UNIVERSITY of RWANDA

ACEIoT

Website: www.aceiot.ur.ac.rw
Mail: aceiot@ur.ac.rw

College of Science and Technology

AFRICAN CENTER OF EXCELLENCE IN INTERNET OF THINGS

Research Thesis Title: Integration of Vision IoT, AI-based OCR and Blockchain Ledger for Immutable Tracking of Vehicle's Departure and Arrival Times

A dissertation submitted in partial fulfilment of the requirements for the award of masters of science degree in internet of things: Embedded computing system

Submitted By:

Name: Marumbo Sichinga (Ref. No: 221000527)

December, 2022

UNIVERSITY of RWANDA

ACEIoT

Website: www.aceiot.ur.ac.rw
Mail: aceiot@ur.ac.rw

College of Science and Technology

AFRICAN CENTRE OF EXCELLENCE IN INTERNET OF THINGS

Research Thesis Title:

Integration of Vision IoT, AI-based OCR and Blockchain Ledger for Immutable Tracking of Vehicle's Departure and Arrival Times

A dissertation submitted in partial fulfilment of the requirements for the award of masters of science degree in internet of things: Embedded computing system

Submitted By

Marumbo SICHINGA (REF.NO: 221000527)

Supervised by:
-Dr Jimmy Nsenga
-Dr Ignace Gatare

December, 2022

Declaration

I SICHINGA Marumbo, Master student from African Center of Excellence in Internet of Things (ACEIoT), at University of Rwanda. I declare that this research thesis is my own original work, and it has never been presented before anywhere in the world.

Marumbo SICHINGA

Ref: 221000527

Signed:

Date: 1/03/2023

Bonafide certificate

This is to certify that this submitted Research Thesis work report is a record of the original work done by **SICHINGA Marumbo** (**Ref. Nu: 221000527**), MSc. IoT-ECS Student at the University of Rwanda / College of Science and Technology / African Center of Excellence in Internet of Things (ACEIoT), the Academic year 2021/2022.

This work has been submitted under the supervision of Dr Jimmy Nsenga and Dr Ignace Gatare

Main Supervisor: Dr Jimmy Nsenga

Date: 02/03/2023

Signature 

Co-Supervisor: Dr Ignace Gatare

Date:

Signature:

The Head of Master's Study and Training

Dr. James Rwigema

Signature:

Date:

Signature……

Abstract

Vehicle based logistics are hinged on their ability to timely deliver goods, services, and people. The classical expression of “time is money” comes alive in the logistics industry yielding potentially huge financial and health consequences in case of missing deadlines. This is especially the case for time sensitive pharmaceuticals, delivery of perishable goods, delivery of people travelling, delivery of services in fault fixing/recovery sector. All these use cases motivate the need for an immutable, secure, and immortalized process of tracking time. To solve this challenge, this thesis presents prototype-based research that integrates the 4th industrial revolution technologies of vision Internet of Things (IoT), Artificial Intelligence (AI) Object detection, Optical Character Recognition (OCR) and blockchain. The developed prototype features a Raspberry-PI board embedding a camera, an Artificial Intelligence (AI) model to recognize plate letters from the image and a crypto wallet to sign the logging of plate number and time events on the NEAR blockchain, an emerging sharded, layer-one blockchain that uses proof-of-stake and is simple to use, secure and scalable. The effective operation of the developed prototype has been validated inside a campus parking and shows an accuracy range of 92-98%. The benefits of transparency, security, and immutability of the blockchain combined with the intelligence, data capture, and processing of IoT will help to improve the development of accountable solutions trusted by all different logistic stakeholders. As one of the main achievements of this research, a related paper has just been accepted for publication in the conference 8th International Conference on Machine Learning Technologies (ICMLT 2023).

Table of Content

Contents

List of Figures

List of Acronyms

IoT- Internet of things

SDK- Software Development Kit

ML- Machine Learning

OCR – Optical Character Recognition

CLI- Command Line Interface

JWT- Json Web Token

OAuth – Open Authorization

ID - Identification

Chapter 1: Introduction

We are currently in the time frame considered to be leading into the 4th industrial revolution. New emerging technologies are being developed and matured enabling us to effectively harness them to make our lives better. Blockchain and Internet of Things (IoT) are considered part of this revolution [1].

The Internet of Things looks at connecting everyday objects to the internet. Large amounts of data are collected using sensors and pre-processed using microcontrollers, actuators act upon the environment IoT systems are deployed. Data collected is sent to the cloud using communication devices attached to microcontrollers for processing and data modelling. This data is relevant in data science fields and artificial intelligence to learn and understand our environments. [2]

Blockchain is a distributed ledger technology that allows for peer networked nodes to share and track assets on a decentralized network. The blockchain-based ledger is immutable and require consensus algorithms to verify and allow for the data to be allowed on the network, data is stored in blocks and chained together to form part of the distributed ledger and decentralized network. This has several benefits like safety, it is computationally expensive to hack and change data on a blockchain, data is transparent and open [3,4].

Transport and logistics are a critical part of today's society. Late arrival of transport vehicles, whether buses trains, planes, and even last mile transport systems like motorbikes can have economic and social effects [5]. For instance, businesses and organizations suffer economic effects from late arrival and departure times of staff to their workplaces [6]. A study on lateness of staff in the US estimated a financial cost of 737 dollars per employee per year for mid-sized tech companies in 2002. This value may have tripled to reflect current market value; indirect costs may include loss of motivation and morale among other staff [7, 6]. Other business sectors such as pharmaceuticals have time constraints that may even impact the health safety of people such as timely delivery of perishable goods and critical pharmaceutical drugs and test samples [8,9]. As stated in [10], recording of vehicle plates, departure and arrival times can help to improve safety and timely delivery of drugs. The aforementioned areas need clear open and immutable ways of quantifying times of arrivals and departures, quantifying the value of deadlines being missed or

met, and the transfer of the cost when times are not met. These critical data need to be stored in a transparent manner that is verifiable and auditable by all parties concerned.

Advances in Artificial Intelligence (AI) and vision Internet of Things (IoT) sensing make it possible for edge devices to be able to use imaging to identify and classify objects, detect features and group objects which can then be used for image processing and decision making. In our case, AI models can be trained to detect the location of number plates on a vehicle's image, then use AI again to recognize letters from number plates from the plate image. Using wireless communication protocols, this information can be submitted to a smart contract over a blockchain for further analysis and reading [6].

This thesis presents prototype-driven research to prove the concept of how the three industrial revolution technologies leading to the 4th generation, namely IoT, AI and Blockchain, can be integrated to develop an immutable tracing of departure and arrival times of vehicles in a logistic value chain. The outcome prototype features a Raspberry-PI board embedding a camera, an Artificial Intelligence (AI) model to recognize plate letters from the image and a crypto wallet to sign the logging of time events on the NEAR blockchain, a sharded, layer-one blockchain that uses proof-of-stake and is simple to use, secure and scalable.

1.1Problem Statement

Vehicle entry and exit from facilities or stations can be tracked through number plate detection using various techniques. Myriad industries, services and personnel stand to benefit from the tracking of vehicle times and arrivals. For example, vehicles that are part of logistic chains or systems that allow for time sensitive delivery of goods are a key consideration; this might include pharmaceutical delivery vehicles, transport of perishable consumables, or last mile delivery. Moreover, vehicles that are part of core organizational needs, such as service operators, network engineers, fault responders, and transport systems could utilize such a system to render their services more efficient, effective, and accessible [11, 12].

IoT edge devices can be an effective way of tracking these vehicles via plate recognition. Using algorithms that detect contours in images of vehicles, plates can be detected and the position of

the number plate in such an image can be accurately determined. The position of the plate can then be cropped and magnified, then using OCR techniques characters of the vehicle plate can be read and stored for future use [13,14].

Moreover, AI models can be developed, trained, and deployed to an edge device for plate detection. Over time, object detection models have been developed for different objects and these models can be used in inference learning to be specified and adapted towards identifying number plates. The OCR techniques can be used to read number plate details from the cropped image in the location where the number plate is detected with a high accuracy [15,16].

However, there are some challenges and limitations to using a centralized Web2 database to store departure and arrival times. The main challenge of a centralized database is data integrity since data can be modified/changed if one has administrative rights [17]. Additionally, given that it is centralized, it means the database is at risk when faults happen as the central point remains the point of truth [18]. Centralized are open to malware and malicious attacks from hackers and the information is not typically open and transparent [17]. The value of this system and extra use cases depends on the transparency and safety of the data. When a value is placed to the cost or gain of time lost or made from departure and arrival times of vehicles, the collected data requires guarantee to be without fault or interference [19].

Thanks to its decentralized feature, Web 3 technologies enable open verification natively. Indeed, Web3/Blockchain solutions offer storage that is distributed which makes it secure and reliable [20]. Data stored on the blockchain is immutable and therefore can be trusted to be effective since it is open and transparent in nature. Blockchain technologies using smart contracts inherently attach value to transactions on the blockchain for reading and writing, the value can be transferred between entities and/or ‘things’ within the system, which means faults and penalties can be transferred without bias inherently on the blockchain. Gains/losses realized can also be quantified inherently on the platform and then at a later stage mapped to a stable coin or currency value for payments [21].

1.2Study Objectives

1.2.1 General Objective

This MSc research projects seeks to demonstrate the feasibility of arrival and departure times open verification in delay-sensitive transportation services

1.2.2 Specific Objectives

To achieve the main goal the following specific objectives will be used as guides:

- SO1: To identify a transportation vehicle at a given geo-location by automatically reading its plate number
- SO2: To verify departure and arrival timing compliance using transportation blockchain smart contracts
- SO3: To validate the good operation of the prototyped end to end system at the university entry gate

The above specific objectives intend to respond the following research questions:

- RQ1: How do we identify and collect arrival and departure times of a vehicle by reading its number plate using IoT and AI?
- RQ2: How do we develop an open verification ecosystem for tracing timing contract failures using Blockchain?
- RQ3: How do we confirm that the integration of IoT, AI and Blockchain can provide a working solution for open verification in delay-sensitive transportation services?

1.3 Hypotheses

The general hypothesis of this study is that value can be attached to recording time in an immutable manner.

Time can be captured in an immutable, safe, and open manner and the value ascribed to this time factor can be determined by a community of system users.

A vision base IoT system linked to a blockchain with a smart contract can be used to store a time factor and determine its value for users.

1.4 Study Scope

The study focus of this research includes:

1. Developing a vision enabled IoT device using a web camera and OpenCV for pragmatic use of an edge device to capture images and then program the IoT device to manipulate and extract data from the image.
2. Use of machine learning (ML) model to detect objects in images for more accurate object detection which leads to more accurate extraction of data from the image.
3. Studying blockchain smart contracts and the benefits of blockchain systems in the logistics industry and how the benefits impact day to day lives of system users.

1.5 Significance of the Study

This study aims to demonstrate a viable use case for using IoT and blockchain technology.

Time and vehicle plate date being store in a secure, immutable manner will be relevant for future systems that require security, like pharmaceuticals

The open nature of the data stored, and group verification of the data makes blockchain suitable for instances of goods that affect communities like water supply or perishable goods

Departure and arrival times being recorded and tracked over time with penalties and rewards built into fair smart contracts that are not centrally controlled will lead to more effective transport systems and trust from users.

1.6 Organization of the Study

Chapter one introduces the project, aims and objectives of the prototype-based research. Chapter two discusses the literature review and gaps identified to achieve the aims of this project. Chapter three looks at the methodology used to carry out the research. Chapter four discusses the system design, simulation, and prototype development. Chapter five discusses the results and findings from the study and finally chapter 6 explores challenges. Limitations, recommendations, and conclusions from the prototype-based research.

1.7 Conclusion

This chapter presents the introduction of this study, problem statements and justification for this study. The integration of these two emerging technologies will be critical to fully harness the benefits of their existence. The intrinsic value of blockchains make the storing of time data valuable, this time specific data can be captured using smart IoT devices at the computing edge of the logistics value chain for vehicles transporting goods

Chapter 2: Literature Review

The internet of things (IoT) allows for a myriad of devices to be connected, collect data, and make intelligent decisions [22]. IoT however suffers from vulnerability of privacy and security breaches. Blockchain technology can be used to address these concerns. It has the inert capabilities of transparency, immutability, and data encryption [22]. Blockchain technology has also been used in the creation of digital currencies, which makes it suitable for transferring value between connected nodes/things, one of its most famous applications being Bitcoin [23]

The integration of IoT and blockchain also has its limitations, because of the consensus algorithms used in public blockchains for Proof of work or Proof of stake the computational cost of using a blockchain and the cost of data storage can be high [24]. Therefore, decisions must be made on what data to store and read from blockchain.

Previous researchers have looked at the subject matter of blockchain and integration with IOT in different fields and applications and from an overarching point of view, we will look at some of these to see the lessons learnt and apply to our experimental project research.

An overarching paper on blockchain and IoT integration highlights the use cases of blockchain and IOT in different uses cases and seeks to provide technical support for parties interested in deploying Blockchain IOT projects (BIoT) [25].

The work highlighted the distributed ledger technology of blockchain and how these have the characteristics of making the network Decentralized as nodes work together to make the system distribute, trustless; nodes are capable of trusting each other as the system runs on full transparency, Collective maintenance; the system is not owned by no one specific the nodes are responsible for maintain the system and exercising the consensus algorithms, Reliable database; every node has a copy of the ledger, the reliability increases as the network grows, Anonymity, there is no need to trust in the network so nodes can remain anonymous.

Blockchains because of their trust and distributed nature can be used to create crypto currencies. They also have ability of storing smart contracts, which are self-executing pieces of code that live on the blockchain. The paper analyzes different blockchain IOT infrastructure design architectures. IoT traditionally sent data to the cloud for supercomputer processing and data storage.

A hybrid architecture was interesting in capturing the current benefits of cloud computing in IOT and blockchain benefits for verification and distributed applications.

Some challenges noted from the paper for the integration of blockchain and IOT applications were scalability, processing power and time, storage sizes and cost of running such networks at large scales. Some recommendations were made around the technical challenges of decentralization, blockchain infrastructure, government, regulations, and legal aspects of use of blockchain. The most important contribution was around the hybrid architecture considering that IoT and blockchain are arguably in their infancy, as they grow more solutions will be solved and their integration will be more seamless.

Their conclusion predicts that IoT is here to stay and that Blockchain IoT integrations will become more widespread and that there isn't a one size fit all answer for these integrations and so implementors must weigh their options and make decisions accordingly.

When deciding which data to store and read from the blockchain we look at an IoT-Health solution that integrates with blockchain.

A study proposed for the "Healthchain" a use case for IOT and Blockchain in health care system [26]. IOT technology has be used in health care to improve efficiency and accuracy, break geographical restrictions with remote monitoring, conduct disease risk assessment and construct disease prediction. The amount of data generated can be large and therefore needs cloud assisted health care systems to data and for further processing. Large scale smart health devices need high computing and the storage of cloud servers, but this can lead to centralized systems that can be affected by attacks. Health data is highly sensitive and should be well protected. Medical disputes require trust from all parties that data was not tampered with.

This work proposed the use of blockchain technology to cover most of these problems. The paper proposed Healthchain, a blockchain-based privacy preserving scheme for health data. In Healthchain, users can periodically upload the health data collected by IoT devices and publish them as a transaction. Doctors or artificial intelligence (AI) health analyzers can diagnose anytime and anywhere based on the IoT data and publishes the diagnosis as a transaction.

Blockchain technology is not designed to store large amounts of data so recording of a user's complete data will not be necessary and will lead to computational problems when maintaining, searching, or verifying.

The work proposed InterPlanetary File System (IPFS), which is a content-addressable, distributed file system to store data with high integrity and resiliency.

Data is distributed and stored in different nodes all over the internet, they have no single point of failure and efficiently distribute large amounts of data without duplications. Hash string of data is stored on the blockchain and mapped out to the IPFS storage.

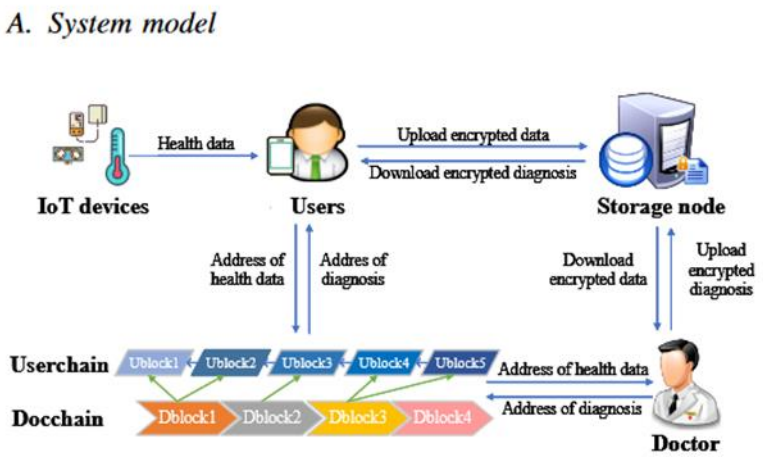

Figure 1: Health Chain System Model

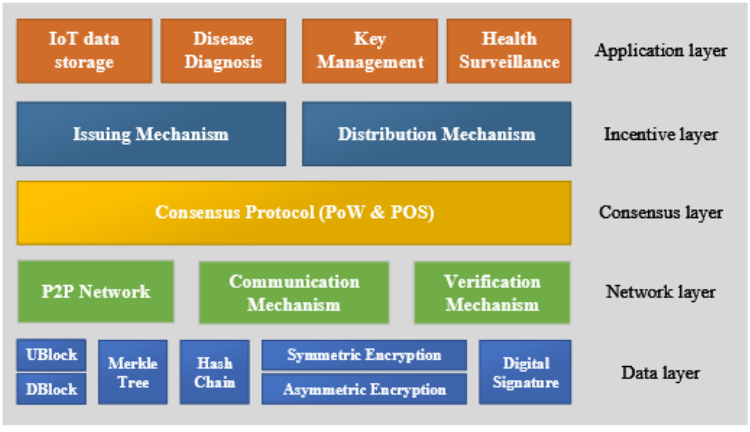

Figure 2:Health Chain architecture diagram

IoT devices are used to track the data of the user and they are connected to one user node which aggregates the data and sends it to the user chain. They can only generate and publish data. The doctor nodes can be doctors or artificial intelligence and can be used to diagnosis of data that is received from the user blockchain. Accounting nodes are deployed by the consortium to verify doctor nodes' transactions are correct and valid. Storage nodes are used to store data. The Userchain and Docchain are the blockchains used to store transactions and data.

The rest of the paper shows the algorithms used to add patients and doctors and dives into evaluations of usage of traditional systems to the proposed solution. The paper shows how IOT and blockchain can be integrated in the space of smart health care where data is being generated in large quantities and needs to be kept secure and still manageable for analysis, processed for reactions at IOT layer and even used in cases of disputes.

Recent research has also been conducted in the storage of vehicle On-Board diagnostics (OBD) data to the blockchain using an InterPlanetary File System (IPFS)[27].

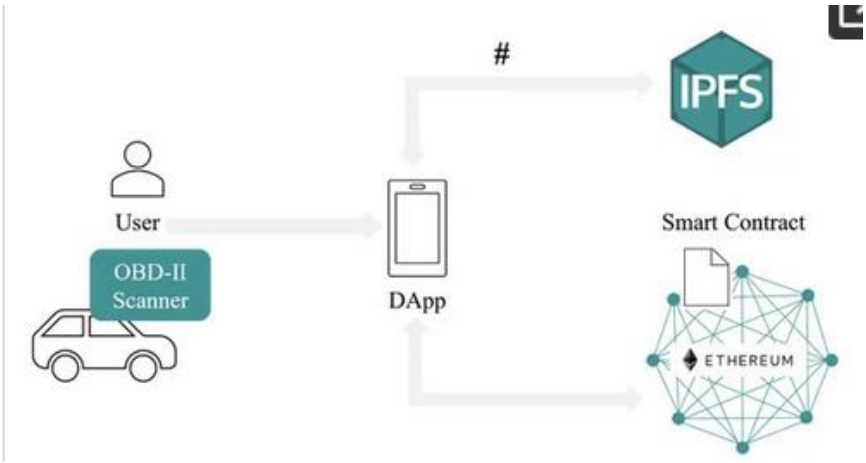

Figure 3: System architecture

The proposed system uses an OBD scanner to read vehicle data and a Decentralized application that connects to a smart contract and the IPFS. The smart contract makes decisions on what data to store and read from the file system. The data recorded is encrypted and safe, it is also distributed across multiple nodes which makes it hard to breach. These benefits come with a high cost to upload data and the need to filter the data to store the relevant data on the blockchain.

In India a study was done using SOBEL Edge detection techniques to detect number plates in cases of traffic offenses and parking management [28]. Edge detection techniques are used to separate license plates from segments of images before optical character recognition techniques are used to extract characters from number plates [28]. The study showed how the IoT system could be integrated with camera's and send processed data to a database for further review. Edge detection techniques can however struggle with multiple objects in an image.

A study done for "Automatic License Plate Recognition (ALPR)" showed high accuracy for object detection using a machine learning object detection model using YOLO detector. The object model was fed a large data set of images and trained to detect license plates from images, this increased accuracy of plate detection and character reading from images.[29]

Similar work involved the use of 'Open License Plate Detection for private parking systems" [30]. The benefits of image detection of license plates from IoT systems were reflected in a proposed solution for parking management.

Smart contracts have been used to transfer value in popular blockchain applications like Bitcoin [31]. Smart contracts using the blockchain offer a trustworthy distributed peer to peer system that is resilient and can be audited [31]. Smart contract payments have been used in construction projects where progress data is captured by unmanned aerial vehicles and sent to a smart contract for disbursement of payments [32]

The blockchain provides data assurance and resilience for data collected from IoT systems [33]. Users will need to trust proposed systems if the system have rewards and fines involved. The transfer of value, including monetary value hinges on the accuracy and reliability of data being used.

From the above studies, we see examples of IoT and blockchain integration solutions. Factors to consider include:

- An emphasis on the type and volume of data that can and should be stored on the blockchain.

- The stage of the IoT system pipeline where the blockchain integration should happen is considered, whether at sensor level, pre-processing, storage, or analysis.
- The stage of the IoT system pipeline where the blockchain integration should happen is considered, whether at sensor level, pre-processing, storage, or analysis.
- The transfer of value between users of the system.

Chapter 3: Research Methodology

In this chapter the methods and approaches used to conduct the study are outlined. This includes the steps undertaken to complete the study, the research process/ideology and tools used in prototyping of the proposed solution.

3.1 Research Process

The research process was first initiated to investigate how to incorporate IoT technologies and Blockchain technology. The prototype research was then narrowed down after comprehensive literature review and business case analysis to vision integrated IoT integrated with the NEAR blockchain to immutably record vehicle departure and arrival times in the logistics sector.

Proposed prototype steps:

i. Literature review
ii. State of the art analysis
iii. Smart contract design, functions, and storage
iv. Set up prototype with Raspberry Pi 3B+ with Canny method
v. Set up prototype with Raspberry Pi 4 with YoloV4 MODEL
vi. System test and debugging
vii. Blockchain submission analysis
viii. Results analysis
ix. Thesis preparations and publication of analysis

3.2 Research Design

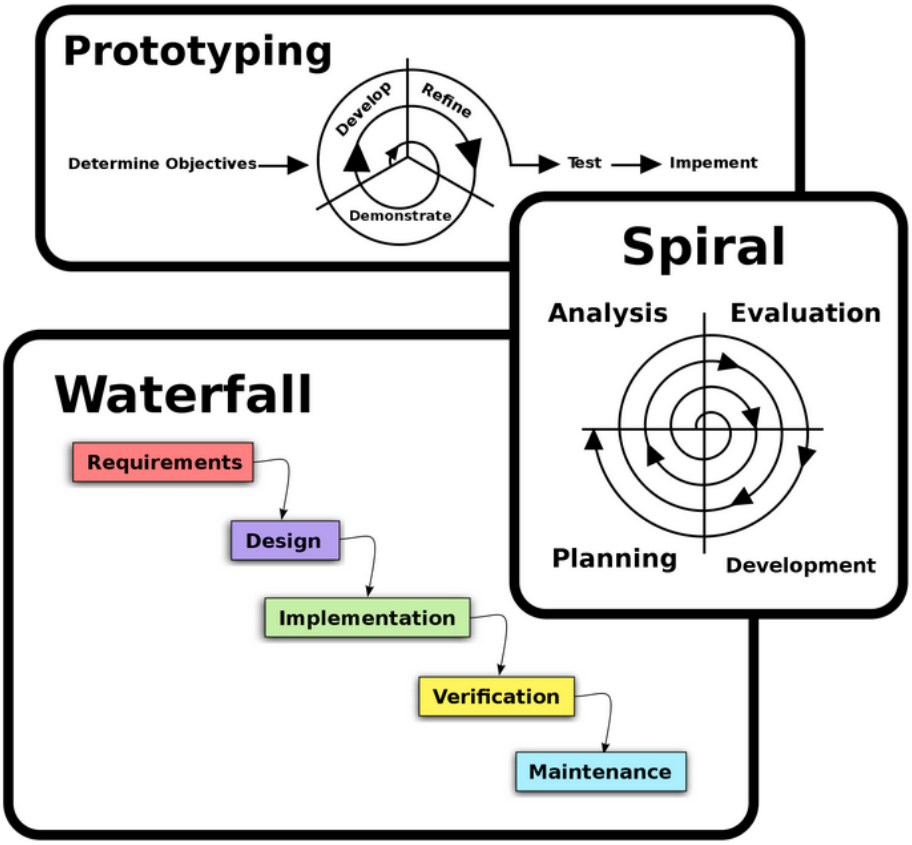

Figure 4: Design Concept models [34,35]

The requirements of the projected are collected then the design of the system is developed. The implementation of the research project follows the prototyping system of developing, refining the idea, demonstrating then iterating from the feedback collected.

1. Requirements

In this phase, we collect all the requirements based on the problems statement and get feedback from the stakeholders and other service providers involved in this study. These requirements are analyzed, and the system requirements documented.

2. System Design:

This is the phase where the system level design is done. This includes the system architecture, block diagram and prototype design.

3. Implementation

This is where we combine the input data and output data to create a complete system. It involves the development of the different prototypes of the system for vision integration and license plate detection. This section involves both the hardware configuration and software programming to create a working prototype that implements the end-to-end design.

4. Testing

This is the phase where the system is tested in an experimental environment to simulate vehicle positioning.

5. Analysis

We analyze the system and make recommendations for future applications.

Chapter 4: System Analysis and Design

In this chapter we look at the overview of the prototype from a block diagram then we dissect the internal workings of the system using an end-to-end architectural design. Then we discuss the design of the smart contract, techniques used for plate detection and image manipulation. We discuss some of the libraries used for text extraction and the specific components used in the prototype.

The overview of the prototype is consisting of a web camera as an input device to a microcontroller. The microcontroller sends data to a blockchain using an inbuilt Wi-Fi module that connects to the internet. The microcontroller is supplied with power to make it a mobile unit that can be used on the processing edge.

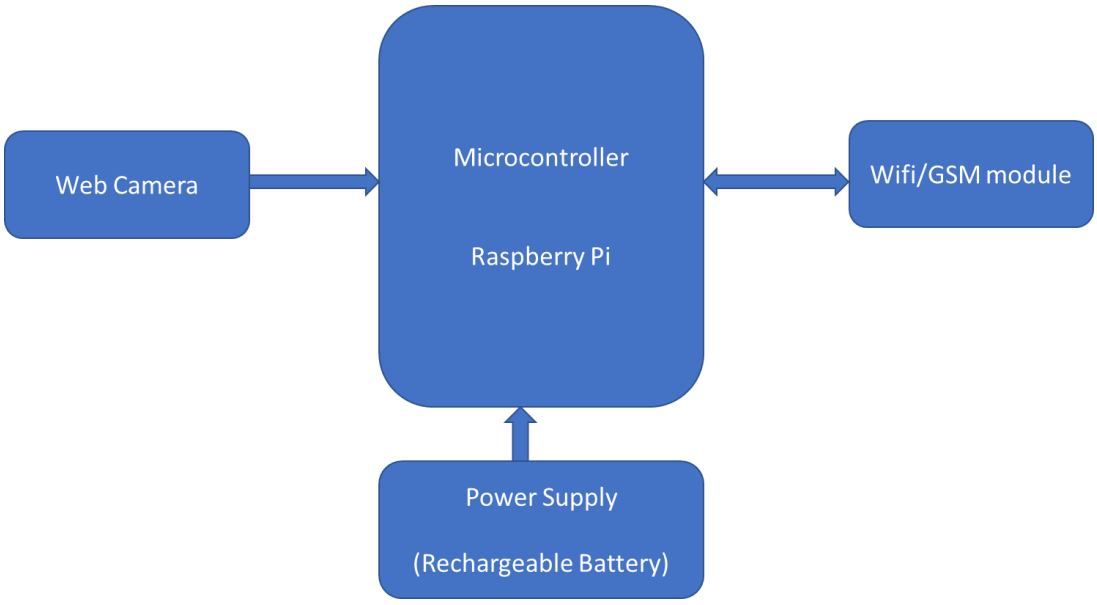

Figure 5: System Block diagram

4.1. System architecture

Figure 5 shows the end-to-end architecture that incorporates a vision sensor (camera), algorithms, and AI models to detect plate numbers. Once the plate numbers have been identified, the system then submits the details to a smart contract on the NEAR blockchain, to be read and written.

The NEAR protocol is a blockchain that uses proof-of-stake and is simple to use, secure and scalable. [36]. NEAR differentiates itself by being developer friends and offers scalability via ‘sharding’ and shared security

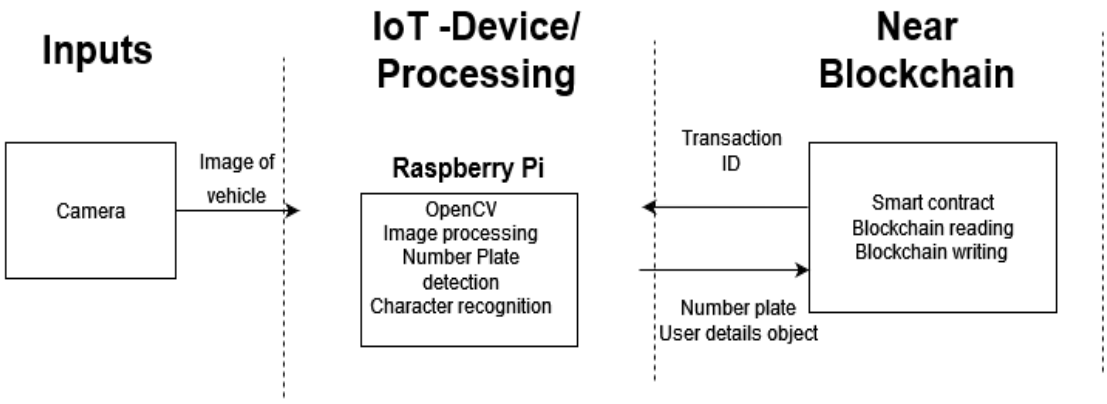

Figure 6: End-to-end prototype system design

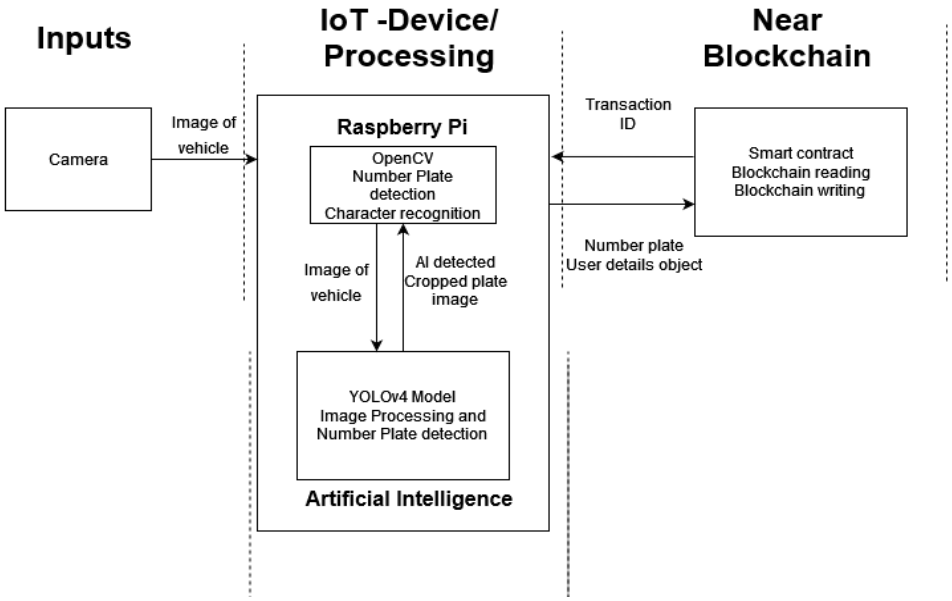

Figure 7: Full system design with AI model

4.2. Blockchain Smart Contract Design

The smart contract has 2 states, you can either evoke a call or a view method on the blockchain. Call methods are primarily used to write to the blockchain, but they can also be used to read and write to the blockchain. These would be considered two transactions, even though the same method makes the process call. Our smart contract has a call method to record the number plate details. The view method is used to read from the blockchain, and our smart contracts evoke several methods to read from the smart contract. We read details of recent/last transaction, request transactions of a specific vehicle and of a specific group of vehicles listed as part of a fleet.

```
1  import {logging} from "near-sdk-as"
2  import { CameraData, Position, SpeedData, VehicleData } from "./model";
3
4
5  let cameraData = new CameraData();
6  let vehicleData = new VehicleData();
7  let speedDataObject = new SpeedData(0,"", 0.0);
8
9  export function SubmitPlateDetails(vehiclePlate:string, datetime: f32):void{
10
11     speedDataObject.SubmitPlateDetails(vehiclePlate, datetime);
12     logging.log("Data added to Plate details")
13
14 }
15
16 export function SubmitOverSpeedTransaction(speed:i32,vehiclePlate:string, datetime: f32):void{
17
18     assert(speed > 40,"Speed must be above 40")
19     speedDataObject.SubmitOverSpeedTransaction(speed,vehiclePlate, datetime);
20     logging.log("Data added to overspeeding transactions")
21
22 }
23
```

Figure 8: Contract code with Call function for submitting plate details

```
SubmitPlateDetails(vehiclePlate: string, cameraTime: f32):string{
    overSpeedTransaction;


    this.vehiclePlate = vehiclePlate;
    this.cameraTime =cameraTime;
    this.signedTime = context.blockTimestamp;
   this.cameraId = context.sender;

    let map = new Map<string, string>();
    map.set("vehiclePlate", `${vehiclePlate}`);
    map.set("cameraId", `${this.cameraId}`);
    map.set("cameraTime", `${this.cameraTime}`);
    map.set("signedTime", `${this.signedTime}`);

    //add map data to vector
   plateDetails.push(map);

    logging.log("Data added to plate details");

    return "Data added to plate details";
}
```

Figure 9: Code extract with Submit Plate Details Model

```
import { PersistentMap, PersistentVector } from "near-sdk-as";
import { Position } from "./model";

export const  plateDetails = new PersistentVector<Map<string,string>>("plateDetails");
```

Figure 10: Code extract with plate details storage definition

The Smart contract [37] is deployed over the NEAR protocol using the Assembly Script SDK [39], which is a TypeScript/JavaScript like language that compiles down to WebAssembly.]. Smart contracts developed in WebAssembly are then deployed on top of the protocol and can be used to make calls to the blockchain to read or write [39].

The NEAR protocol employs ‘sharding’ for processing every single transaction. This solution was proposed to solve problems with other blockchains which are noted as having low throughput, since each node would process each single transaction, which would in turn lead to high ‘gas’

prices (the cost of using the blockchain). 'Sharding' helps to solve the high cost of blockchain storage, bandwidth, and processing by splitting tasks to smaller processing units, this makes it easier to scale and affordable [40]. The NEAR protocol provides both main accounts and test accounts for developing concepts and ideas before launching. These come with wallets and explorers for tracking transactions that are happening on the blockchain and smart contract specific activity.

4.3. Vision IoT Device Design

The Internet of Things (IoT) defines a network of thing that are attached with sensors, run software, for the purpose of connecting and exchanging data with other devices/things and systems over the internet [41]. IoT devices will typically have a control/processing unit to handle edge processing of data collected from sensors and transfer data collected and processed using communication devices and different protocols to other devices, processing systems or cloud storage over the internet. In this research we used two processing units to simulate computer vision and transferring of data from processor to the internet.

4.3.1. Process one:

A Raspberry Pi 3B+ was used in conjunction with a webcam for computer vision to read images. The Raspberry Pi is a low cost, credit-card sized computer. It is a capable, compact device that enables edge computing and programming with Python [42]. Using the Python programming language and a library called OpenCV, we captured images from a webcam and stored them for further processing. The images were run through an algorithm built off the Canny Edge Method to mark out the edges in the image. After edge detection, a list of contours is detected then subsequently filtered to detect rectangle shapes and closed figure among the obtained results [43]. After detecting the number plate, the boundaries of the number plate are masked and then the image is cropped out prepping it for Optical Character Recognition. The Pytesseract library is used to read characters on an image. It has a function for reading images to strings [44]. The string read is stored as the number plate value before making a call to the smart contract using the values detected and overloaded for time and device details. The NEAR protocol provides a command line interface for deploying smart contracts, adding keys associated with accounts and smart contracts,

and calling methods in smart contracts [45]. The methods can be evoked from an operating system CLI call from a python script. Calls can also be made through an http request or gRPC call. The blockchain responds with a transaction ID if there are no errors or connection problems that can be viewed on the NEAR transaction explorer.

4.3.2. Process two:

The second process involved adding an artificial intelligence model for number plate detection. A Raspberry Pi 4 was utilized, with a TensorFlow lite model that was pre-trained with images of vehicles and bounding values for number plates. Inference learning techniques were used to develop the TensorFlow lite model that was developed on top of the YOLOV4 object detection model [46]. YOLOV4 provides the benefits of faster and more accurate object detection. Weights can be adjusted to allow for detection of objects of distinctive features; in this case the weights pertain to number plate features. Images are taken from a webcam using the OpenCV library in a python script, and the image is then fed to a TensorFlow lite model for number plate detection. Once the plate is detected, the bounding area of the number plate in the image is masked and cropped. The cropped image is then run through the PyTesseract optical character recognition library for reading the characters off the cropped image. Once the characters are read and stored in a variable, the script calls the smart contract to write the value of the plate and overhead data for raspberry pi and vehicle details for storage on the NEAR protocol blockchain.

4.3.3. Component Overview

1. Input device

Web Camera: Logitech C510 HD 720

Figure 11: Web Camera

Logitech®

HD Webcam C510

TECHNICAL SPECIFICATIONS

- HD video calling (1280 x 720 pixels) with recommended system
- Video capture: up to 1280 x 720 pixels
- Logitech More HD technology
- Photos: up to 8 megapixels (software enhanced)
- Built-in mic with Logitech RightSound™ technology
- Hi-Speed USB 2.0 certified (recommended)
- Universal clip fits laptops, LCD or CRT monitors

Logitech webcam software:

- Logitech Vid™ HD
- Logitech RightLight™ 2 technology
- Video and photo capture
- Magix™ photo and video editing
- 1-click Facebook™ and YouTube™ HD-upload (registration required)
- Logitech® Video Effects™: fun filters, avatars, face accessories, video masks and mask maker

Works with most instant messaging applications
1 Intel® Pentium® 4 (2.8 GHz) recommended.

Figure 12: Technical specifications of the Web Camera

2. Processors

Microcontrollers:

- Raspberry Pi3B+

Figure 13: Raspberry Pi3B+

The Raspberry Pi 3 Model B+ is the final revision in the Raspberry Pi 3 range.

- Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz
- 1GB LPDDR2 SDRAM
- 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE
- Gigabit Ethernet over USB 2.0 (maximum throughput 300 Mbps)
- Extended 40-pin GPIO header
- Full-size HDMI
- 4 USB 2.0 ports
- CSI camera port for connecting a Raspberry Pi camera
- DSI display port for connecting a Raspberry Pi touchscreen display
- 4-pole stereo output and composite video port
- Micro SD port for loading your operating system and storing data
- 5V/2.5A DC power input
- Power-over-Ethernet (PoE) support (requires separate PoE HAT)

Figure 14: Raspberry Pi3B+ specifications

Raspberry Pi 3B+ vs Raspberry Pi 3B

No	Specification	Raspberry Pi 3B+	Raspberry Pi 3B
1	Release Date	14th March 2018	29th Feb 2016
2	SOC Type (Processor)	Broadcom BCM2837B0 (with metal cover)	Broadcom BCM2837
3	Core Type	Cortex-A53 64-bit	
4	No. of Cores	4 Cores	
5	GPU	VideoCore IV	
6	CPU Clock	1.4 GHz	1.2 GHz
7	Memory	microSD	
8	RAM	1 GB LPDDR2	
9	Ethernet	Gigabit over USB 2.0 (Max 300Mbps)	Megabit (Max 100Mbps)
10	USB Port	4 x USB 2.0	
11	HDMI	1 x full size HDMI	
12	WiFi	802.11 b/g/n/ac (Shielded)	802.11 n
13	Bluetooth	4.2 (Shielded)	4.1 LE
14	Antenna	PCB Antenna (Similar to Rpi Zero W)	Chip Antenna
15	GPIO	40 pins	
16	Operating System	Latest Raspbian (> March 2018)	Backward compatible
17	Dimension	85mm x 56mm	
18	POE	Yes, with PoE HAT	No
19	Power Rating	1.13A @ 5V	1. 34A @ 5V

Figure 15: Comparison of the 3B+ vs 3B

- **Raspberry Pi4**

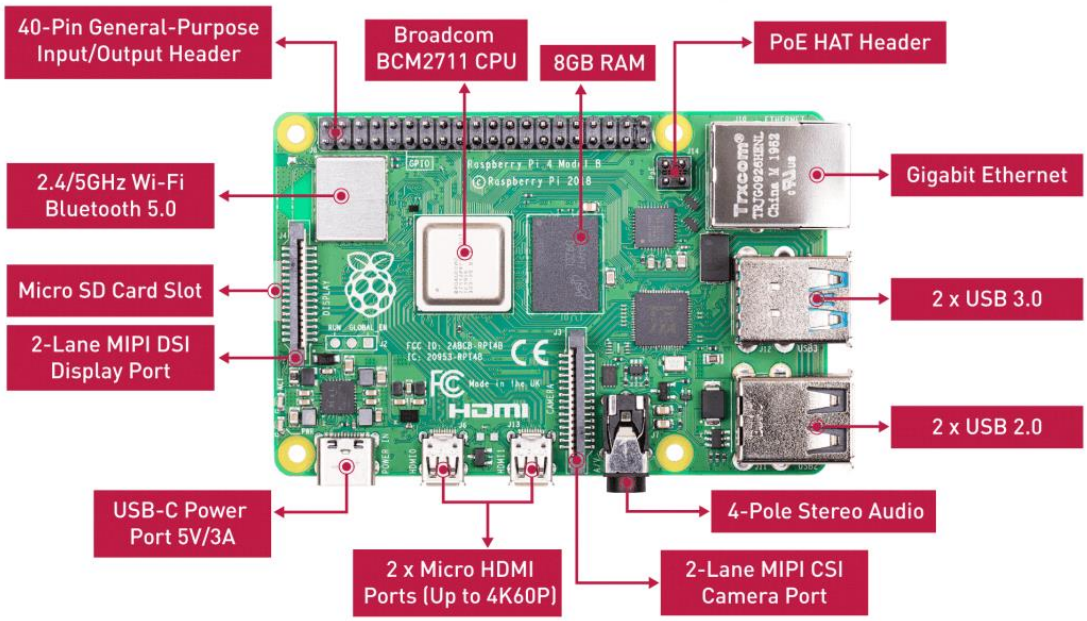

Figure 16: Raspberry Pi4

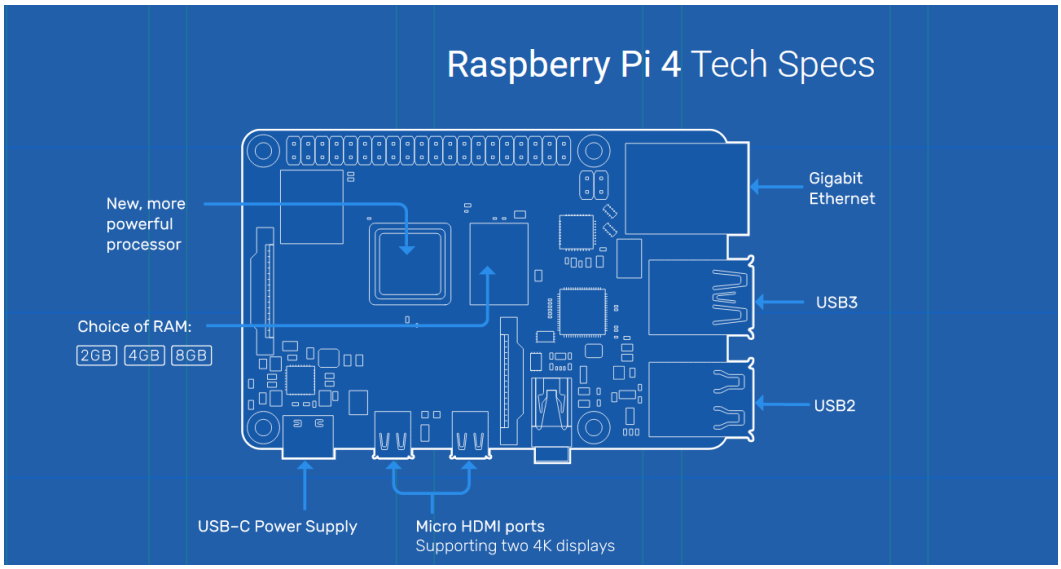

Figure 17:Raspberry Pi4 Technical specifications

- Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
- 1GB, 2GB, 4GB or 8GB LPDDR4-3200 SDRAM (depending on model)
    - GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE
- Gigabit Ethernet
- USB 3.0 ports; 2 USB 2.0 ports.
- Raspberry Pi standard 40 pin GPIO header (fully backwards compatible with previous boards)
- 2-lane MIPI DSI display port
- 2-lane MIPI CSI camera port
- 4-pole stereo audio and composite video port
- H.265 (4kp60 decode), H264 (1080p60 decode, 1080p30 encode)
- OpenGL ES 3.1, Vulkan 1.0
- Micro-SD card slot for loading operating system and data storage
- 5V DC via USB-C connector (minimum 3A*)
- 5V DC via GPIO header (minimum 3A*)
- Power over Ethernet (PoE) enabled (requires separate PoE HAT)
- Operating temperature: 0 – 50 degrees C ambient

* A good quality 2.5A power supply can be used if downstream USB peripherals consume less than 500mA in total.

Figure 18: Raspberry Pi4 specifications

4.4. Overview of License plate detection Techniques

4.4.1. OCR techniques with canny method [47]

License plate detection using the canny edge detection uses image manipulation to find patterns and objects in images.

Open CV (Open-Source Computer Vision) provides a function for canny edge detection.

Canny edge detection is an edge detection Theory developed by John F. Canny. It is a multi-stage algorithm which uses Noise reduction and a function to find the intensity gradient of the image.

Noise reduction is used to remove noise on the image using a 5 x 5 Gaussian filter. Images are converted to a gray scale and the edges are smoothed out.

To find the smoothened edges, the image is filtered with a Sobel kernel in both horizontal and vertical direction. To get angle and direction of each pixel we use the formulas below:

$$Edge\_Gradient\ (G) = \sqrt{G_x^2 + G_y^2}$$

$$Angle\ (\theta) = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$

Equation 1: Angle and direction formulas

The next step is to apply non-maximum suppression. After gradient magnitude and direction, a full scan of image is done to remove any unwanted pixel which may not constitute the edge. Every pixel is checked if it is a local maximum in its neighborhood in the direction of the gradient.

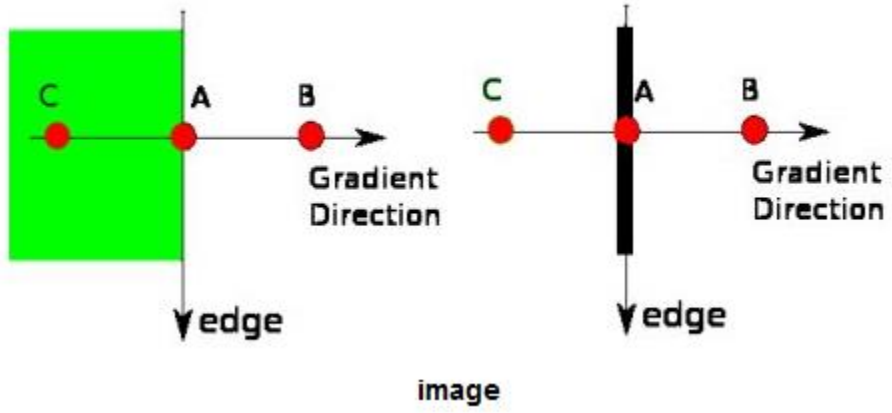

Figure 19: non-maximum suppression

Point A is on the edge (in vertical direction). Gradient direction is normal to the edge. Point B and C are in gradient directions. So, point A is checked with point B and C to see if it forms a local maximum. If so, it is considered for the next stage, otherwise, it is suppressed (put to zero).

The final stage is Hysteresis Thresholding which is the stage used to decide which edges are really edges and which are not. For this, we need two threshold values, minVal and maxVal. Any edges with intensity gradient more than maxVal are sure to be edges and those below minVal are sure to be non-edges, so discarded. Those who lie between these two thresholds are classified edges or

non-edges based on their connectivity. If they are connected to "sure-edge" pixels, they are part of edges. Otherwise, they are also discarded

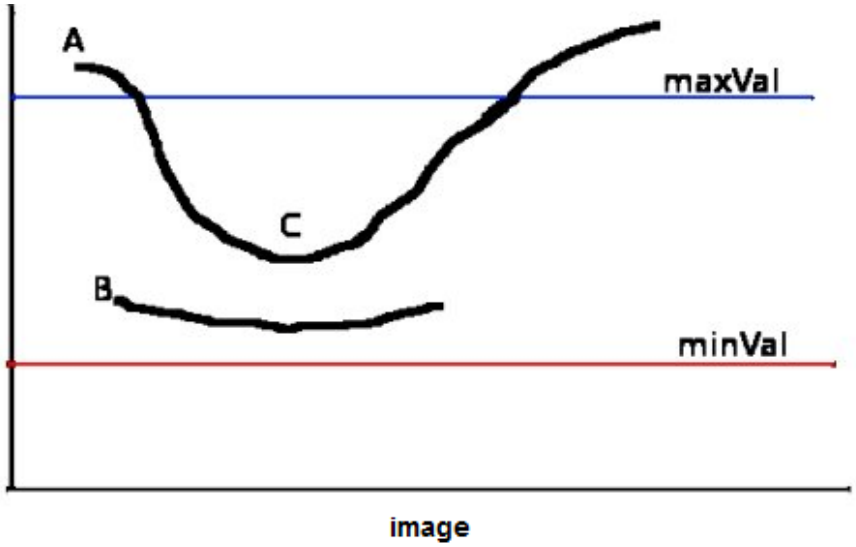

Figure 20: Hysteresis Thresholding

The figure below shows an example of an image processed with edge detection and image results.

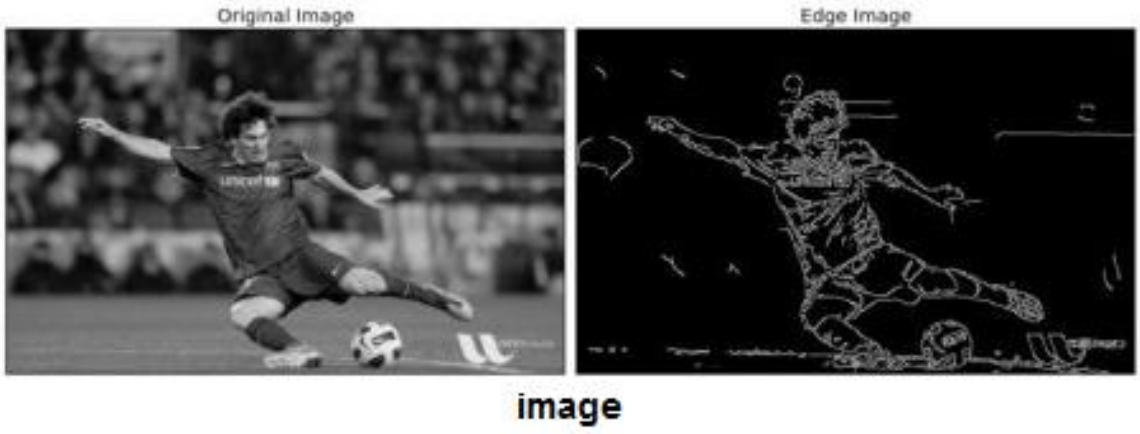

Figure 21: processed image example

In vehicle plate detection the same concept is applied to find contours which can be mapped to rectangles and assumed to be a vehicle number plate.

4.4.2. Vehicle license plate detection using Canny method applied in vehicles overview [48]

An image of a vehicle is taken using a camera and supplied to the program programmatically.

Figure 22: Vehicle image

The image is then converted to a gray scale image using open CV.

```
img = cv2.resize(img, (620,480) )
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) #convert to grey scale
```

Figure 23: Image conversion using openCV

A filter is then applied to the image to sharpen edges and prepare it for the canny method for edge detection.

Figure 24: Filter application (1)

```
gray = cv2.bilateralFilter(gray, 11, 17, 17)
```

Figure 25: Filter application (2)

Figure 26: Vehicle after filter application

```
edged = cv2.Canny(gray, 30, 200) #Perform Edge detection
```

Figure 27: Canny Edge detection

After applying the Canny image processing the preceding image is produced with edges highlighted.

Figure 28: Processed image

Using OpenCV we can now detect contours in our image.

```
nts = cv2.findContours(edged.copy(), cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
cnts = imutils.grab_contours(cnts)
cnts = sorted(cnts, key = cv2.contourArea, reverse = True)[:10]
screenCnt = None
```

Figure 29: Contour detection using openCV

After detecting the contours, we can apply a count over the counters to detect rectangles and ratios that would evaluate to a license plate.

```
# loop over our contours
for c in cnts:
                # approximate the contour
                peri = cv2.arcLength(c, True)
                approx = cv2.approxPolyDP(c, 0.018 * peri, True)
                # if our approximated contour has four points, then
                # we can assume that we have found our screen
                if len(approx) == 4:
                      screenCnt = approx
                      break
```

Figure 30: Rectangles and ratios detection

Figure 31: Plate detection

```
# Masking the part other than the number plate
mask = np.zeros(gray.shape,np.uint8)
new_image = cv2.drawContours(mask,[screenCnt],0,255,-1,)
new_image = cv2.bitwise_and(img,img,mask=mask)
```

Figure 32: Plate masking

After the plate is detected, the bounding area is highlighted, and the section is cropped for text extraction.

Figure 33: Number plate boundaries marked

```
# Now crop
(x, y) = np.where(mask == 255)
(topx, topy) = (np.min(x), np.min(y))
(bottomx, bottomy) = (np.max(x), np.max(y))
Cropped = gray[topx:bottomx+1, topy:bottomy+1]
```

Figure 34:Cropping code

Figure 35:Extracted number plate image

```
#Read the number plate
text = pytesseract.image_to_string(Cropped, config='--psm 11')
print("Detected Number is:",text)
```

Figure 36: Image to text conversion code

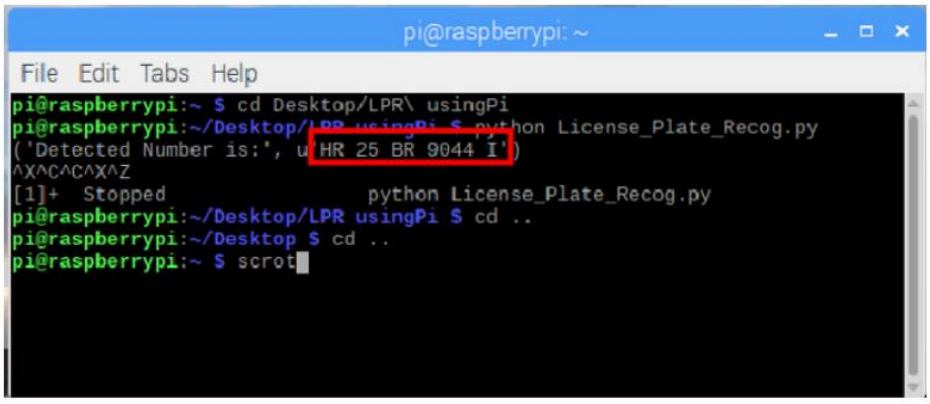

Figure 37: Result from Pi3b+ of number plate

After text extraction the details are ready for submitting to the blockchain.

4.4.3. YOLO V4 explanation/Overview

YOLO or You Only Look Once, is a real-time object detection algorithm. YOLO uses a single neural network to perform both classification and prediction of bounding boxes for detected objects. it is heavily optimized for detection performance and is fast. It works by repurposing traditional image classifiers to be used for the regression task of identifying bounding boxes for objects.

The Original YOLO — YOLO was the first object detection network to combine the problem of drawing bounding boxes and identifying class labels in one end-to-end differentiable network.

YOLOv2 and Yolov4 presented incremental steps in the detection model.

YOLOv4 is the model used in this research. The building blocks of YOLOv4 include inputs of images and patches. The “Backbone” uses different engines including CSPDarknet53, SpineNet, “Neck” for collecting features using path-aggregation blocks and “Head” for final object detection using dense prediction or Sparse prediction (Faster R-CNN).

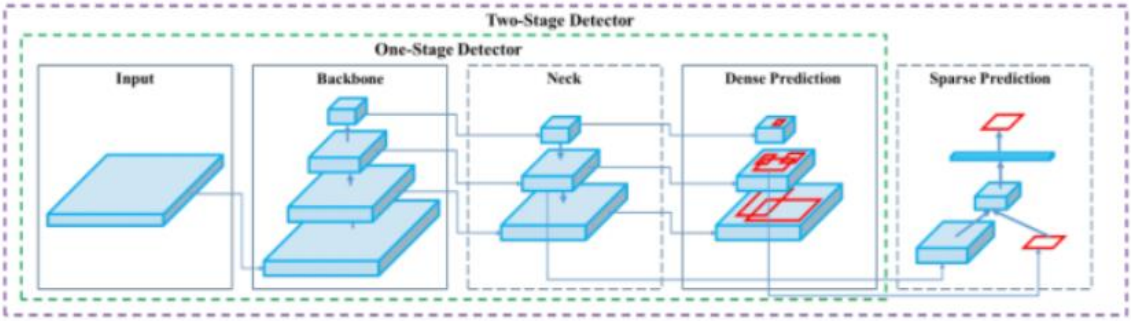

Figure 38: YoloV4 two stage detector

Backbone is the deep learning architecture that acts as a feature extractor. The Neck collects feature maps from different stages of the backbone. Head is the object detector that gives the co-ordinate to the location of the object in the image. Yolov4 helps to customize the model for object detection and can be adapted for the specific features of a license plate.

4.4.4. Library for text extraction

- Pytesseract

Pytesseract or Python-tesseract is an OCR library for python that helps programmers interact with the Tesseract-OCR Engine. It can read and recognize text in images and is commonly used in python ocr image to text use cases.

The Pytesseract library was used to read the characters from the cropped image of the license.

Chapter 5 Results and Analysis

In this section we discuss the different results from different approaches experimented with emphasis on YoloV4 model plate detection and results post submission of details to the blockchain. Post Canny method detection for license plates in images, research was done to develop an ML model using Edge Impulse for plate detection.

5.1. Edge Impulse:

Using Edge Impulses' web application and CLI tool, a target device was added to ease of deployment of final model.

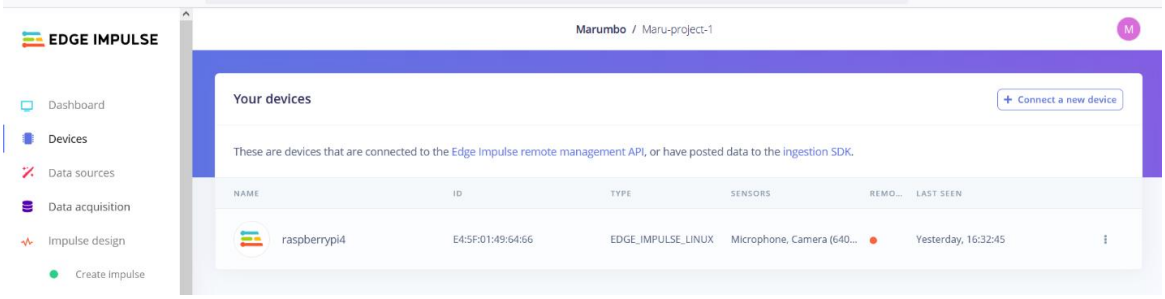

Figure 39: Target device

Initial process involves adding sample images with bounding boxes to mark license plates. The images are used to train and test the model. 301 images were used to train a model and test a model on edge impulse.

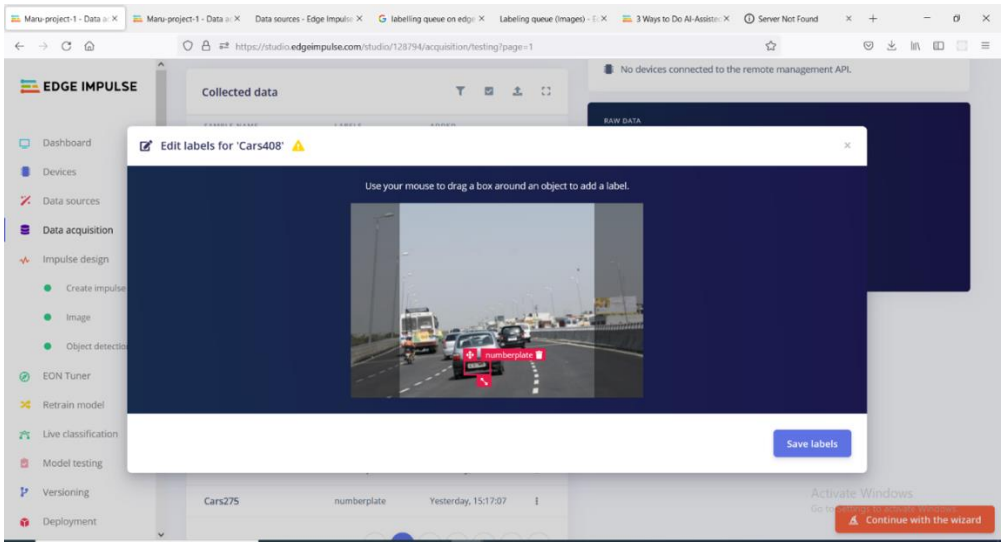

Figure 40:Labelling of training data

Images are split 80/20 for training and testing of the model.

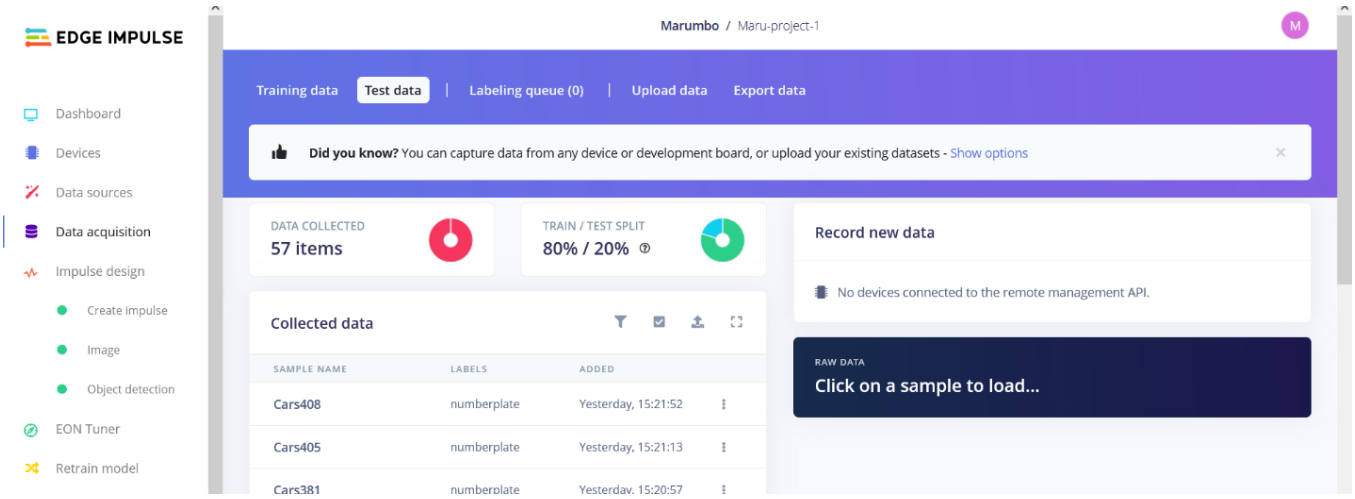

Figure 41: Data acquisition

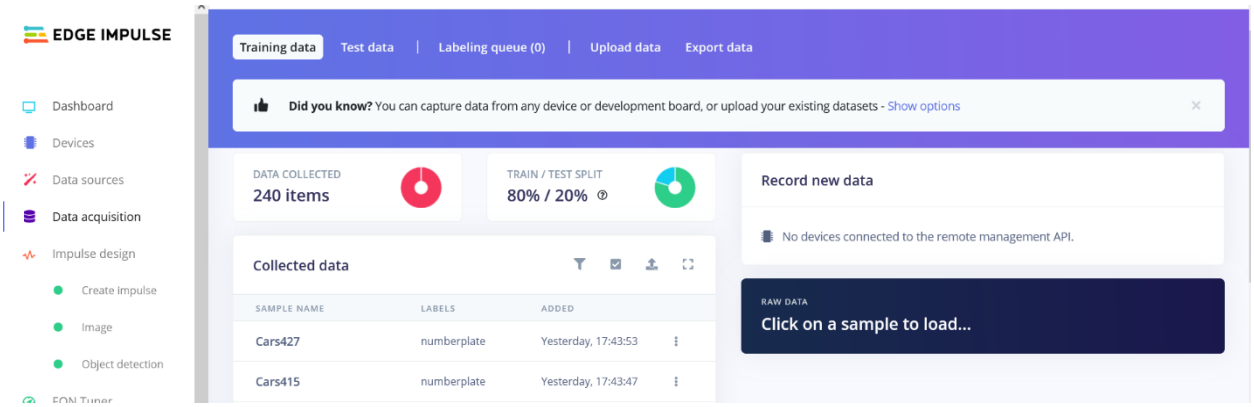

Figure 42: Data splitting

The final output is a Faster Objects More Objects MobileNet model that can be used for object detection.

The best model generated in this research had an accuracy of 75.5%. This accuracy was sufficient for initial round of testing, a YoloV4 was used for higher object detection accuracy.

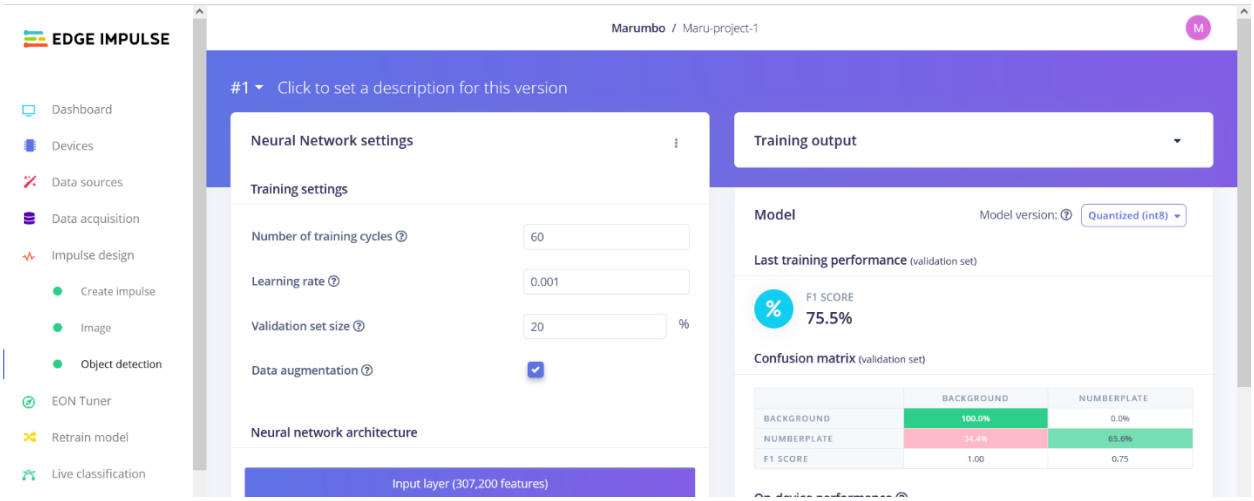

Figure 43: Model training score

5.2. YoloV4 detection

The second phase of the research involved using the YoloV4 object detection model that was customized for plate detection. Accuracy increased to 92%-98%. Below are sample images of different vehicles used to show accuracy of model.

Figure 44: Image of sample vehicle

Figure 45: Image of vehicle with plate detected and confidence levels of model

Figure 46: Cropped image of area with detected plate

Figure 47: Image of second sample vehicles from rear view for plate detection and reading

Figure 48: Cropped image

Using PyTesseract the text from the cropped image is extracted for submission to the blockchain

```
Class: license_plate, Text Extracted: RAE985D

License Plate #:  RAE985D
Object found: license_plate, Confidence: 0.92,
```

Figure 49: Submission to the blockchain

The near protocol provides a command line interface for calling contracts. The Python code uses the operating system functionality to call the contract [20].

- **An example of a call to the contract to submit plate details to the blockchain**

near call $CONTRACT SubmitPlateDetails '{"vehiclePlate":" RAF459B ", "datetime":23344.1234}' --accountId marusichweb3.testnet

- **Sample response from the command line interface**

Scheduling a call: dev-1657889824160-52385125116901.SubmitPlateDetails({"vehiclePlate":" RAF459B ", "datetime":23344.1234})

Doing account.functionCall()

Receipt: 5egLDBBEFMVDQ4ET4uLeQcyehJGtJAPy3QNDvwquHWLU

Log [dev-1657889824160-52385125116901]: Data added to plate details

Log [dev-1657889824160-52385125116901]: Data added to Plate details

Id: 8HoKuRJ3mo46KhYAHxcHSbb5Q3JTtMcLhNaYNuSr9dwP

https://explorer.testnet.near.org/transactions/8HoKuRJ3mo46KhYAHxcHSbb5Q3JTtMcLhNaYNuSr9dwP

"

The transaction Id is useful for tracking contract details, cost of call, call signers and results of transaction.

The block the data is added to can be traced and used for time verification.

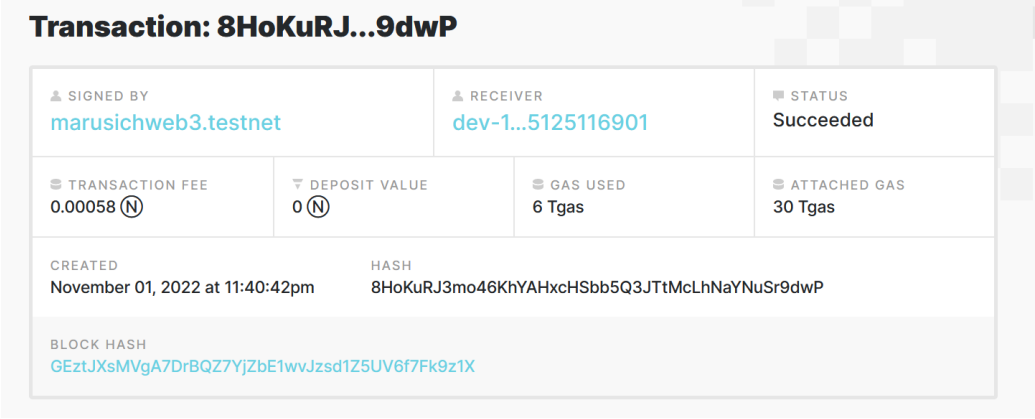

Figure 50: Block data

Figure 51:Near transaction details of submission

Chapter 6: Conclusion and Recommendation

The integration of IoT, AI and Blockchain offers a new range of innovative solutions for state-of-the-art open challenges. As explored in this research, one of the challenges is to trace the departure and arrival times of vehicles in a logistic value chain especially when missing deadlines can yield huge financial and even health consequences.

This study presents a research-driven design and prototyping process to build an end-to-end system that uses a Raspberry Pi embedding a web camera, a trained AI model to detect location of the vehicle plate on an image and recognize the plate numbers from image, a wallet to sign the logging of events on a remote blockchain ledger based on the NEAR protocol. NEAR blockchain has been selected thanks not only to its scalability and low transaction features but also its developer experience making the development of smart contracts easier for traditional web 2 developers.

The experimentation carried out in the campus parking shows that the proposed system achieves a range of 92%-98% of accuracy for object detection. The NEAR blockchain provides an immutable, open, and secure store for vehicle plate data. This prototype can serve to provide time-based accountability solutions trusted by all different logistic stakeholders.

6.1. Recommendations

This system can be used in public transport systems to determine fines and rewards for buses and trains arriving on time or being delayed.

Medical delivery systems and medical test facilities using samples can use this system as part of their accountability line for users, to gain trust from parties involved and increase efficiency of service delivery.

Fault management systems in the energy and telecommunications sector can benefit from a system of recording time in an immutable manner to verify response times to faults and allow end users to transfer value when faults are resolved in time and for companies to pay penalties for not addressing faults in a reasonable time.

6.2. Future Works

Future works will focus on improving the accuracy of the plate recognition AI process and to implement automated smart contract business logics that enable to automatically charge a faulty stakeholder from its collateral tokens to be deposited when accepting a logistic mission.

References

[1] https://law.unimelb.edu.au/__data/assets/pdf_file/0005/3385454/Schwab-The_Fourth_Industrial_Revolution_Klaus_S.pdf

[2] https://www.oracle.com/in/internet-of-things/what-is-iot/

[3] https://bitcoin.org/bitcoin.pdf

[4] Zheng, Zibin & Xie, Shaoan & Dai, Hong-Ning & Chen, Xiangping & Wang, Huaimin. (2017). An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends. 10.1109/BigDataCongress.2017.85.

[5] Savelberg, Fons & Warffemius, Pim M.J. & Kroes, Eric. (2017). ESTIMATION OF THE SOCIAL COSTS OF TRAIN DELAYS IN THE NETHERLANDS.

[6] Veenstra Jay, 2015-11, The Cost of Tardiness, Toyo University Repository for academic purposes, , 86, http://id.nii.ac.jp/1060/00007955

[7] A. Shtayat, M. Abu Alfoul, S. Moridpour, N. Al-Hurr, K. Magableh, and I. Harahsheh, "Waiting Time of Public Transport Passengers in Jordan: Magnitude and Cost," The Open Transportation Journal, vol. 13, no. 1, Dec. 2019, doi: 10.2174/1874447801913010227

[8] Hsu, Chaug-Ing & Hung, Sheng-Feng & Li, Hui-Chieh. (2007). Vehicle routing problem with time-windows for perishable food delivery. Journal of Food Engineering. 80. 465-475. 10.1016/j.jfoodeng.2006.05.029.

[9] Sylverken AA, Owusu M, Agbavor B, Kwarteng A, Ayisi-Boateng NK, Ofori P, et al. (2022) Using drones to transport suspected COVID-19 samples; experiences from the second largest testing centre in Ghana, West Africa. PLoS ONE 17(11): e0277057. https://doi.org/10.1371/journal.pone.0277057

[10] Haji, M.; Kerbache, L.;Sheriff, K.M.M.; Al-Ansari, T. Critical Success Factors and Traceability Technologies for Establishing a Safe Pharmaceutical Supply Chain. Methods Protoc. 2021, 4, 85. https://doi.org/10.3390/mps4040085

[11] Gnap, Jozef & Varjan, Pavol & Semanova, Stefania. (2017). Logistics of Entry and Parking of Vehicles at Large Production Companies. MATEC Web of Conferences. 134. 00016. 10.1051/matecconf/201713400016.

[12] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5821242/

[13] https://circuitdigest.com/microcontroller-projects/license-plate-recognition-using-raspberry-pi-and-opencv

[14] https://circuitdigest.com/tutorial/image-segmentation-using-opencv

[15]https://www.crayondata.com/machine-learning-license-plate-recognition-an-ideal partnership/

[16] https://medium.com/geekculture/license-plate-detection-using-artificial-intelligence-ai-9fe6de3b941a

[17] https://www.educative.io/answers/what-are-centralized-decentralized-and-distributed-systems

[18] https://www.tensorflow.org/lite/examples/object_detection/overview

[19] https://supplain.io/news/web3-solve-web2-problems

[20] https://www.ibm.com/topics/benefits-of-blockchain

[21] Z. Ren, K. Cong, T. Aerts, B. de Jonge, A. Morais and Z. Erkin, "A Scale-Out Blockchain for Value Transfer with Spontaneous Sharding," 2018 Crypto Valley Conference on Blockchain Technology (CVCBT), 2018, pp. 1-10, doi: 10.1109/CVCBT.2018.00006.

[22] Panarello A, Tapas N, Merlino G, Longo F, Puliafito A. Blockchain and IoT Integration: A Systematic Survey. Sensors. 2018; 18(8):2575. https://doi.org/10.3390/s18082575

[23] A. Kaushik, A. Choudhary, C. Ektare, D. Thomas and S. Akram, "Blockchain — Literature survey," 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), 2017, pp. 2145-2148, doi: 10.1109/RTEICT.2017.8256979.

[24] Md Sadek Ferdous, Mohammad Jabed Morshed Chowdhury, Mohammad A. Hoque,A survey of consensus algorithms in public blockchain systems for crypto-currencies,Journal of Network and Computer Applications,Volume 182,2021,103035,ISSN 1084 8045,https://doi.org/10.1016/j.jnca.2021.103035.

[25] 9. Nartey, Clement & Tchao, E. T. & Gadze, Dzisi & Keelson, Eliel & Klogo, Griffith & Kommey, Benjamin & Diawuo, Kwasi. (2021). On Blockchain and IoT Integration Platforms: Current Implementation Challenges and Future Perspectives. Wireless Communications and Mobile Computing. 2021. 1-25. 10.1155/2021/6672482.

[26] 7. J. Xu et al., "Healthchain: A Blockchain-Based Privacy Preserving Scheme for Large-Scale Health Data," in IEEE Internet of Things Journal, vol. 6, no. 5, pp. 8770-8781, Oct. 2019, doi: 10.1109/JIOT.2019.2923525.
[27] Ye H, Park S. Reliable Vehicle Data Storage Using Blockchain and IPFS. Electronics. 2021; 10(10):1130. https://doi.org/10.3390/electronics10101130
[28] V. Nishad, V. Kumar and P. Mittal, "License Plate Detection by Edge Detection Algorithm and Smart Exposition with IoT," 2021 International Conference on Simulation, Automation & Smart Manufacturing (SASM), 2021, pp. 1-6, doi: 10.1109/SASM51857.2021.9841216.
[29] R. Laroca et al., "A Robust Real-Time Automatic License Plate Recognition Based on the YOLO Detector," 2018 International Joint Conference on Neural Networks (IJCNN), 2018, pp. 1-10, doi: 10.1109/IJCNN.2018.8489629.

[30] Dutta, Pushan and Chakraborty, Arnab and Bhadury, Soumyadeep and Sinha, Aniruddha and Mitra, Sayan and Das, Saswata, Smart Usage of Open Source License Plate Detection and Using IoT Tools for Private Garage and Parking Solutions (February 7, 2020). Proceedings of Industry Interactive Innovations in Science, Engineering & Technology (I3SET2K19), Available at SSRN: https://ssrn.com/abstract=3533565 or http://dx.doi.org/10.2139/ssrn.3533565
[31] K. Christidis and M. Devetsikiotis, "Blockchains and Smart Contracts for the Internet of Things," in IEEE Access, vol. 4, pp. 2292-2303, 2016, doi: 10.1109/ACCESS.2016.2566339.
[32] Hesam Hamledari, Martin Fischer,Construction payment automation using blockchain - enabled smart contracts and robotic reality capture technologies, Automation in Construction,Volume 132,2021,103926,ISSN 0926 5805,https://doi.org/10.1016/j.autcon.2021.103926
[33] X. Liang, J. Zhao, S. Shetty and D. Li, "Towards data assurance and resilience in IoT using blockchain," MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM), 2017, pp. 261-266, doi: 10.1109/MILCOM.2017.8170858.
[34]https://upload.wikimedia.org/wikipedia/commons/thumb/5/5f/Three_software_development_patterns_mashed_together.svg/2000px-Three_software_development_patterns_mashed_together.svg.png
[35] https://www.lumitex.com/blog/prototyping-methodology

[36] https://medium.com/coinmonks/the-blockchain-as-a-protocol-to-transfer-value-1d976f28cc0b

[37] https://docs.near.org/develop/contracts/whatisacontract

[38] https://webassembly.org/

[39] https://github.com/near/near-sdk-as

[40] https://near.org/papers/nightshade/

[41]https://www.reddit.com/r/nearprotocol/comments/lu9cyk/advantages_and_disadvantages_of_near_protocol/

[42] https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi/

[43] https://iotdesignpro.com/projects/real-time-license-plate-recognition-using-raspberry-pi-and-python

[44] https://github.com/tesseract-ocr/tesseract

[45] https://docs.near.org/tools/near-cli

[46] https://arxiv.org/abs/2004.10934

[47] https://docs.opencv.org/4.x/da/d22/tutorial_py_canny.html

[48] https://circuitdigest.com/microcontroller-projects/license-plate-recognition-using-raspberry-pi-and-opencv

[49] https://medium.com/analytics-vidhya/yolo-explained-5b6f4564f31

[50] https://medium.com/aiguys/yolo-v4-explained-in-full-detail-5200b77aa825

[51]https://nanonets.com/blog/ocr-with tesseract/#:~:text=Pytesseract%20or%20Python%2Dtesseract%20is,image%20to%20text%20use%20cases.

[52] https://pypi.org/project/pytesseract/