

# Internship Report

Vehicle tracking, AI chat bot

Narayan Bandodker

2004

---

## Report of Internship done at phAldelta

### Vehicle tracking; AI chat bot

Completed by  
**Narayan Bandodker**  
2004

for the partial fulfillment of  
**MCA Degree for Semester VI**  
Discipline of Computer Science and Technology,  
Goa Business School,  
Goa University

At  
**phAldelta**  
**Panjim, Goa**

Under the guidance of  
**Shalabh Kumar**  
(Founder/CEO, phAldelta)

and

**Sharanya Ratish**  
(Manager - HR/Business Development, phAldelta)

and

**Ashpak Shaikh**  
(Software Development, phAldelta)





phAIdelta

Date: 31/05/2023

TO WHOMSOEVER IT MAY CONCERN

This is to certify that Mr. Narayan Bandodkar, a student of Master of Computer Applications (MCA) of Goa University, Goa, is currently undergoing/has completed his/her final semester project (Semester VI) at **phAIdelta**, from 2<sup>nd</sup> Jan 2023 to 30<sup>th</sup> June 2023

During his tenure, he has met his superiors' expectations and is found to be regular and sincere.

This certificate is being issued on his request to be submitted with the project report at Goa University.

*The final internship completion certificate will be provided on completing his internship.*

Yours Sincerely

For phAIdelta

Sharanya Ratish

Manager – HR and Business Development

+91 98196 52514  
+91 80107 68815

bd@phaidelta.com  
www.phaidelta.com

303, K Block, Adwalpalkar Shelters,  
Kerant Caranzalem, Taleigo, Goa 403 002

---

GOA UNIVERSITY



GOA BUSINESS SCHOOL

CERTIFICATE OF EVALUATION

This is to certify that Mr. Narayan Shrinivas Bandodker has successfully completed his internship at **phAldelta, Panjim, Goa**, in partial fulfillment of the award of the degree in Master of Computer Application.

Examiner 1

Examiner 2

Place: Goa University

Date: 14/06/2023

Dean, Goa Business School



---

## Acknowledgement

The internship opportunity I had with phAIdelta was a great chance of learning and professional development. I am very grateful for having an opportunity where I met many amazing people and professionals who led me through this internship. I would like to express my sincere gratitude and appreciation towards all of them, who supported me even when I was at my lows.

I would firstly like to thank **Mr. Shalabh Kumar** (Founder, CEO of phAIdelta) for considering me for this role and helping me throughout the internship and for providing me the necessary guidance and support, and helping me a lot with the initial tasks and with projects' documentation. I would also like to thank **Mr. Ratish Radhakrishnan** (Founder) who provided valuable help and feedback during development.

I would like to extend my gratitude to **Mrs. Sharanya Ratish** (Manager - HR/BD) and **Ms. Sailee Madkaikar** (Finance and Accounts) for incredible amount of help in onboarding, with helpful advice and time keeping, and also in helping with managing various matters in the internship process. I thank **Mr. Ashpak Shaikh** (Software Development team) for being a wonderful coworker and a really great team member, helping me with app development cycle planning, teaching me various aspects of backend development, and also other things such as electronics and IoT related tech.

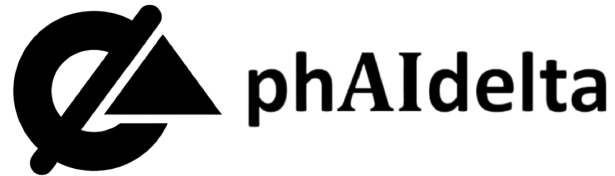
I thank **Prof. Jyoti D. Pawar** (Professor of MCA and Dean of Goa Business School - Goa University), **Prof. Ramdas N. Karmali** (Professor of MCA, GBS), **Prof. Ramrao S. Wagh** (Program Director of MCA) and all the faculty of MCA, Goa University for their constant encouragement and support during the project work.

Today at the apex point of my internship, I gratefully remember my parents, professors and friends for their non-stop support and helping me climb up in this career path, without whom this internship would not have been this awesome and a fun experience for me.

Finally, I would also like to express my gratitude towards the entire phAIdelta company staff for being such wonderful and amazing people. They have enhanced my potential and have given me enough confidence to face the challenges of real life that await me.

I want to also give a shout-out to my friends on the Discord community for always being there to talk to.

---



# Internship Report

Vehicle tracking, AI chat bot

Narayan Bandodker

2004



---

# Report of Internship done at phAldelta

## Vehicle tracking; AI chat bot

Completed by  
**Narayan Bandodker**  
**2004**

for the partial fulfillment of  
**MCA Degree for Semester VI**  
Discipline of Computer Science and Technology,  
Goa Business School,  
Goa University

At  
**phAldelta**  
**Panjim, Goa**

Under the guidance of  
**Shalabh Kumar**  
(Founder/CEO, phAldelta)

and

**Sharanya Ratish**  
(Manager - HR/Business Development, phAldelta)

and

**Ashpak Shaikh**  
(Software Development, phAldelta)

TO WHOMSOEVER IT MAY CONCERN

This is to certify that Mr. Narayan Bandodkar, a student of Master of Computer Applications (MCA) of Goa University, Goa, is currently undergoing/has completed his/her final semester project (Semester VI) at **phAIdelta**, from 2<sup>nd</sup> Jan 2023 to 30<sup>th</sup> June 2023

During his tenure, he has met his superiors' expectations and is found to be regular and sincere.

This certificate is being issued on his request to be submitted with the project report at Goa University.

*The final internship completion certificate will be provided on completing his internship.*

Yours Sincerely

For phAIdelta



**Sharanya Ratish**

**Manager – HR and Business Development**



+91 98196 52514  
+91 80107 68815



bd@phaidelta.com  
www.phaidelta.com



303, K Block, Adwalpalkar Shelters,  
Kerant Caranzalem, Taleigo, Goa 403 002



---

## GOA UNIVERSITY



## GOA BUSINESS SCHOOL

### CERTIFICATE OF EVALUATION

This is to certify that Mr. Narayan Shrinivas Bandodker has successfully completed his internship at **phAldelta, Panjim, Goa**, in partial fulfillment of the award of the degree in Master of Computer Application.

---

Examiner 1

---

Examiner 2

**Place:** Goa University

**Date:** 14/06/2023

---

Dean, Goa Business School

---

## Acknowledgement

The internship opportunity I had with phAIdelta was a great chance of learning and professional development. I am very grateful for having an opportunity where I met many amazing people and professionals who led me through this internship. I would like to express my sincere gratitude and appreciation towards all of them, who supported me even when I was at my lows.

I would firstly like to thank **Mr. Shalabh Kumar** (Founder, CEO of phAIdelta) for considering me for this role and helping me throughout the internship and for providing me the necessary guidance and support, and helping me a lot with the initial tasks and with projects' documentation. I would also like to thank **Mr. Ratish Radhakrishnan** (Founder) who provided valuable help and feedback during development.

I would like to extend my gratitude to **Mrs. Sharanya Ratish** (Manager - HR/BD) and **Ms. Sailee Madkaikar** (Finance and Accounts) for incredible amount of help in onboarding, with helpful advice and time keeping, and also in helping with managing various matters in the internship process. I thank **Mr. Ashpak Shaikh** (Software Development team) for being a wonderful coworker and a really great team member, helping me with app development cycle planning, teaching me various aspects of backend development, and also other things such as electronics and IoT related tech.

I thank **Prof. Jyoti D. Pawar** (Professor of MCA and Dean of Goa Business School - Goa University), **Prof. Ramdas N. Karmali** (Professor of MCA, GBS), **Prof. Ramrao S. Wagh** (Program Director of MCA) and all the faculty of MCA, Goa University for their constant encouragement and support during the project work.

Today at the apex point of my internship, I gratefully remember my parents, professors and friends for their non-stop support and helping me climb up in this career path, without whom this internship would not have been this awesome and a fun experience for me.

Finally, I would also like to express my gratitude towards the entire phAIdelta company staff for being such wonderful and amazing people. They have enhanced my potential and have given me enough confidence to face the challenges of real life that await me.

I want to also give a shout-out to my friends on the Discord community for always being there to talk to.



---

## Table of Contents

<b>Report of Internship done at phAIdelta.....</b>	<b>2</b>
<b>Acknowledgement.....</b>	<b>5</b>
<b>I. Introduction.....</b>	<b>8</b>
<b>II. Company Profile.....</b>	<b>9</b>
<b>III. Project 1 - Optical tracking Vehicle Parking solution.....</b>	<b>10</b>
Overview.....	10
Optical tracking.....	10
UI and backend interface.....	10
Vehicle detection.....	11
Stable tracking.....	11
ANPR.....	11
<b>IV. Project 2 - Interactive Chatbot for statistics.....</b>	<b>12</b>
Overview.....	12
NLU.....	12
Actions.....	12
<b>V. My contributions.....</b>	<b>13</b>
Task 1: Design vehicle detection model and tracking.....	13
Task 2: Entry/Exit detection.....	15
Task 3: Analytics bot model training.....	16
Task 4: AI chatbot UI.....	17
<b>VI. Some AI/Vision concepts I learned and experimented.....</b>	<b>18</b>
a. Homography and perspective transform.....	18
b. Transformer model.....	20
<b>VII. Tools and Technologies Used.....</b>	<b>21</b>
<b>VIII. Conclusions / Summary.....</b>	<b>24</b>
<b>IX. Screenshots / Images.....</b>	<b>25</b>
Vehicle Parking.....	25
Analytics bot.....	25
<b>X. Project Timeline / Weekly Log.....</b>	<b>27</b>
January 2023.....	27
February 2023.....	27
March 2023.....	28
April 2023.....	29
May 2023.....	30

---

---

<b>XI. My reflections.....</b>	<b>31</b>
<b>References.....</b>	<b>32</b>
Other references.....	32



---

## I. Introduction

The purpose of this report is to summarize, reflect on and analyze my full-time internship at **phAIdelta**, Panjim.

I joined as an intern at **phAIdelta** on 2nd January 2023, and have been here since then. I have been working from home and conducting meetings and calls via VoIP. This report contains a collection of projects that I have worked on during my internship period at **phAIdelta** and also the new things I've learned, new experiences gained while working.

In the following sections, I will talk about the company, the work there, the culture, etc. I shall then elaborate on the projects I've worked on during the internship period, and a brief description of the projects with its core components, along with the tasks allotted to me and completed by me in those projects.

The report highlights my learning experience and my contributions to the organization as an intern. This will describe the knowledge that I gained by successfully completing the tasks that were assigned to me.

I will also be talking about the tools and technologies that were used, followed by the internship timeline. I shall conclude by sharing my general experience and how it has helped me to grow in the professional front.

---

## II. Company Profile

**Name of Company:** phAIdelta LLP

**Headquarters:** Panjim, Goa

**Website:** <https://www.phaidelta.com/>

**Email ID:** [admin@phaidelta.com](mailto:admin@phaidelta.com)

At **phAIdelta**, we partner with each of our clients to develop and improve their business by providing strong process and technology consulting services.

Our Core Team, with their collective experience of numerous decades in Consulting and deployments in the services industry, helps organizations by diagnosing, measuring, and analyzing processes and coming up with value-creation projects which we monitor and help implement. All of this is backed by leveraging our expertise with domain-specific analytics.

Our focus on Lean methodology with technology implementations means we help Organizations save costs and time. We believe in creating a lean product for enabling faster availability. This helps organizations invest and focus more on market feedback and improving the product.

But most of all we understand that choosing a partner for every organization has one underlying factor and that is TRUST. **At phAIdelta we deliver to our commitments no matter what!**

### Services provided:

- ★ Process Consulting
- ★ Data Engineering and Analytics
- ★ Robotic Process Automation
- ★ Artificial Intelligence
- ★ Custom Software Development
- ★ Chatbot technology

### Our Mission

*Simplifying Data-Based Decision Making using Technology!*

---

### III. Project 1 - Optical tracking Vehicle Parking solution

#### Overview

The goal of this project is to provide a low-cost, high-performance solution to parking lots and building owners that uses existing surveillance camera technology in order to detect presence of vehicles within a perimeter, track movement and notify the client of entry and exit of vehicles. A dashboard that is accessible remotely in any web browser, provides a live feed of the camera, and a history of all detected vehicles. It supports gate entry/exit which allows vehicles to be marked when they enter or leave the premises.

#### Optical tracking

The use of CCTV cameras in or around parking areas is not uncommon. Owners of the building, say an office, would want to protect the vehicle owners by monitoring vehicles that enter at a certain time, and if an unexpected vehicle enters at outside working hours, say midnight. This is usually the job of a security guard that constantly monitors footage for suspicious activity. This task is possible to automate via computer vision technology, and the advancements in optical tracking can help in speed up and also improve detection rate.

The idea is that a camera feed can be directly given to a detection algorithm that is trained to monitor vehicles, and a movement in them can trigger the optical tracking system which can accurately follow a vehicle's movement.

#### UI and backend interface

The dashboard UI is made in plain HTML with Bootstrap JS/CSS library and bootswatch for theming. The page is served by the backend using Jinja template engine. The UI has the job of displaying information of vehicle entry/exit and also to show live video feed. It is able to display live video using WebRTC streaming, which is also used to update live events of entry/exit. WebRTC can create a video element that receives live video frames from the backend, and WebRTC *DataChannel* is used for the events. This makes it efficient to serve events quickly without polling.

---

The backend is based on the **aihttp** Python package that hosts the complete application of vehicle tracking and video processing. It handles camera input, video processing to reduce noise, dispatch video frames to WebRTC clients.

What about the ML models? That is done on a model inference server that hosts the tracking and detection models. The backend sends requests to this server and the response is the tracking points. The models are run using 🤗**HuggingFace** transformer framework, and use GPU acceleration to speed up the inference. An estimated 30fps can be seen in detection and ~12fps for tracking. This is sufficient for stable tracking of slow-moving vehicles.

## Vehicle detection

A custom dataset was used to train an AI detection model which can detect vehicles in different orientations. A YOLOv5-based model was fine-tuned with this task, and deployed to the model server. The YOLOv5 model has video frames as input, and provides X,Y,W,H coordinates as output. We can use this further to track it, and even log vehicle detections.

## Stable tracking

The tracker works by identifying previous vehicle detections in subsequent video frames. The tracker's job is to follow the same vehicle's motion as closely as possible without losing track. OpenCV has in-built multi-object tracking support via `cv::TrackerCSRT` class. We can supply video frames along with previous detection boxes, and the tracker will try to follow the box even if detection is lost.

## ANPR

**Automatic Number-Plate Recognition** is a process which requires the app to detect a (Indian) number plate on the detected vehicle and convert the printed numbers to text characters using OCR. This becomes more complicated as the number plate standard allows two-line plates and also the pattern or font can be different. There can also be slight changes in the digits themselves, such as being 0-padded or not, two letters or just one letter for the region code, having different spacing between letters, etc. This makes OCR much more difficult, and may require a tailor-made model that can perform the OCR more accurately.



---

## IV. Project 2 - Interactive Chatbot for statistics

### Overview

The chatbot, nicknamed **Analytics bot**, is an **AI** chatbot. It makes use of the well-known **RASA** chat framework that helped bootstrap the project with some models that help the bot in making decisions. The bot is helpful to users making frequent statistical calculations on data that requires only text input to perform the tasks. For example, recent temperature data in a climate-controlled room may reside within a certain range. The bot can be asked if at any point the temperature has crossed a threshold limit. Any data that requires quick aggregation such as finding average, any outliers can be requested.

Such tasks where any raw data can be processed into an easy-to-read output with only a few text commands is the goal of this bot. This data can be further displayed as a plotted chart.

### NLU

This is the basis for the Rasa chatbot, and is the hardest task to get it done correctly. NLU or Natural Language Understanding helps the model know the user's intent from a given English sentence. For example: "I would like to book a ticket to watch Spider-Man" will be treated as an intent "booking\_ticket", with parameters, aka **entity** being "Spider-Man" for **movie\_title**. This is called *entity extraction* and helps in the bot being more user-friendly. An NLU model can be trained with such training examples with minor variations that help the model understand the intent correctly.

### Actions

Actions to perform when an intent is selected. This is the business logic of the bot and handles CRUD operations on behalf of the user. For the above example, booking a ticket can insert a row in a DB table to ensure the movie ticket is generated. The action also has the task of generating response text that will be shown to the user. For example, "Ticket booked successfully" can be shown.

---

## V. My contributions

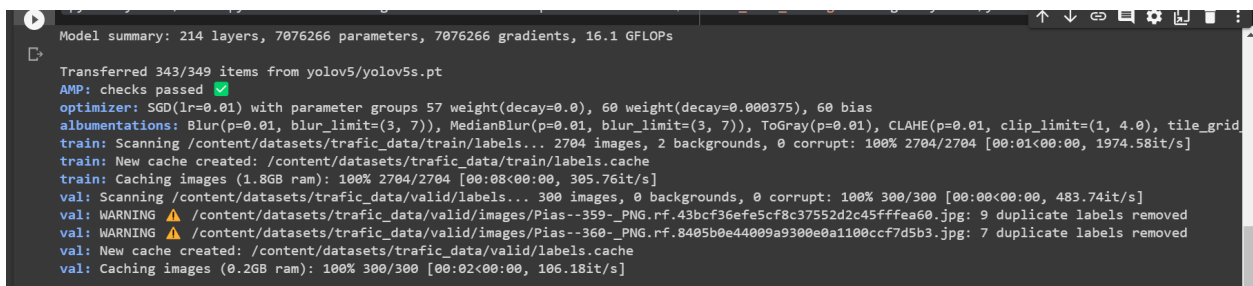
I have been assigned as an Associate Software Engineer, and I have been given the role of designing and developing core elements of the projects to push the concept into a MVP phase, giving ideas and improvements along the way. Along with development tasks, I had been assigned other tasks for helping one of my coworkers who has less experience in coding with advice and hosting explainer sessions to teach important concepts in backend development.

As I was on-boarded with the projects, I have started with understanding the goal, preparing questions and preparing a design document that outlines all required components, features needed and the minimum set of requirements for the project to reach MVP stage. This helped me stay focused and know the progress of the project at any given time.

### Task 1: Design vehicle detection model and tracking

I was asked to work on the detection and tracking module of the Vehicle entry/exit project. This is the core part for discovering when vehicles are present, so that the rest of the system can track and store vehicle detections. I first used OpenCV and prepared a PoC app that takes raw images of highway traffic and I used YOLOv5 to detect. YOLOv5 is a very fast detection model that can also be trained on a different dataset. We need to prepare the dataset first by labeling the images. I used a public dataset for this proof of concept which was based on Indian Vehicles.<sup>[1]</sup> The YOLOv5 script was installed and training process was done on a Google Colab notebook using GPU.

```
!python yolov5/train.py --cache ram --img 640 --batch 48 --epochs 60 --data  
"$vehicle_data_config" --weights yolov5/yolov5s.pt --device 0
```



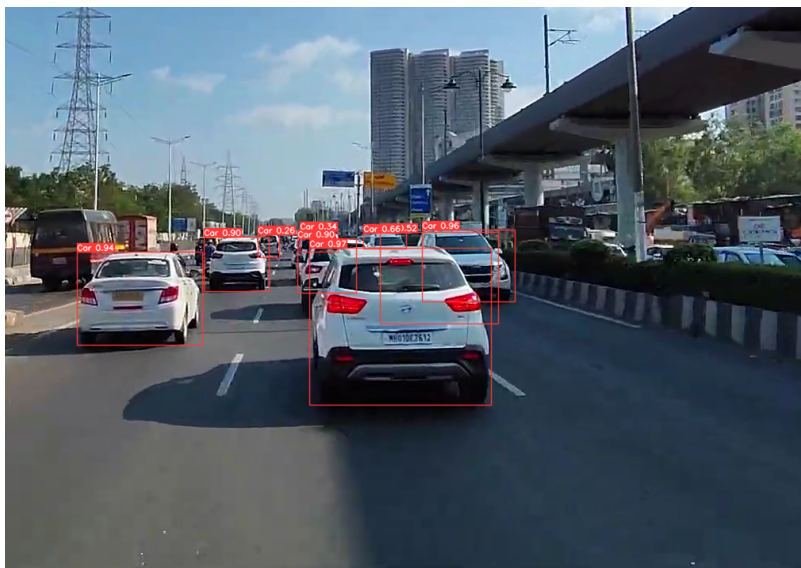
```
Model summary: 214 layers, 7076266 parameters, 7076266 gradients, 16.1 GFLOPs  
Transferred 343/349 items from yolov5/yolov5s.pt  
AMP: checks passed  
optimizer: SGD(lr=0.01) with parameter groups 57 weight(decay=0.0), 60 weight(decay=0.000375), 60 bias  
augmentations: Blur(p=0.01, blur_limit=(3, 7)), MedianBlur(p=0.01, blur_limit=(3, 7)), ToGray(p=0.01), CLAHE(p=0.01, clip_limit=(1, 4.0), tile_grid  
train: Scanning /content/datasets/traffic_data/train/labels... 2704 images, 2 backgrounds, 0 corrupt: 100% 2704/2704 [00:01<00:00, 1974.58it/s]  
train: New cache created: /content/datasets/traffic_data/train/labels.cache  
train: Caching images (1.8GB ram): 100% 2704/2704 [00:08<00:00, 305.76it/s]  
val: Scanning /content/datasets/traffic_data/valid/labels... 300 images, 0 backgrounds, 0 corrupt: 100% 300/300 [00:00<00:00, 483.74it/s]  
val: WARNING /content/datasets/traffic_data/valid/images/Pias--359-.PNG.rf.43bcf36efe5cf8c37552d2c45fffea60.jpg: 9 duplicate labels removed  
val: WARNING /content/datasets/traffic_data/valid/images/Pias--360-.PNG.rf.8405b0e44009a9300e0a1100ccf7d5b3.jpg: 7 duplicate labels removed  
val: New cache created: /content/datasets/traffic_data/valid/labels.cache  
val: Caching images (0.2GB ram): 100% 300/300 [00:02<00:00, 106.18it/s]
```

I found sample footage to test the newly trained model and then ran the model on the video (here I've used a screenshot for easier comparison).

```
!python yolov5/detect.py --line-thickness 1 --weights
vehicle-front-back-best.pt --source "./Screenshot.png"
```



Sample video frame. Courtesy: YouTube @VDLibrary-xc7yn <sup>[2]</sup>



YOLOv5 output, cropped for clarity

---

From the above, we can see vehicles have been identified correctly with bounding boxes. This model can be used for classifying various objects as well, such as pedestrians and road signs. But for our purpose, only vehicle detection is sufficient.

## Task 2: Entry/Exit detection

The way this works is we can detect the trajectory of the vehicle when moving within the premises. We check to see the direction the vehicle moves (i.e. degrees angle). This angle is determined by the slope of the trajectory by using trigonometric functions to get the angle.



Image showing the gate line for detection (gray horizontal line) with green arrow indicating vehicle movement direction, and red arrow indicating direction of gate normal.

To do this, we first log the coordinates in a list, and get the most recent 2 points. We then calculate its slope-angle using arctan (i.e.  $\tan^{-1}$ ), which can be used later on.

```
bz_p1, bz_p2 = bz_coords[0], bz_coords[-1]
dx = bz_p1[0]-bz_p2[0]
dy = bz_p1[1]-bz_p2[1]
angle = np.arctan2(dy, dx)
speed = np.sqrt(np.square(dy)+np.square(dx))
is_moving = speed >= self._min_movement_detection_speed
```



---

Further on, we can use the Euclidean distance formula to calculate the velocity of the vehicle, and determine if it is in motion.

Now it is easy to detect if a vehicle is entering or exiting, based on this angle when it crosses the gate. We can check for intersection with the gate, and look at the angle at that time, and determine if entering (angle is facing normal of gate) or exiting (angle is 180 degrees from normal of gate).

## Task 3: Analytics bot model training

I've been tasked to add intents along with their NLU for training data to create the bot's model. This data is based on conversational flow (User -> Bot -> User, etc.) but can include inputs that can be extracted. For example, "Book a restaurant reservation at 9:00AM". This will extract the time "9:00AM" to be used for processing the reservation action.

Each user request is called an "intent". This intent refers to sentences that the user can ask. The above booking restaurant example can be called "intent\_book\_reservation". This intent will be triggered as soon as a sentence similar to the example is received. The more sentences and variations, the better it will distinguish it.

Here is how an NLU is written in Rasa:

```
- intent: query_capabilities
  examples: |
    - What can you do?
    - What are you able to do?
    - What tasks can you perform?
    - What are your capabilities?
    - Tell me what can you do.
```

This seems tedious to write, especially when there are inputs to take care of, which can occur with different values (eg. cuisines available from a list). With few examples, the bot may not be able to classify the intent correctly, or not extract entities. To solve this, we need to augment sentences with different synonyms and entity values. I.e. use data augmentation tool to create NLU from a template.

This is what can be done using **Chatette**, a data dataset generator for Rasa NLU. It can take sentences with marked entities, and synonym words, it will generate variations of these sentences and output as a file the NLU training data.

The above can be written as follows using Chatette's DSL syntax:

---

```
%[&query_capabilities](10)
  What [tasks?] can you [do|perform]?
  What are your ~[capabilities]?

~[capabilities]
  capabilities
  functions
  commands
```

This will more-or-less generate a similar NLU, but with much more variations and the “capabilities” synonym will substitute values. This will generate 10 random sets of sentences.

Entities also need to be labeled. In Rasa, we use square-bracket and parenthesis to do it:

```
I want to order [sushi](dish)
Book at [10:00AM](time)
```

Rasa will use its entity extractor to detect and parse entities and give them to the processing stage.

## Task 4: AI chatbot UI

For demonstration of the app, I’ve been asked to create a page with the chat widget on a floating button, and make use of REST API to connect with the chatbot backend. Rasa has a defined schema for this API<sup>[3]</sup>, and the UI can send text, then wait for the response data and display it for the user.

I used **create-react-app**, which is a bootstrapping tool to quickly make a React application. I used MUI<sup>[4]</sup> to create UI elements such as navbar, floating button, text, etc. and used Axios<sup>[5]</sup> for making requests to the chatbot server.

The main challenge was the chat widget. Before I started that, I used the Browser’s console to chat by calling the API function, so that I can confirm data is coming correctly. I created a textbox with a submit button to send the REST request using Axios. The result is stored in a div component.

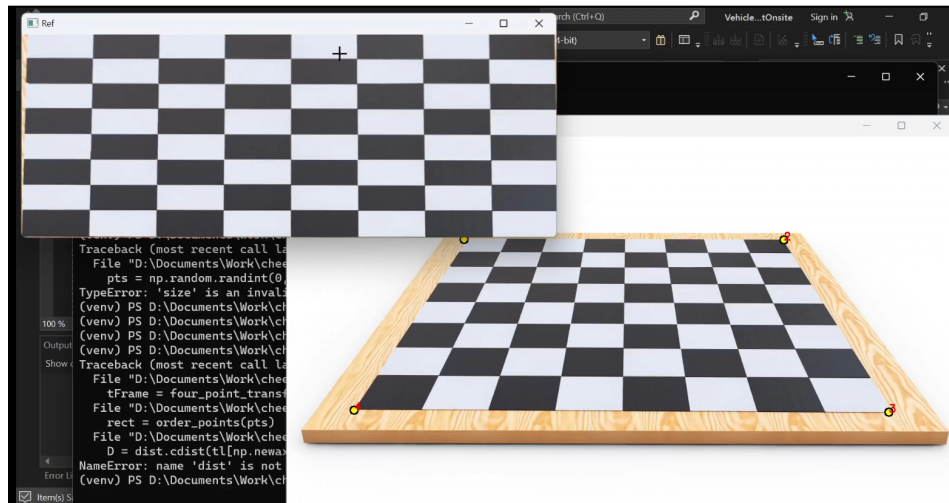
I then used a component called **react-chat-widget** that contains a textbox, send button, chat history and renders messages as Markdown. This, combined with MUI’s **Floating Action Button** completed the UI layout of the application. I completed the integration of Axios with the input text, and created the Dockerfile for deployment.

---

## VI. Some AI/Vision concepts I learned and experimented

### a. Homography and perspective transform

One important algorithm that I came across while researching for the vehicle tracking project was camera perspective correction. Not always the camera will be perfectly aligned and face head-on with the area. We need to correct for this angle by artificially rotating the image based on markers placed at landmarks. These markers can be skewed based on their original position so that the perspective is corrected. Finding the transform needed to move between these two perspectives is called Homography. For example, in the below demo, a chessboard pattern at a front-facing leaning angle can be flattened to a (more or less, depending on quality of markers placed) top-down perspective. This can be useful for determining the actual piece locations in the board.



Demonstrating (rather crudely) correcting perspective of chess board

There's two steps to calculate the Homography of an object: We need a reference set of markers in the perspective we desire, i.e. Top-down perspective, and the second requirement is the same points in the new perspective image. Using a checkerboard pattern such as the one above is ideal as it has 180 degrees rotation symmetry, called a “**calibration pattern**”.

---

OpenCV can detect the corners of each box and use that as reference, as so:

```
pat_success_perp, cb_corners_persp = cv2.findChessboardCorners(
    cb_preprocess(img_board_perspective),
    cb_size,
    flags=cv2.CALIB_CB_ADAPTIVE_THRESH + cv2.CALIB_CB_EXHAUSTIVE
)
pat_success_td, cb_corners_td = cv2.findChessboardCorners(
    cb_preprocess(img_board_topdown),
    cb_size_2,
    flags=cv2.CALIB_CB_ADAPTIVE_THRESH + cv2.CALIB_CB_EXHAUSTIVE
)
```

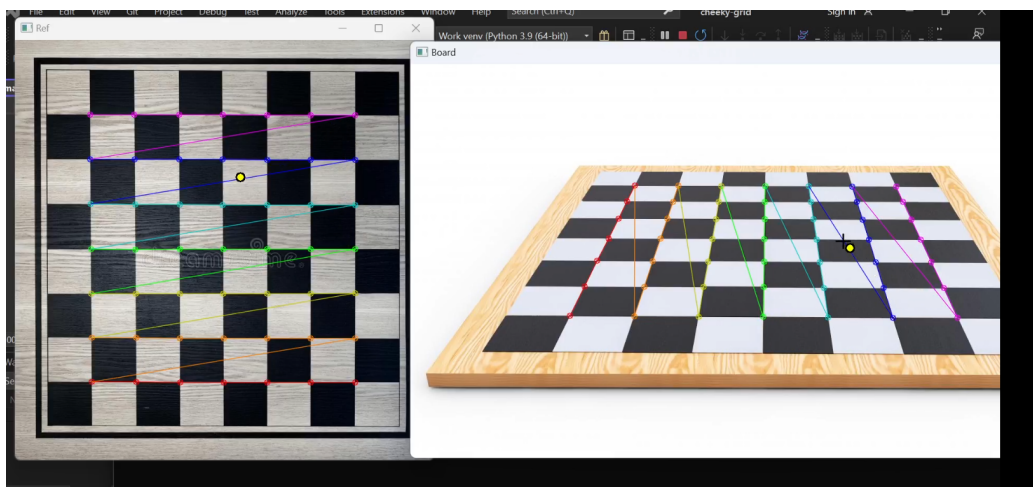
We then find the homography matrix using the above two corner points:

```
chess_homography, has_done_homography =
cv2.findHomography(cb_corners_persp, cb_corners_td)
```

Now we can calculate a point on the image as needed using perspective transform function:

```
pts = cv2.perspectiveTransform(np.array([pt1], dtype='float32'),
chess_homography)
```

When used correctly, we can transform any point on the perspective image to top-down X-Y coordinates:



Demo of Yellow dot on right image being transformed to X-Y point on left image

When using a video, we can perform the `cv2.findHomography` steps for every frame to calculate the new homography matrix.

---

## b. Transformer model

Transformers are the next-step in machine learning, and we can easily adopt it for any Computer Vision (Detection/recognition) or NLP tasks (chatbots, summary generation). Outperforming older RNN models, it can be used by-default in place of LSTM. We experimented with using a Transformer model for OCR tasks using the TrOCR model by Microsoft. It is a general-purpose OCR used mainly for bill and receipt reading, and can be tasked to recognize any form of text, such as Number plates.

Transformers by 🤗 **Hugging Face** makes it simple to implement and train models with minimal lines of code. This can be done by loading a model structure with base weights, and then fine-tuning with our own dataset. Here, we can create a model and download the weights from the Hugging Face hub:

```
from transformers import VisionEncoderDecoderModel, Seq2SeqTrainer,
Seq2SeqTrainingArguments

model =
VisionEncoderDecoderModel.from_pretrained("microsoft/trocr-base-stage1")
```

This base model can be used straight away, but will not work correctly for all use cases. We use the **Seq2SeqTrainer** class to fine-tune the model with our dataset, in our case it is *image -> text* dataset. It is important to note that the dataset must contain images of a certain resolution and color space for the specific model, so preprocessing steps are needed.

Once training is done, we can load the new model weights in our application and used by passing images to it. It is as simple as using OpenCV to read the image and pass it straight through the model pipeline:

```
import cv2
img = cv2.imread("plate4.png")

#Process image
processed_out = processor(img,
return_tensors="pt")['pixel_values'].to('cuda')

#Transform to tokens - OCR
transformed = model.generate(processed_out)

#Decode - convert tokens to text
text = processor.batch_decode(transformed, skip_special_tokens=True)
```



---

## VII. Tools and Technologies Used

This section lists the technology I used in both of the projects, and what I learned while using them.



### *OpenCV*

A very comprehensive package of vision algorithms that help in image processing, manipulation and transformation. Take a video feed and it can be decomposed to a simple representation that can be fed directly into an ML model for any task (such as vehicle tracking). It is technically complex with a very steep learning curve, a lot of methods that modify the image in various ways, but once the goal is realized, it is easy to use the correct set of functions to achieve the desired result. For example, image de-noising requires various transformations called **morphological transformations** that slowly and subtly denoise the image with fine tuning.



### *Huggingface Transformers*

An important library in the Machine Learning world that collects many categories of ML models together into an easy-to-use and easy-to-learn structure. It has a “hub” that users can submit their own ML models for others to use via this library. There are text-based models (NLP, text-text) that can be useful in making chat bots, and image-based models (Image-text, Image-Image) that can be used in for example, Detection task as used for the Vehicle detection task.



### *PyTorch*

Used as a result of using 🤗Huggingface Transformers. It is a framework for ML models from start to end. You can develop a model from scratch, or work with an existing model to fine-tune it. In our case, we can fine-tune the TrOCR model to fit our need for ANPR by supplying our own dataset. It has native support for using dedicated hardware such as a GPU or even TPU to accelerate the training process, which was done on Google Colab notebooks.



### *WebRTC*

Web Real-time communication is a new web API standard that gives browsers access to media capability for real-time, aka Live video and audio streaming. It simplifies **peer-to-peer** communication, so that audio and video, and even text can be streamed between the application and the server without worrying about streaming codecs, NAT bypassing, etc. as this is all taken care of by the standard.



### *AioHTTP + AioRTC*

**AioHTTP** is an asyncio-based Python web application framework designed to serve content and create API very easily. It includes server framework and client request generation support. An added extension package, **AioRTC** can be used to add WebRTC support that can very easily serve audio/video content to other WebRTC peers. Video is fed using the camera source via OpenCV,

---

and is encoded in real-time to a live video format. AioHTTP helped in creating the necessary APIs and Jinja template engine used to create the dashboard.



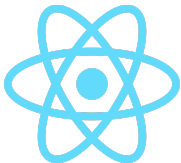
### ***Rasa Bot Framework***

A set of APIs and pre-built models designed to make it easy to create interactive chatbots using nothing more than **.yaml** scripts. The required inputs are: **intents** for figuring out what the user is asking to do, **NLU** for understanding sentence structure, extracting entities and synonyms, and finally **actions** that performs the task that fulfills the user's intent. Making these scripts is a process of understanding user goals and planning for scenarios of user following a certain set of steps. For example, asking for information on a sensor device needs its name and location, which needs to be provided by the user. This can be done with sequential questions, or all at once in one sentence. It uses an NLU model that needs to be trained with our training data and the model predicts the action only from user input text. It can also store context of the conversation so that future questions can use this **short-term memory** instead of asking again.



### ***PostgreSQL***

A database powerhouse that can store data in many formats, provides a good learning curve and is easy to configure. PostgreSQL can act as a simple Relational DB such as MySQL/MariaDB, but when needed it can be very powerful. It can handle any kind of data, be it primitive like string or even JSON. It has a slightly different structure, where your tables are stored in a **Schema object**, and the Schema itself is stored in a **Database object**. This makes projects with multiple services have their own schema within the same database, but still isolated from other database objects.



### ***React framework***

A frontend framework that focuses on being modular, having a component structure for basic elements, such as buttons, panels, etc. Material UI is also a good companion library for creating fantastic UIs. React needs NodeJS to run and build into a static webpage. Hosting a React app for production is not straight-forward, as a WebServer such as Nginx is needed to host the generated page after building the app, instead of using the development server.



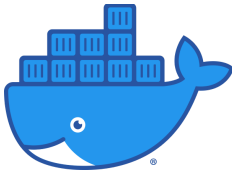
### ***SQLAlchemy***

It is referred to as an ORM - Object relational mapping - which means every table and every row in them is equivalent to a Python object. If a table of users is required, a class called **User** will be used for it, and any object of **User** will in fact be a row in that table. SQLAlchemy can be used to make the database queries agnostic to what DB engine you are using. If you swap from PostgreSQL to another such as MariaDB, it can be done with minimal change as SQLAlchemy will handle the connection and sessions.



### ***FastAPI***

It can be seen as “yet another backend framework” akin to Flask or Django, but FastAPI is actually fast, in both developing an app and running it. It uses **Pydantic** for runtime type checking so that input can be validated, async code which threw me off at first but I got used to it. It was used in the backend of the chatbot app. Various APIs are created to interact with the user, moving data between the frontend and the backend using JSON-formatted data, via REST requests.



### ***Docker / docker-compose***

One of the problems in deployment is service management and packaging the application to be deployed on the server. Usually this is done on dedicated machines, but on real-world deployment servers, your app isn’t always the only one running. Many other apps need to be simultaneously running within the budget of one server instance. Installing your app would ruin package dependencies of others as packages are installed globally. **Docker** helps by isolating everything about your app inside a “container”. Just like a VM, it can create its own storage volume, network interface, OS, etc. and it also simplifies deployment using the **docker-compose** tool that runs the app’s services.



### ***Git***

Git is a **version control system** (VCS) that can manage changes to the files in a project. It’s useful in looking back at the project and what it looked like in the past. Besides being a simple time machine, it is extremely useful in working with multiple developers with many components, as Git is a **Distributed VCS** hence code is shared among every developer. When changes are made, they are merged together into the remote branch. It is very likely that two people modify the same file and cause a **merge conflict**, which needs to be resolved by the one pushing the code last. Git is also used in deploying code to the server. When ready with a version, it can be **tagged** and that version can be pulled on the server. This helps to keep track which version is currently deployed.



### ***Jenkins***

It is an Open Source CI/CD tool which is self-hosted. It comes with a web UI and many plugins that can help with deploying the application automatically. It also has Docker support so that testing can be done in containers efficiently, and deployment can also be done in its separate container. It runs tests after every commit done to the repo, and automatically fetches changes and deploys them if test cases pass.

---

## VIII. Conclusions / Summary

The Optical tracking Vehicle Parking solution enables clients to monitor, track and get notified when activity relating to vehicle movement, vehicle registration details and out-of-hours activity occurs. The system offers access to live video-stream using WebRTC. The descent vehicle tracking and gate entry/exit offers clients a database of events for future reference, and alerting when it happens off-hours if specified.

The Statistics chatbot provides the users a simple and intuitive way of performing basic statistical operations in the Web Browser, while also being comprehensive enough to perform tasks such as data loading, understanding user intent in Natural English sentences instead of strict commands. User-centric input will give freedom to enter in any format and still be understood by the system with a particular intent matching the input. Commands can be strung together, such as finding average for data in a certain date range, producing summary of the data, detecting outliers, etc.

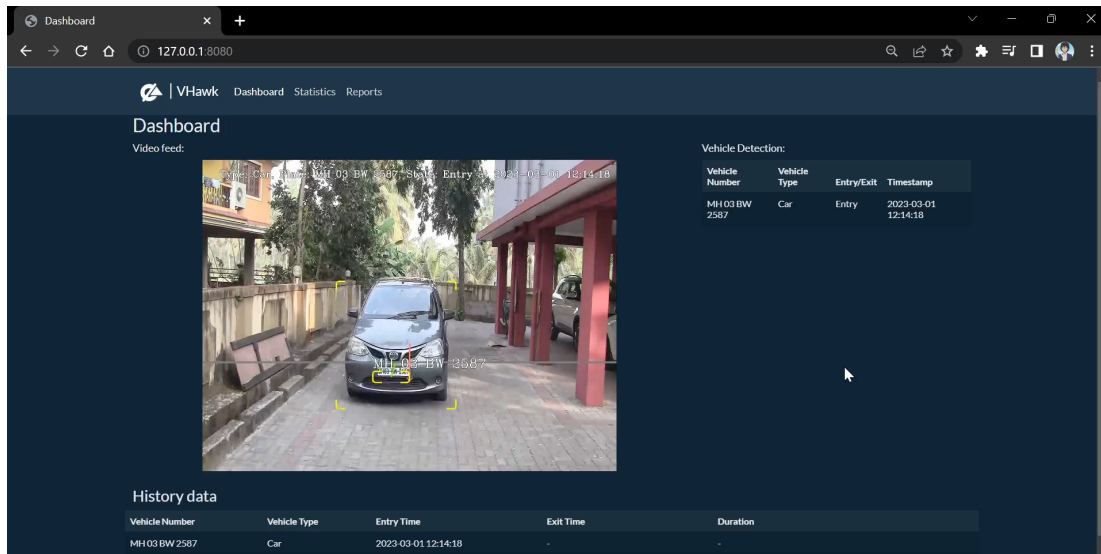
Each of the tools and technologies used during the internship has improved the code flow and development cycle of the apps, as they have greatly improved performance and helped me follow good coding practices in performing the tasks.

Deployment tools such as Docker were used in deploying the Analytics bot to production. I've learned the differences in local development build versus how deployment is done for production, as there are certain steps to be taken to ensure the build is stable by running tests (CI) and is deployed properly on the server (CD). This should be done on every commit that reaches the deployed server to make sure only stable builds are reaching it.

---

## IX. Screenshots / Images

### Vehicle Parking



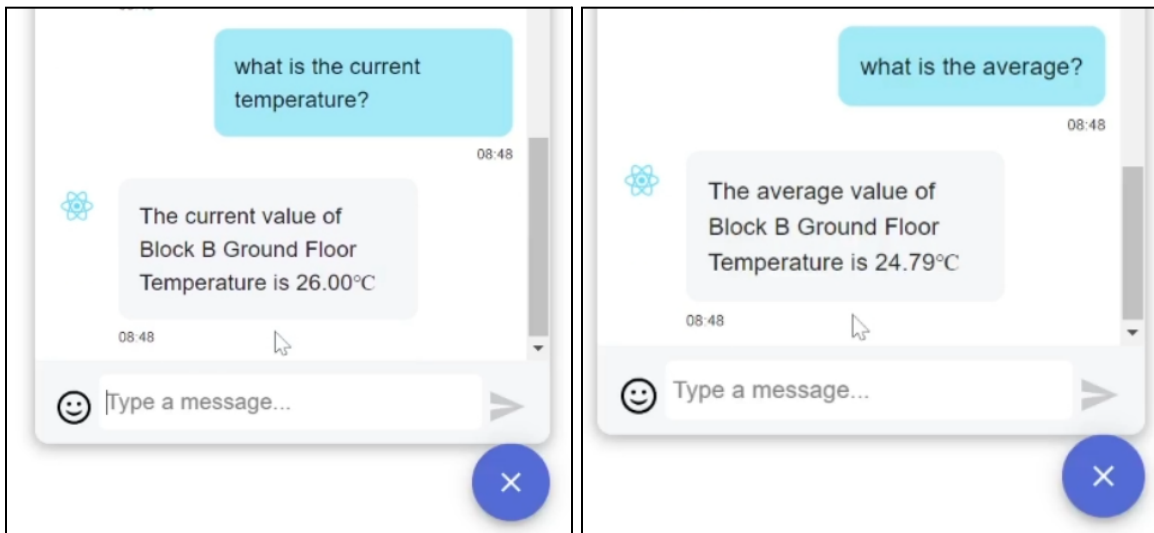
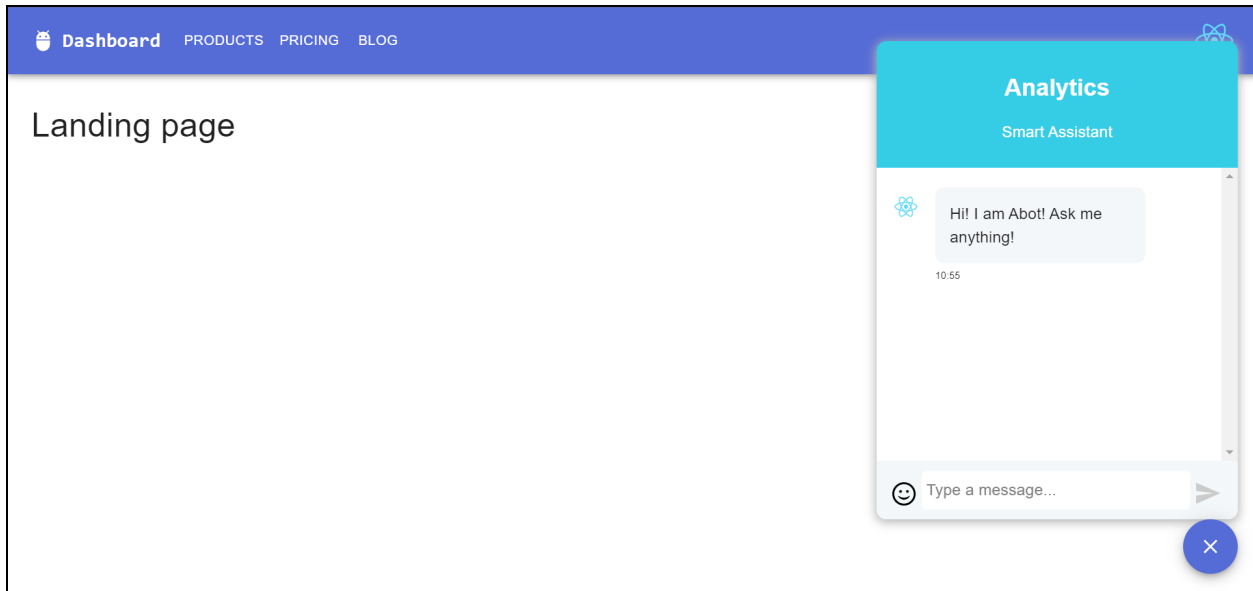
App dashboard

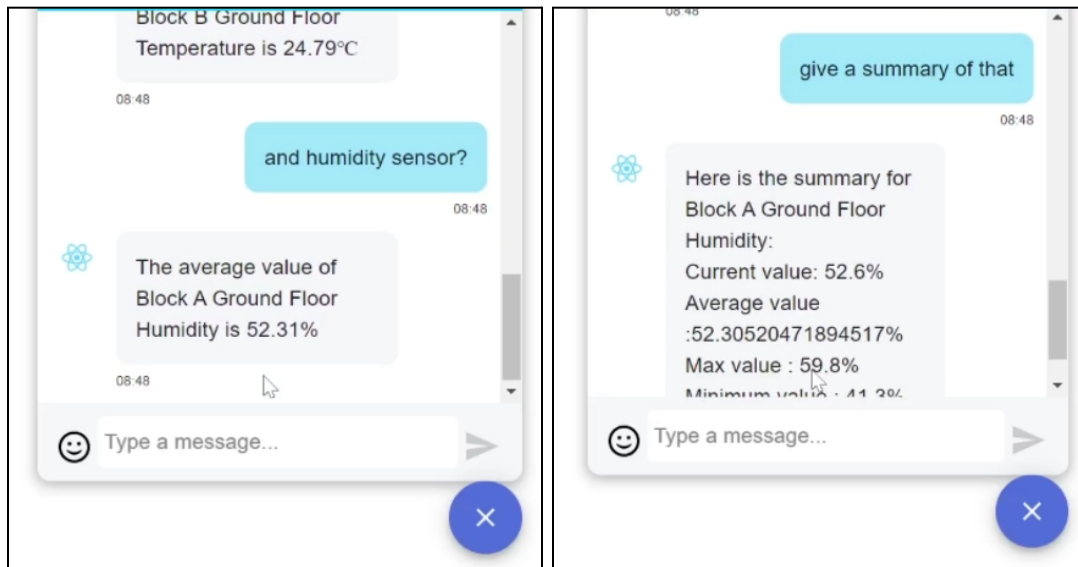


ANPR integration test



## Analytics bot





---

## X. Project Timeline / Weekly Log

### January 2023

#### *Week 1*

- Task 1: Vehicle detection dataset gather, cleaning and colab notebook to train
- Task 2: Vehicle detection model training using YOLOv5
- Task 3: Video input OpenCV, image processing

#### *Week 2*

- Task 1: Model server deployment
- Task 2: Image filtering
- Task 3: Multiprocessing image processing

#### *Week 3*

- Task 1: TrOCR based OCR model
- Task 2: Application testing, issue solving

#### *Week 4*

- Task 1: TrOCR usage and training
- Task 2: TrOCR fine-tuning for Indian Number-plates

#### *Week 5 (end of month)*

- Task: TrOCR training task

### February 2023

#### *Week 1*

- Task 1: Test models on recorded footage
- Task 2: Entry/exit detection
- Task 3: TrOCR number plate detection task

#### *Week 2*

- Task 1: Detection issue Light-level
- On-boarding discussion of Analytics bot
- Task 2: Entry/exit detection
- Task 3: UI/dashboard build

---

### *Week 3*

- Task 1: Live video camera input
- Task 2: RASA bot usage learning

### *Week 4*

- Task 3: Rasa custom action create and demo
- Discuss design plan of the bot (UI, deployment, scope)

### *Week 5 (end of month)*

- Task 1: Start Chat bot demo UI with widget using React + MUI
- Task 2: App data flow / component diagrams create

## **March 2023**

### *Week 1*

- Task 1: Rasa doc writeup update
- Task 2: JIRA update task
- Task 3: Vehicle demo footage record with number plate
- Meeting on Chatbot: Design discussion, update design doc to add requirements, tasks, etc.

### *Week 2*

- Task 1: Rasa Bot design Doc: code writing - React UI for chat bot PoC using widget
- Task 2: Rasa Agent/endpoint: UI component in frontend with a layout design, research issues (version conflict, outdated modules)
- Meeting: Discussion of Rasa backend connection
- Task 3: UI PoC using widget for the bot, perform integration

### *Week 3*

- Meeting: PoC demo of client UI
- Task 1: NLU trainer, chat bot others in UI framework components
- Task 2: Research - plotted graph from image decompose to data points and interpolation

### *Week 4*

- Task 1: Plot curve from image (estimation)
- Task 2: Deployment: Git setup, repo create, project tasks assign to team (JIRA), frontend side changes to be able to work on git with deployment (CI/CD) research

---

### *Week 5 (end of month)*

- Task 1: Git setup, create all repos with submodules and init with templates
- Task 2: Jira task update: add to backlog, update and upload documentation to Jira
- Task 3: Frontend write and upload, test on UAT server
- Meeting call for integration

## **April 2023**

### *Week 1*

- Task 1: Tech Assessment/Research: Rasa policy and pipeline for FeatureExtractor and EntityExtractor improvement, try out botpress on UAT.
- Task 2: Deployment: UAT deployment docker setup.

### *Week 2*

- Task 1: Deployment: Fix Rasa deployment issues, run Frontend from docker, install docker-compose and adjust the setup.
- Task 2: Meeting: Docker and git merge.

### *Week 3*

- Task 1: Deployment: Install and configure Jenkins CI, configure git project for push webhook, add CD pipeline for docker deployment.
- Task 2: Meeting: catch up call on the progress, discuss DevOps and MLOps deployment research.

### *Week 4*

- Task 1: Development: Rasa FastAPI deployment readiness, proper project structure, and Pydantic implementation, requirements, etc.
- Task 2: Deployment: Resolve UAT server memory issue, clear old docker containers and unused images
- Task 3: SQLAlchemy connection with PostgreSQL, env variables, Pydantic settings
- Task 4: Input time range, id, etc. for conversation

### *Week 5 (end of month)*

- Task 1: Development: Docker deployment on UAT with new changes, integrate Telegram, and work on intent misclassification.
- Task 2: Duckling server for datetime, number parsing

- 
- Task 3: Image generation for report, base64 data transmission trick over JSON, markdown generation fix.

## May 2023

### *Week 1*

- Development: String formatting, refactoring, statistics subproject, report integration.
- Meetings: Code issue discussion, website details, daily updates.

### *Week 2*

- Development: Bug fixes, NLU enhancements, refactoring, deployment requirements.
- Meetings: Daily updates, task discussions, integration catchup.

### *Week 3*

- Development: Backend improvements, bug fixes, Plotly chart generation, Dockerfile enhancements.
- Meetings: Code discussions, demo preparation.

### *Week 4 (end of month)*

- Development: Training, NLU improvements, data loading, schema updates.
- Meetings: Demo progress, pair programming, test cases.



---

## XI. My reflections

During the internship, I realized that coding in the classroom is very different from coding in a work environment. My style of coding does not go well with production coding style. There is a lot more to look after, such as the coding conventions to follow (capitalization of identifiers, are we using camelCase or snake\_case), lots of logging output and communication with peers about the changes you have done. The last one is the most important since if I change something without informing the others, their code might stop working and it will be difficult to troubleshoot exactly why.

I learned about planning, the importance of having a design document to quickly refer to when something seems confusing. Also, sometimes it is okay to not get it correct right away. This is the way of development. We need to debug, ask questions to my awesome peers and most importantly take breaks when I need to blow off some steam. I usually just play a musical instrument to relieve stress, and that has worked out for me.

Regarding the projects I've worked on, I've learned many new things about Computer Vision and Machine Learning. The recent advancements in both have also helped me in solving problems much easier. Such as, instead of using simple optical tracking algorithms, there are now Deep-learning object trackers that are much faster and more accurate.

This company being in the field of IoT has also brought me many new concepts to look into and also taught me how real-time data is processed. When you get data from tens or hundreds of sensors at once every minute 24/7, you realize the importance of scaling and parallel processing. Message queue protocols like MQTT become vital to use and optimize to be able to handle the volume of data.

One major hurdle during the internship I had was **communication**. I have social anxiety and have trouble talking to people. Due to this, many times there were misunderstandings that could have been avoided and saved a lot of wasted time. When put in the spotlight I had a hard time explaining and demonstrating my work. This is one aspect that I would like to improve.

---

## References

- <https://towardsdatascience.com/custom-dataset-in-pytorch-part-1-images-2df3152895>
- <https://scheele.hashnode.dev/build-a-dockerized-fastapi-application-with-poetry-and-gunicorn>
- <https://fastapi.tiangolo.com/>
- <https://rasa.com/docs/rasa/>
- <https://huggingface.co/>
- <https://docs.opencv.org/4.x/>
- <https://zbigatron.com/mapping-camera-coordinates-to-a-2d-floor-plan/>
- <https://rdmilligan.wordpress.com/2015/06/28/opencv-camera-calibration-and-pose-estimation-using-python/>
- [https://colab.research.google.com/github/NielsRogge/Transformers-Tutorials/blob/master/TrOCR/Fine tune TrOCR on IAM Handwriting Database using Seq2SeqTrainer.ipynb](https://colab.research.google.com/github/NielsRogge/Transformers-Tutorials/blob/master/TrOCR/Fine_tune_TrOCR_on_IAM_Handwriting_Database_using_Seq2SeqTrainer.ipynb)

## Other references

- [1] <https://www.kaggle.com/datasets/ashfakyeafi/road-vehicle-images-dataset>
- [2] <https://www.youtube.com/watch?v=pRBDFNV2qG8>
- [3] <https://rasa.com/docs/rasa/connectors/your-own-website>
- [4] <https://mui.com/material-ui/>
- [5] <https://axios-http.com/>