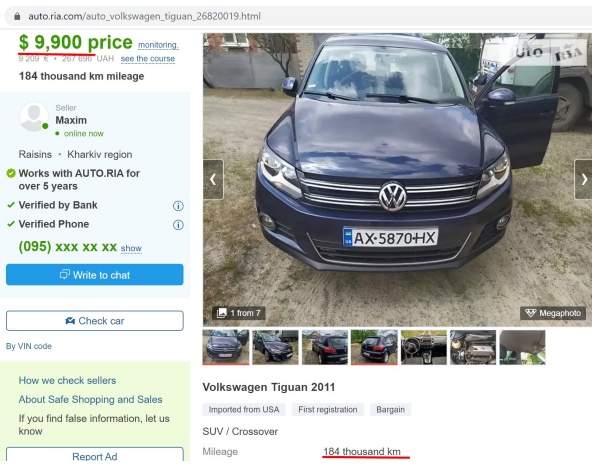
- First accessible listing is **1000000**
- Last accessible listing is **11143392**

I have also written a script for both local and AWS Lambda *AWS Lambda – Serverless Compute - Amazon Web Services* 2020 gathering these images. As there are no registration numbers I will get images from this resource, recognize the license plate numbers and store the resulting data
This will, certainly be the main difference to other solutions(6)

(A) Poctra (*2012 VOLKSWAGEN TIGUAN S, WVGAV7AXXCW521718*)



(B) RST listing of a car



(C) AutoRia listing of the same car

FIGURE 3.7: Relevant data from sources

Chapter 4

Plate recognition:

As on both resources plate numbers can be missing - to identify listings and cars we will need to have a plate number recognizing system. For cloud infrastructure easiest CV solution is:

4.1 Rekognition

In *What Is Amazon Rekognition? - Amazon Rekognition* 2020 it is stated that:
"Finally, in public safety applications, you can identify vehicles based on license plate numbers from images taken by street cameras."

But, it is not customizeable because of its SaaS model - it can only detect what it's creators embedded: face, objects, **text**. During tests, on the images, there was a lot of different text next to the registration number (e.g. on plastic frame), and free limit is up to 5000 images per month, so this is not a solution I have chosen. I have looked for an **open source one**.

4.2 Open ALPR

> OpenALPR is an open source Automatic License Plate Recognition library written in C++ with bindings in C, Java, Node.js, Go, and Python. The library analyzes images and video streams to identify license plates. The output is the text representation of any license plate characters.

openalpr (2020)
And Commercial solution with both IP camera and Cloud-hosted ALPR API *OpenALPR API — openalpr 2.7.102 documentation* (2020)

> The OpenALPR CarCheck API is a web-based service that analyzes images for license plates as well as vehicle information such as make, model, and color. The CarCheck API service is easy to integrate into your application via a web-based REST service. When you send image data to the OpenALPR API, we process that data and return JSON data describing the license plate and vehicle.

Commercial solution has very wide functionallity. I was surprised with the photo 4.1 - with bad lightning, dark car, without car make logo - I would not guess the vehicle so great. In opposite to open source one, which has same incredible speed but not good quality results while test 4.6

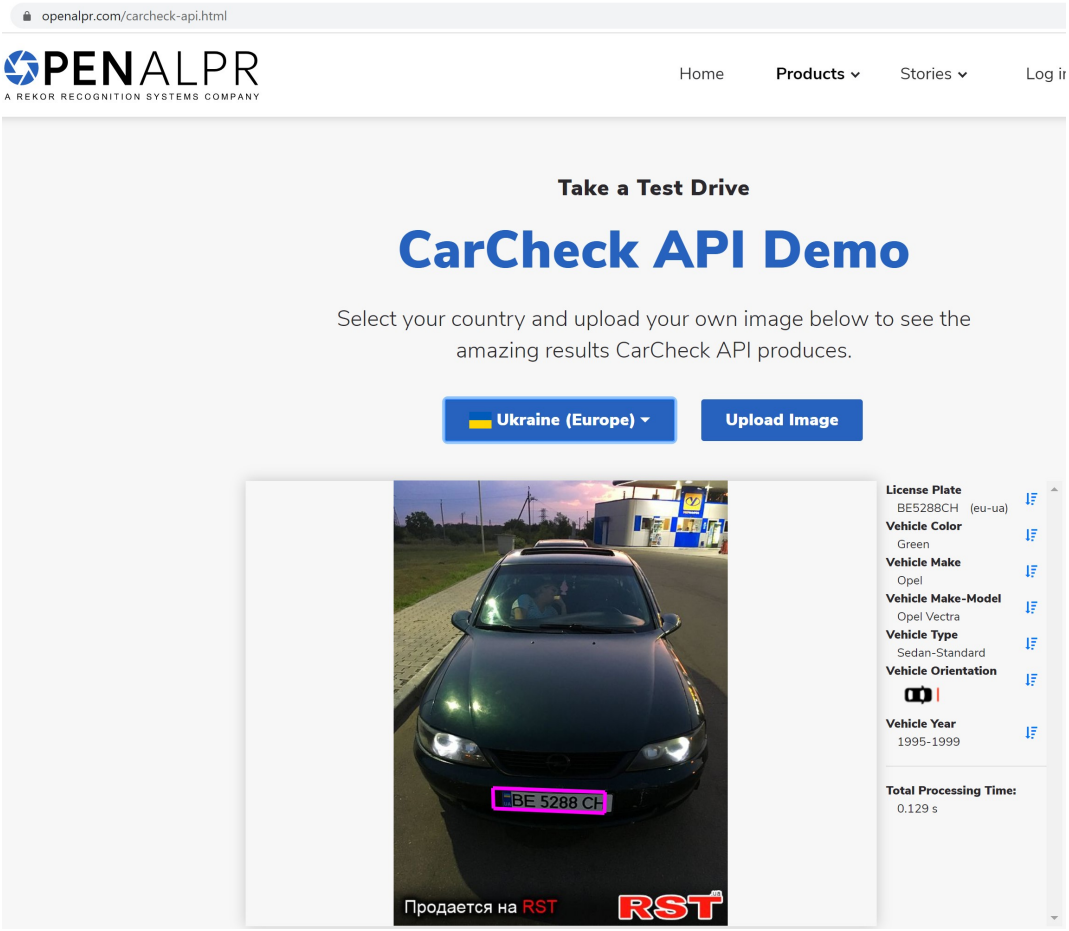


FIGURE 4.1: *OpenALPR CarCheck API - Automatic License Plate Recognition* (2020)

4.3 ALPR in Unscontrained Scenarios

> This work proposes a complete ALPR system focusing on unconstrained capture scenarios, where the LP might be considerably distorted due to oblique views. Our main contribution is the introduction of a novel Convolutional Neural Network (CNN) capable of detecting and rectifying multiple distorted license plates in a single image, which are fed to an Optical Character Recognition (OCR) method to obtain the final result.

(Silva and Jung, 2019)
Paper is available at: (Silva and Jung, 2018) Implementation: (sergiomsilva, 2019)

4.4 Nomeroff-net

"Nomeroff Net is a opensource python license plate recognition framework based on the application of a convolutional neural network on the Mask RCNN architecture, and cusomized OCR-module powered by GRU architecture." Cherniy and Probachay, 2020b

> We present a conceptually simple, flexible, and general framework for object instance segmentation. Our approach efficiently detects objects in

> an image while simultaneously generating a high-quality segmentation mask for each instance. The method, called Mask R-CNN, extends Faster R-CNN by adding a branch for predicting an object mask in parallel with the existing branch for bounding box recognition. Mask R-CNN is simple to train and adds only a small overhead to Faster R-CNN, running at 5 fps. Moreover, Mask R-CNN is easy to generalize to other tasks, e.g., allowing us to estimate human poses in the same framework.

He et al., 2017 **Why Nomeroff Net**

- Actual
- Opensource - customizable
- Easy installation
- Based on Tensorflow - easy switch to GPU Computing 4.5
- Easy to move into cloud
- Trained and optimized for Ukrainian 2.2 2.3 plates - accuracy tests below

4.5 Optimization

Cherniy (2019) optimization tests - 10 runs x 260 images Numbers in this table are rounded, original data can be found at Cherniy and Probachay (2020a)

TABLE 4.1: Nomeroff net speed test

Equipment	**[1]**	**[1] + [2]**	**[1] + [2] + XLA**
Time (s)	415.73	145.77	140.0
Speed (s/pic)	1.56	0.56	0.53868
Equipment number	**Equipment name**		
[1]	Intel® Core™ i9-9900K 16M Cache, 3.6 GHz - 5.00 GHz		
[2]	Nvidia GeForce® GTX 1080 Ti 11 Gb DDR5		

XLA: Optimizing Compiler for Machine Learning | TensorFlow 2020

4.6 Tests

To check which library will be the best in my solution I have used 372 pictures from `https://nomeroff.net.ua/datasets/autoriaNumberplateDataset-2018-11-20.zip` which did not have labels, so I labeled it manually. It was not okay for speed tests, so I parsed pictures from RST and I got 2506 pictures 4.2 Test 0. After filtering them I received 2025 of 2014 or 2015 Format with hard angles, unclear picture, etc.

TABLE 4.2: Speed test

Library:	**OpenALPR**	**Unconstrained**	**Nomeroff**
Test 0 [1]	2 min	17 min	43 min
Test 0 [2]	2 min	15 min	35 min
Test 1 [2]	2 min	14 min	30 min
Test 1 [1*]			
Test 0	2506 Images	**Test 1**	2025 Images
[1] - vGPU1	AWS p2.xlarge (NVIDIA® Tesla® K80)		
[1*] - vGPU2	AWS p3.2xlarge (NVIDIA® V100)		
[2] - Local	Intel® Core™ i7-8650U + NVIDIA® GeForce® GTX 1060 Mobile		

TABLE 4.3: Quality test

Library:	**OpenALPR**	**Unconstrained**	**Nomeroff net**
Test 0 [R]	92	217	329
Test 0 [RW]	4	1	8
Acc (R-RW)/N	**0.23656**	**0.58**	**0.8629**
Test 1 [R]	481	967	1950
Test 1 [RW]	13	6	36
Accuracy	**0.231**	**0.47457**	**0.9452**
Test 0	372 Images	Good quality	Good visibility
Test 1	2025 Images	Different quality	Hard angles
[R]	Recognized regular 2004 or 2015 plate		
[RW]	Number recognized incorrectly		

Chapter 5

Infrastructure development

As stated before I have created both local and cloud scripts. Now, I will mostly focus on cloud solutions, local remain only for optional testing. Moreover, I prefer IaC approach, so even infrastructure tends to be coded - which can be done in AWS via CloudFormation *AWS CloudFormation - Infrastructure as Code & AWS Resource Provisioning* 2020 This is how AWS recommend to process Image recognition.

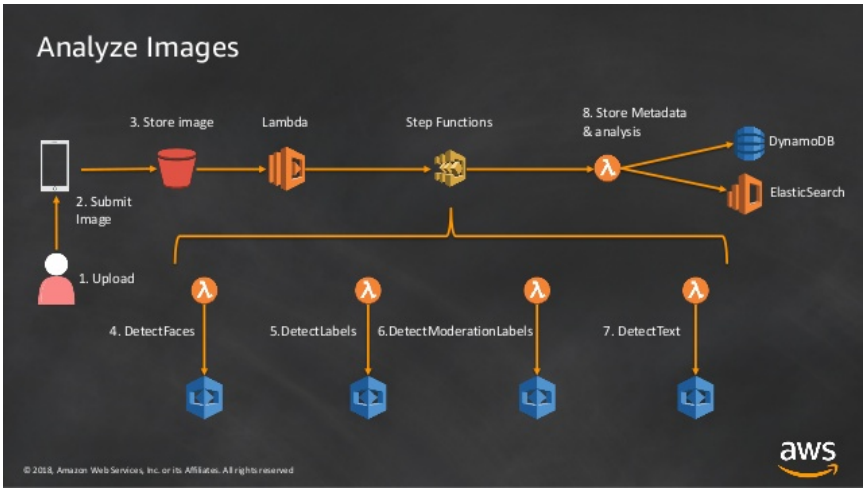


FIGURE 5.1: *Build Computer Vision Applications with Amazon Rekognition* 2020

Instead of Rekognition and Lambda I used EC2 with Nomeroff net library.

Amazon EC2 (2020) " Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud.. "

EC2 was chosen because it has massive computing power with NVIDIA GPU Instances. GPU version Tensorflow can be used and optimized 4.5.

Plate recognition is only a part of this system 5.3 I have developed. Source data is being parsed via Lambda *AWS Lambda – Serverless Compute - Amazon Web Services* 2020;
Stored in S3 *Cloud Object Storage | Store & Retrieve Data Anywhere | Amazon Simple Storage Service (S3)* 2020;
then analyzed and identified via EC2 *Amazon EC2* 2020;
and finally stored in DynamoDB *Amazon DynamoDB - Overview* 2020.
The request from client side comes to API Gateway, is handled by another Lambda that pulls data from DynamoDB and returns JSON. Whole solution can be deployed with CloudFormation within 15 minutes and ready to work. Deletion takes around 2 minutes.

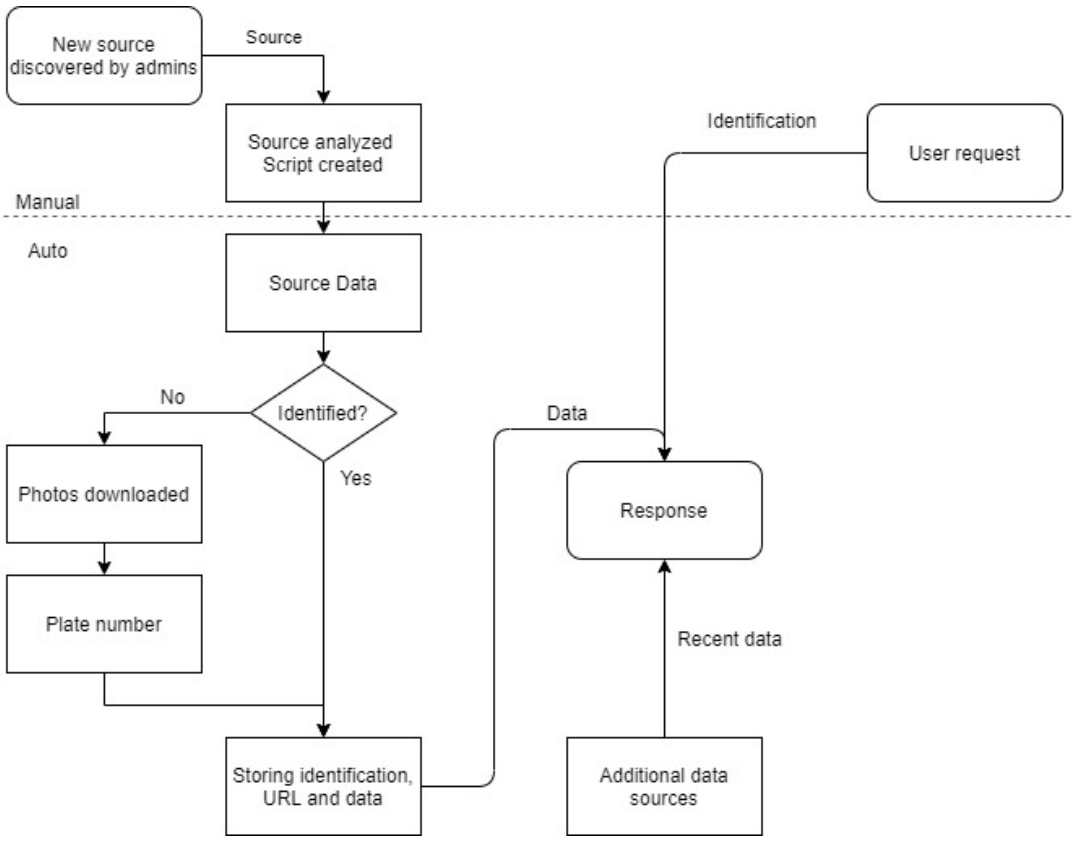


FIGURE 5.2: Platform infrastructure model

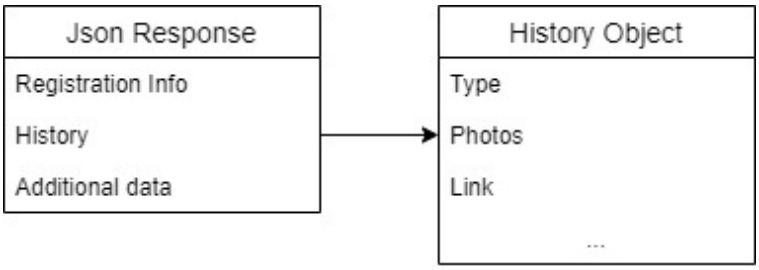


FIGURE 5.3: Response structure

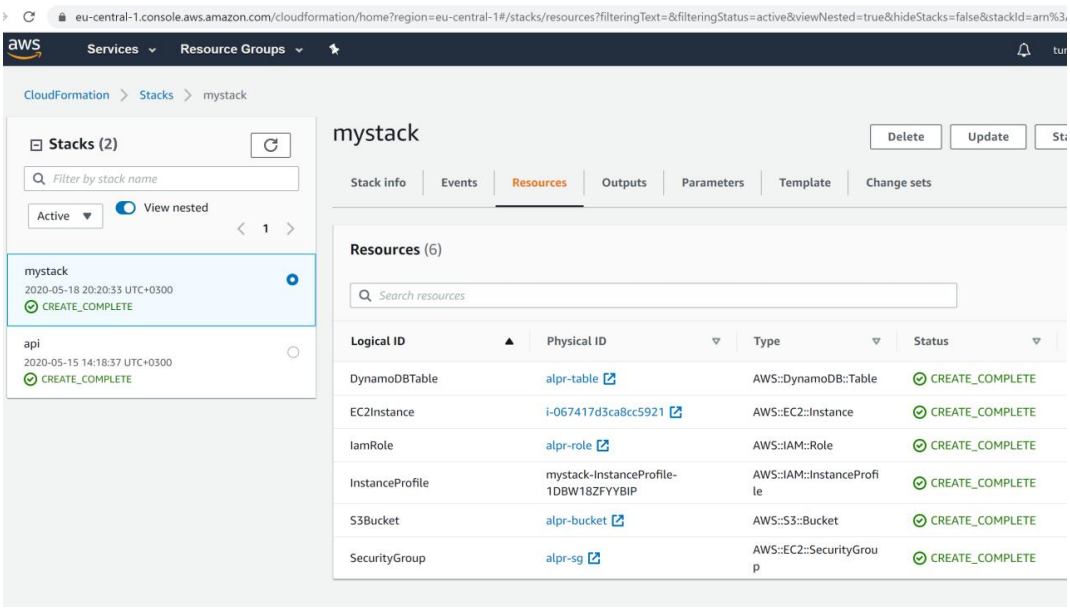
```
{
"Registration info":{
   "VIN":"1ZZ",
   "Plate Number":"BH0880IB"
},
"History":[
   {
      "Type":"General",
      "Photos":[
         "http://img03.platesmania.com/200507/m/14669420.jpg",
         "http://img03.platesmania.com/190703/m/13097492.jpg"
      ],
      "Link":"http://platesmania.com/ua/nomer14669420"
   }]
}
```


(A)

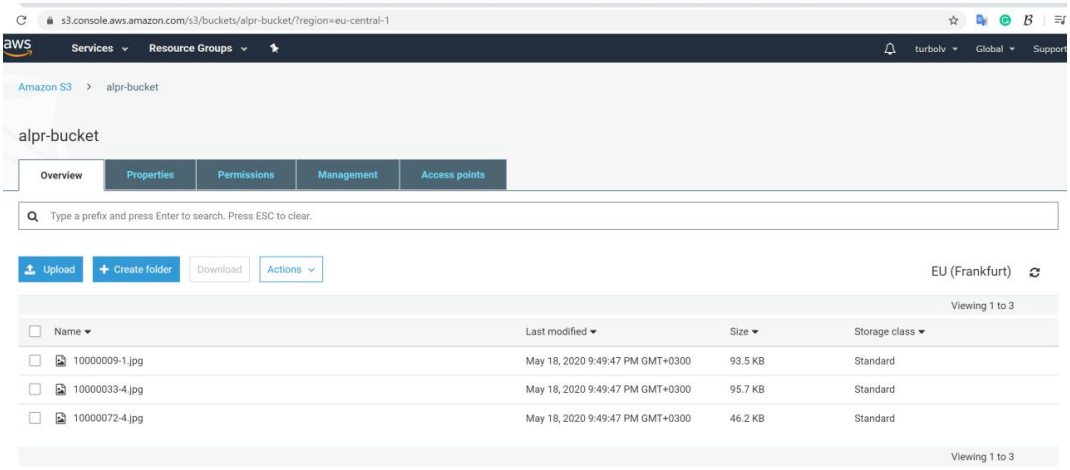

```
{
   "Registration info":{
      "VIN":"1ZZ",
      "Plate Number":"BH0880IB"
   },
   "History":[
      {
        "Type":"Autoria",
        "Photos":["https://cdn2.riastatic.com/
        photosnew/auto/photo/vaz_21099__71712m.jpg"],
        "Link":"https://auto.ria.com/auto_vaz_21099_100002",
        "SourceData":{
           "userId":26782,
           "locationCityName":"Полтава",
           "cityLocative":"Полтаве",
           "exchangeType":"Любой",
           "addDate":"2006-09-18 16:34:24",
           "soldDate":"2008-02-20 17:16:04",
           "autoData":{
              "active":false,
              "description":"Сигнализация,  центр. замок",
              "year":2001,
              "autoId":100002,
              "raceInt":69
            }
        }
      }]
}
```


(B)

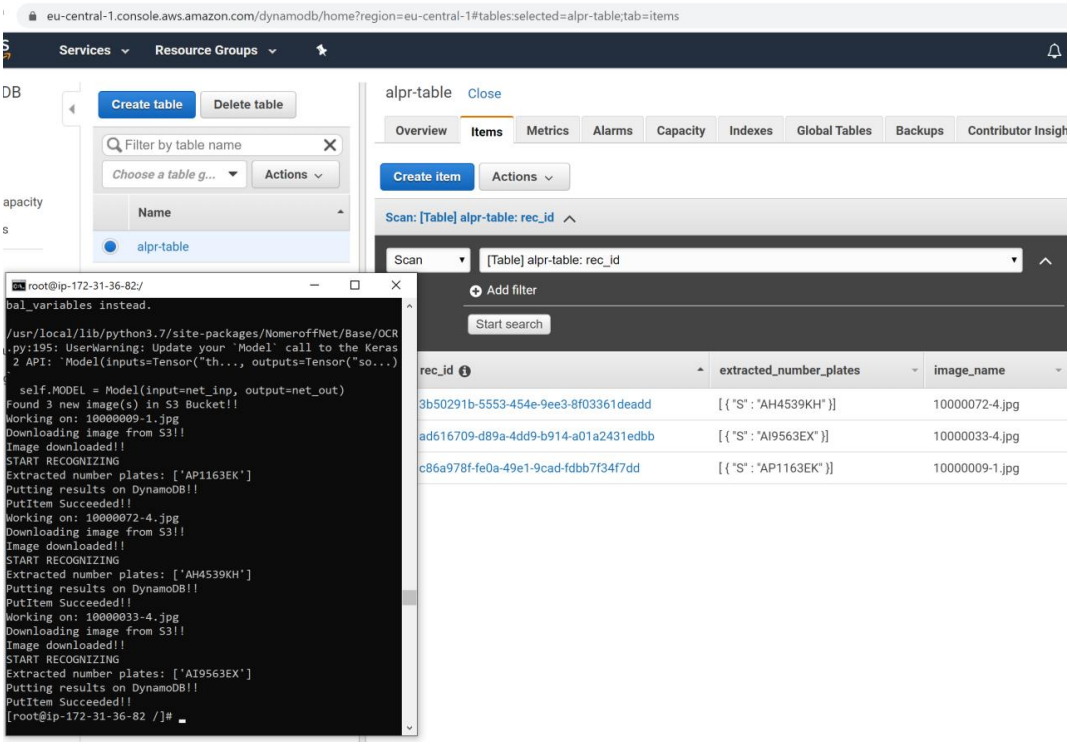
FIGURE 5.4: Examples of response data



(A) S3 for images, EC2 for Recognition, and DynamoDB for storing

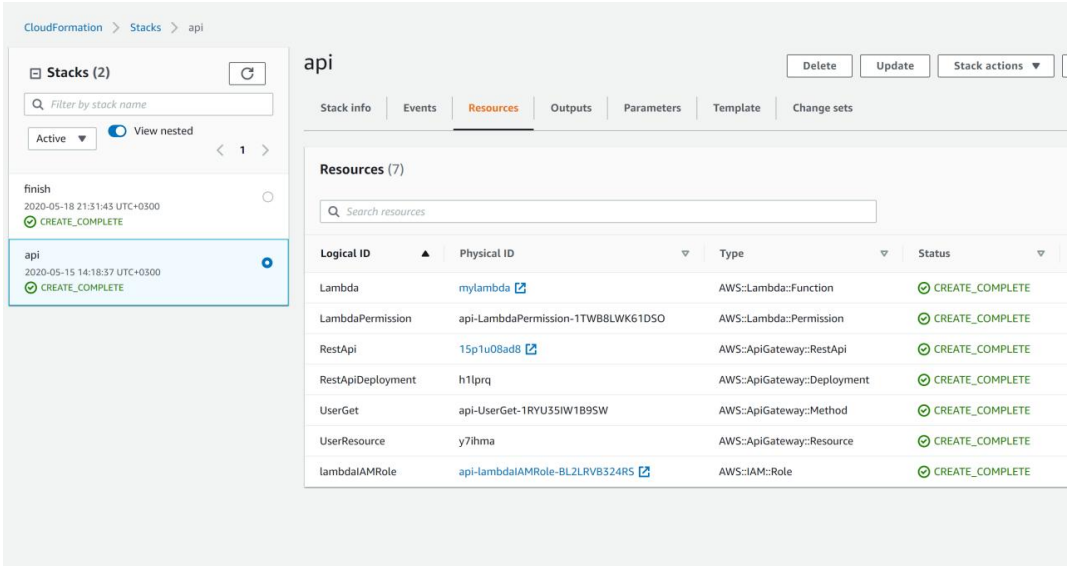


(B) Images Uploaded to S3

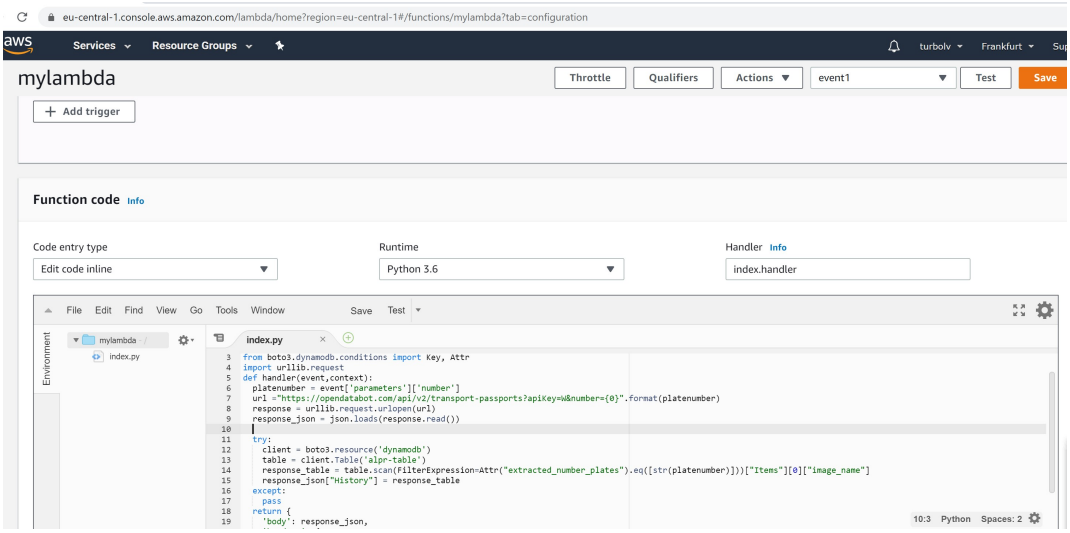


(C) EC2 recognized -> DynamoDB stored

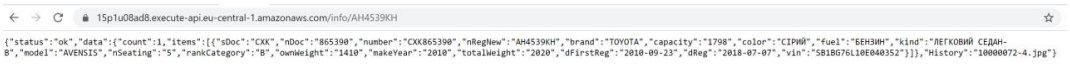
FIGURE 5.5: Plate number recognition system



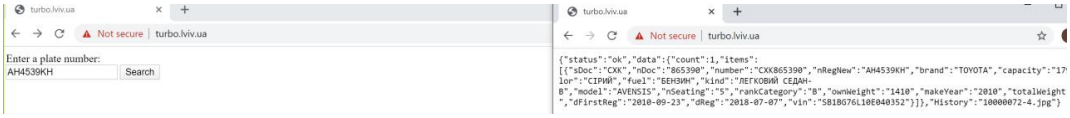
(A) Lambda for info retrieve, RestApi for returning it



(B) Lambda gets info from table and from API



(C) Api response



(D) Website for API

FIGURE 5.6: Output API creation

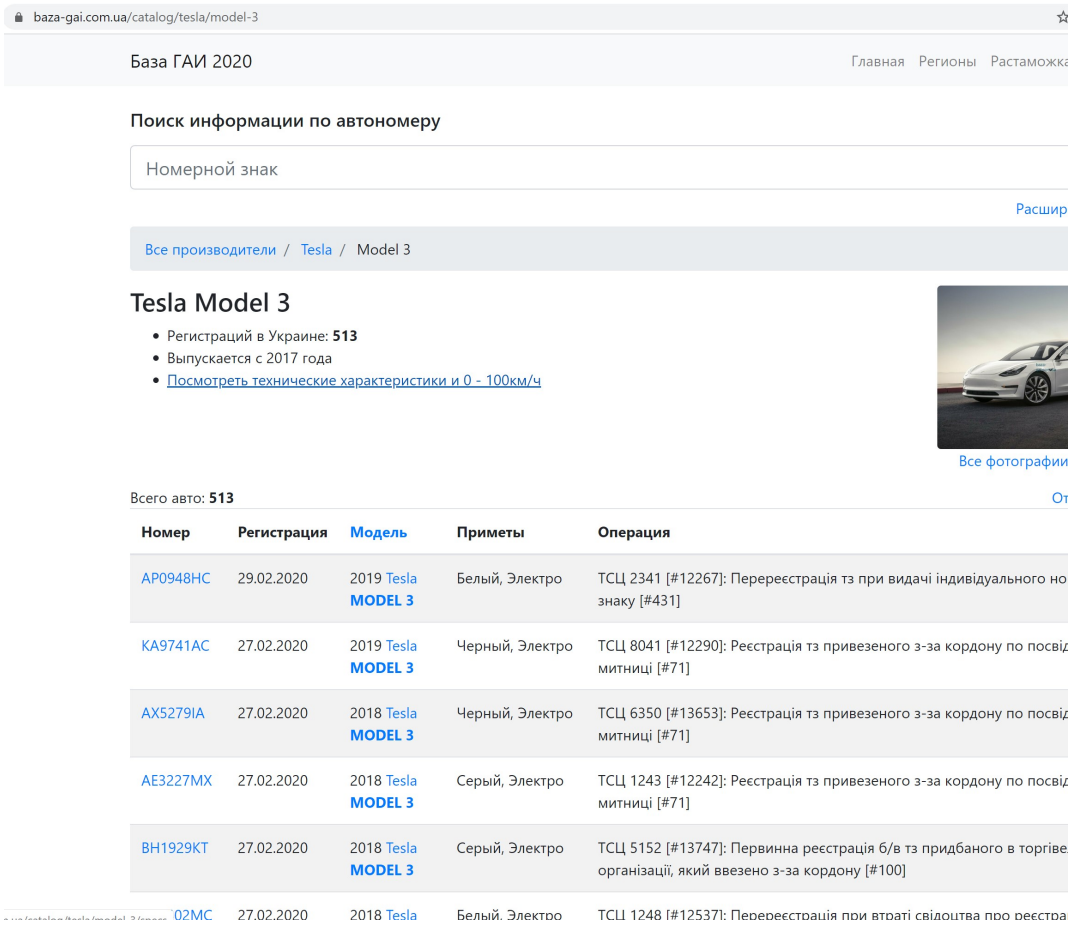
Chapter 6

Other solutions

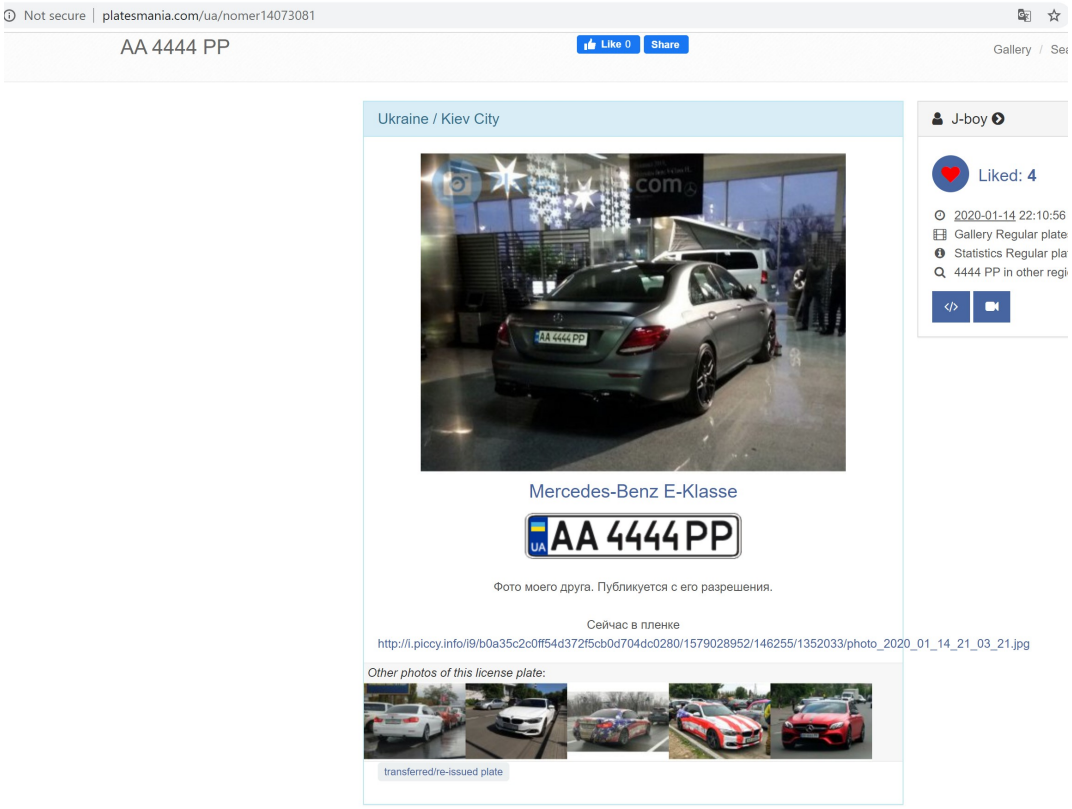
There are some solutions that just use the open dataset 3.1.1 .

FIGURE 6.1: *UNDA website*

But, even with this data there are some interesting realisations of it. Second solution also uses it, but they have added very interesting function - to check all registration by vehicle Make and Model 6.2a
Last one 6.2b is completely different, it is a service where people send their photos of different vehicles and the photo can be found using plate number.



(A) *baza-gai Tesla Model 3* (2020)



(B) *AA 4444 PP, Mercedes-Benz E-Klasse (Kiev City) License plate Ukraine* (2020)

FIGURE 6.2: Exciting examples

Chapter 7

Conclusion

I have developed a unique system that can help people get information about a car using plate number or VIN code. This system is cloud-based so it can be deployed when needed and updated easily. So, two vectors i set up were accomplished.
I practiced in

- LaTex writing
- citing
- labeling images
- gathering and analyzing data
- installing and testing CV models
- designing cloud solutions
- automated cloud infrastructure setup
- designing a database

I have contacted people from mentioned here projects for advices.

The CloudFormation script `https://carplates.s3.eu-central-1.amazonaws.com/SetupRecognizer.yaml` The API is available with the link: `turbo.lviv.ua` I will continue my work on the project and form the database and develop automatical updates to it, discover and adapt new sources of data.

Bibliography

AA 4444 PP, Mercedes-Benz E-Klasse (Kiev City) License plate Ukraine (2020). [Online; accessed 11. May 2020]. URL: http://platesmania.com/ua/nomer14073081.

Amazon DynamoDB - Overview (2020). [Online; accessed 14. May 2020]. URL: https://aws.amazon.com/dynamodb.

Amazon EC2 (2020). [Online; accessed 11. May 2020]. URL: https://aws.amazon.com/ec2.

Autocheck. *What is a vehicle identification number (VIN)?* URL: https://www.autocheck.com/vehiclehistory/vin-basics.

autoevolution. *Where can I find the VIN?* https://www.autoevolution.com/news/how-to-read-your-toyota-vin-code-80056.html. [Online; accessed 21-January-2020].

AWS CloudFormation - Infrastructure as Code & AWS Resource Provisioning (2020). [Online; accessed 11. May 2020]. URL: https://aws.amazon.com/cloudformation.

AWS Lambda – Serverless Compute - Amazon Web Services (2020). [Online; accessed 11. May 2020]. URL: https://aws.amazon.com/lambda.

baza-gai Tesla Model 3 (2020). [Online; accessed 11. May 2020]. URL: https://baza-gai.com.ua/catalog/tesla/model-3.

Build Computer Vision Applications with Amazon Rekognition (2020). [Online; accessed 18-April-2020]. URL: https://www.slideshare.net/AmazonWebServices/build-computer-vision-applications-with-amazon-rekognition-123237318.

Checkvinfree. *Using VIN to DECODE where your Vehicle is made*. https://www.checkvinfree.com/using-vin-to-decode-where-your-vehicle-is-made/. [Online; accessed 21-January-2020].

Cherniy, Oleg (2019). *How to optimize models and hardware for fast inference on a highload project*. [Online; accessed 10. May 2020]. URL: https://aiukraine.com/wp-content/uploads/2019/07/YAk-optimizuvati-modeli-ta-_zalizo_-pid-shvidkij-inferens-na-highload-proekti.-RIA.com_.-AI-Ukraine-2019-Kiyiv-2019.pdf.

Cherniy, Oleg and Dmytro Probachay (2020a). *nomeroff-net*. [Online; accessed 10. May 2020]. URL: https://github.com/ria-com/nomeroff-net/blob/master/banchmarks/runtime.ipynb.

— (2020b). *"Nomeroff Net. Automatic numberplate recognition system from AUTO.RIA.com. Version 0.2.3"*. https://nomeroff.net.ua. [Online; accessed 18-April-2020].

Cloud Object Storage | Store & Retrieve Data Anywhere | Amazon Simple Storage Service (S3) (2020). [Online; accessed 11. May 2020]. URL: https://aws.amazon.com/s3.

Data.gov.ua. *Information about vehicles and their owners*. URL: https://data.gov.ua/dataset/06779371-308f-42d7-895e-5a39833375f0.

He, Kaiming et al. (2017). "Mask R-CNN". In: *arXiv*. eprint: 1703.06870. URL: https://arxiv.org/abs/1703.06870.

HSC.gov.ua. *A service based on open data on the first registration of cars in Ukraine was opened*. Service Center of the Ministry of Internal Affairs. URL: http://hsc.gov.ua/2018/08/28/v-ukrayini-zapustili-servis-na-osnovi-vidkritih-danih-pro-pershu-reyestratsiyu-avto/.

MTIBU. *General Information*. URL: http://www.mtsbu.ua/en/about_us/general/.
— *Insurance Policy validation*. URL: https://policy-web.mtsbu.ua/Search/Main/en.
— *MTIBU API*. https://policy.mtsbu.ua/DpApiDoc/doc/index.html. URL: http://www.mtsbu.ua/files/For_SK/EP/PEP_questions_vg\%20.pdf.
Olx. *Olx UA main page*. URL: https://www.olx.ua/.
openalpr (2020). *openalpr*. [Online; accessed 11. May 2020]. URL: https://github.com/openalpr/openalpr.
OpenALPR API — openalpr 2.7.102 documentation (2020). [Online; accessed 14. May 2020]. URL: http://doc.openalpr.com/api.html.
OpenALPR CarCheck API - Automatic License Plate Recognition (2020). [Online; accessed 14. May 2020]. URL: https://www.openalpr.com/carcheck-api.html.
Opendatabot. *Opendatabot*. URL: https://opendatabot.ua/press.
— *Opendatabot API*. URL: https://docs.opendatabot.com/#/transport/transportPassportsCollection
— *Using bot for Car data*. URL: https://opendatabot.ua/blog/ru/234-open-cars#collapseBots.
Poctra. *2012 VOLKSWAGEN TIGUAN S, WVGAV7AXXCW521718*. https://poctra.com/2012-VOLKSWAGEN-TIGUAN-S/id-As2aYw5rpWb1CMyO/MIAMI-FL.
Ria.com. *Auto Ria API Documentation*. URL: https://github.com/ria-com/auto-ria-rest-api/tree/master/AUTO_RIA_API.
— *Auto.ria.com main page*. URL: https://auto.ria.com/.
— *Ria.com API Portal*. URL: https://developers.ria.com/.
RST. *RST main page*. URL: https://rst.ua/.
sergiomsilva (2019). *alpr-unconstrained*. [Online; accessed 14. May 2020]. URL: https://github.com/sergiomsilva/alpr-unconstrained.
Silva, S. M. and C. R. Jung (2018). "License Plate Detection and Recognition in Unconstrained Scenarios". In: *2018 European Conference on Computer Vision (ECCV)*, pp. 580–596. DOI: 10.1007/978-3-030-01258-8_36.
Silva, Sergio Montazzolli and Claudio Rosito Jung (2019). *License Plate Detection and Recognition in Unconstrained Scenarios*. [Online; accessed 14. May 2020]. URL: http://sergiomsilva.com/pubs/alpr-unconstrained.
UNDA website.
What Is Amazon Rekognition? - Amazon Rekognition (2020). [Online; accessed 18-April-2020]. URL: https://docs.aws.amazon.com/rekognition/latest/dg/what-is.html.
Wikipedia (2020). *Vehicle registration plates of Ukraine — Wikipedia, The Free Encyclopedia*. http://en.wikipedia.org/w/index.php?title=Vehicle%20registration%20plates%20of%20Ukraine&oldid=946536827. [Online; accessed 21-January-2020].
XLA: Optimizing Compiler for Machine Learning | TensorFlow (2020). [Online; accessed 10. May 2020]. URL: https://www.tensorflow.org/xla.