PROJECT REPORT

TITLE: AI Enable Car Parking using OpenCV

MADE BY:

ABHA RANI (21BAI1678)

SHRADDHA PANDEY (21BAI1673)

ARYAN SETHI (21BCE0704)

PARTH MATHUR (21BCE2607)

INDEX

Project Overview:

Introduction:

The AI-Enabled Car Parking System is a sophisticated solution designed to streamline the process of parking vehicles using Artificial Intelligence (AI) and Computer Vision techniques.

Objective:

The primary objective of this project is to develop a smart parking system that leverages OpenCV, a popular computer vision library, to accurately detect and guide vehicles into parking spaces autonomously.

Key Features:

Real-time vehicle detection and tracking using OpenCV.

Automated parking guidance for drivers.

Integration with a user-friendly interface for both drivers and parking operators.

Data logging and reporting for monitoring and analysis.

Technologies and Tools:

OpenCV: Utilizing the OpenCV library for image processing, object detection, and tracking.

Python: Programming language for implementation.

GUI Framework (optional): For creating a user-friendly interface.

Hardware (optional): Cameras for capturing real-time footage.

Components:

a. Vehicle Detection and Tracking: Implementing object detection algorithms to identify vehicles in real-time camera feed.

Utilizing tracking algorithms to follow vehicle movements within the parking area.

b. Parking Guidance System: Using AI algorithms to calculate optimal parking spots based on vehicle size and available space. Providing visual and/or auditory guidance to drivers for parking maneuvers.

c. User Interface: Creating a user-friendly interface for drivers to interact with the system (optional). Providing information about available parking spaces, guidance instructions, and feedback.

d. Data Logging and Reporting: Implementing a system to log data on parking statistics, such as occupancy, duration, and frequency. Generating reports for analysis and optimization.

Workflow:

a. Vehicle Detection: Capture real-time video feed from cameras placed in the parking area. Apply image processing techniques to detect vehicles.

b. Tracking: Implement tracking algorithms to monitor vehicle movements.

c. Parking Guidance: Analyse available parking spots based on detected vehicles and free spaces. Provide guidance to drivers through visual indicators and/or audio cues.

d. User Interface (optional): Create an intuitive interface for drivers to interact with the system.

e. Data Logging and Reporting: Record parking data for analysis and reporting purposes.

Testing and Validation:

Conduct rigorous testing to ensure accurate vehicle detection and parking guidance.

Validate system performance under various conditions (e.g., different lighting, weather, vehicle sizes).

Challenges:

Environmental factors (e.g., lighting conditions, weather) affecting object detection.

Real-time processing requirements for responsive parking guidance.

Benefits:

Improved parking efficiency, reducing congestion and minimizing time spent searching for parking spaces.

Enhanced user experience for drivers, leading to increased customer satisfaction.

Future Enhancements:

Integration with IoT devices for real-time updates and notifications.

Incorporation of advanced machine learning techniques for improved accuracy.

Conclusion:

The AI-Enabled Car Parking System utilizing OpenCV represents a cutting-edge solution to address the challenges of parking in crowded urban environments. By harnessing the power of computer vision and artificial intelligence, this system promises to revolutionize the parking experience for both drivers and parking operators.

AI Enable Car Parking Using OpenCV

Car parking is a common problem faced by drivers in busy urban areas. For example, imagine you are driving to a shopping mall during peak hours. As you approach the mall, you notice that the parking lot is full, and several other cars are circling around looking for available spots.

You join the queue of cars, hoping to find an available spot soon. However, as time passes, you realize that the parking lot is overcrowded, and it's becoming increasingly difficult to find a spot. You start to feel frustrated and anxious, knowing that you might be late for your appointment or miss out on a great shopping opportunity.

AI-enabled car parking using OpenCV is a computer vision-based project that aims to automate the parking process. The project involves developing an intelligent system that can identify empty parking spaces and it gives the count of available parking spots.

The system uses a camera and OpenCV (Open Source Computer Vision) library to capture live video footage of the parking lot.

Purpose:

The purpose of a project on AI-enabled car parking using OpenCV is to create an advanced parking system that leverages artificial intelligence and computer vision technologies to streamline the process of parking vehicles. The project aims to address several key objectives:

Efficient Space Utilization: By utilizing computer vision techniques, the system can accurately detect and track vehicles in real-time. This allows for optimized parking space allocation, ensuring that each spot is used efficiently.

Improved User Experience: The project aims to provide a user-friendly interface for drivers, offering guidance on available parking spaces and assisting them in parking their vehicles. This enhances the overall experience for drivers and reduces the time and frustration associated with finding parking.

Reduced Congestion and Traffic: By facilitating quicker and more efficient parking, the system helps to alleviate congestion in parking lots and on surrounding roads. This can lead to reduced traffic and improved flow in busy areas.

Enhanced Safety: The project can contribute to improved safety in parking areas by providing visual or auditory guidance to drivers, minimizing the risk of accidents or collisions during parking manoeuvres.

Problem Statement:

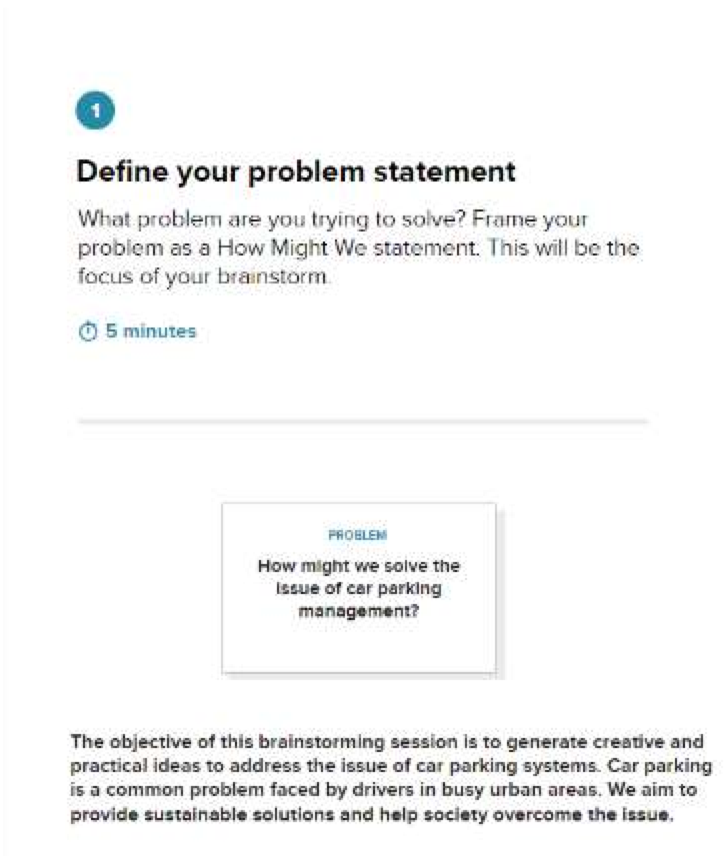
The existing car parking systems in urban environments face significant challenges in terms of space optimization, time efficiency, and user convenience. Conventional systems often allocate parking spaces without considering vehicle dimensions, leading to inefficient use of available parking real estate. Drivers spend considerable time searching for suitable spots, causing congestion and increased environmental impact due to prolonged idling. Manoeuvring vehicles in confined spaces also poses safety concerns. Moreover, the lack of clear guidance exacerbates the parking process. Additionally, the absence of comprehensive data collection and analysis capabilities hinders parking operators' ability to make informed decisions about space allocation and utilization. In light of these issues, this project aims to develop an AI-enabled car parking system using OpenCV, employing computer vision and artificial intelligence techniques to address these challenges and enhance the overall parking experience for both drivers and operators. The system will provide real-time vehicle detection and tracking, automated parking guidance, a user-friendly interface, and robust data logging and reporting functionalities.

Ideation Phase
Brainstorm & Idea Prioritization

Date	18 October 2023
Team ID	Team-593009
Project Name	AI-enabled car parking system using OpenCV
Maximum Marks	4 Marks

Defining our problem statement and brainstorming ideas

Brainstorming ideas is a creative process where a group generates a list of potential solutions, suggestions, or concepts for a specific problem or project.


1

Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM

How might we solve the issue of car parking management?

The objective of this brainstorming session is to generate creative and practical ideas to address the issue of car parking systems. Car parking is a common problem faced by drivers in busy urban areas. We aim to provide sustainable solutions and help society overcome the issue.

2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Abha

Parking Attendant Assistance: Employ parking attendants to guide drivers to available spots and assist with parking maneuvers.

Drone-Based Surveillance: Employ drones equipped with cameras to survey parking areas and identify vacant spots, especially in large or remote locations.

Using Computer Vision Techniques (OpenCV): Install cameras with a top-down view of the parking area. Utilize computer vision algorithms to analyze the video feed and identify vacant parking spaces.

Shraddha

Ground-Level Sensors: Employ ground-level sensors, such as ultrasonic or magnetic sensors, to detect the presence or absence of vehicles in individual parking spots.

Crowdsourcing: Allowing drivers to share information about available parking spots on app with others .

Aryan

Parking Data Analytics: Analyze parking data to identify usage patterns and occupancy enabling data-driven parking management decisions.

Parking Spot Markings: Use clear and visible parking spot markings to make it easier for drivers to identify vacant spaces.

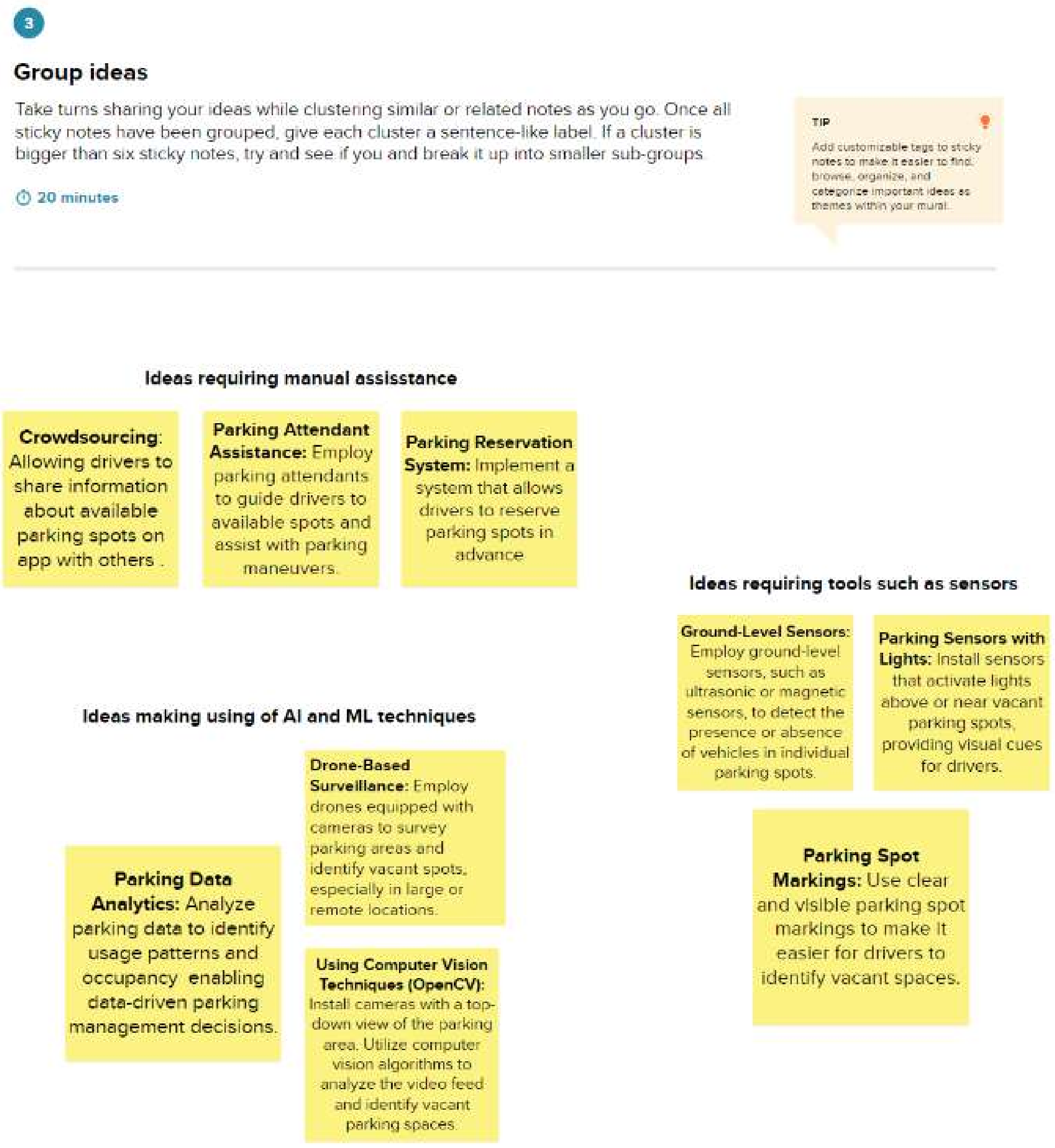
Parth

Parking Sensors with Lights: Install sensors that activate lights above or near vacant parking spots, providing visual cues for drivers.

Parking Reservation System: Implement a system that allows drivers to reserve parking spots in advance

Clustering the ideas

Clustering involves grouping ideas of similar categories together.



3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

Ideas requiring manual assisstance

Crowdsourcing: Allowing drivers to share information about available parking spots on app with others .

Parking Attendant Assistance: Employ parking attendants to guide drivers to available spots and assist with parking maneuvers.

Parking Reservation System: Implement a system that allows drivers to reserve parking spots in advance

Ideas requiring tools such as sensors

Ground-Level Sensors: Employ ground-level sensors, such as ultrasonic or magnetic sensors, to detect the presence or absence of vehicles in individual parking spots.

Parking Sensors with Lights: Install sensors that activate lights above or near vacant parking spots, providing visual cues for drivers.

Parking Spot Markings: Use clear and visible parking spot markings to make it easier for drivers to identify vacant spaces.

Ideas making using of AI and ML techniques

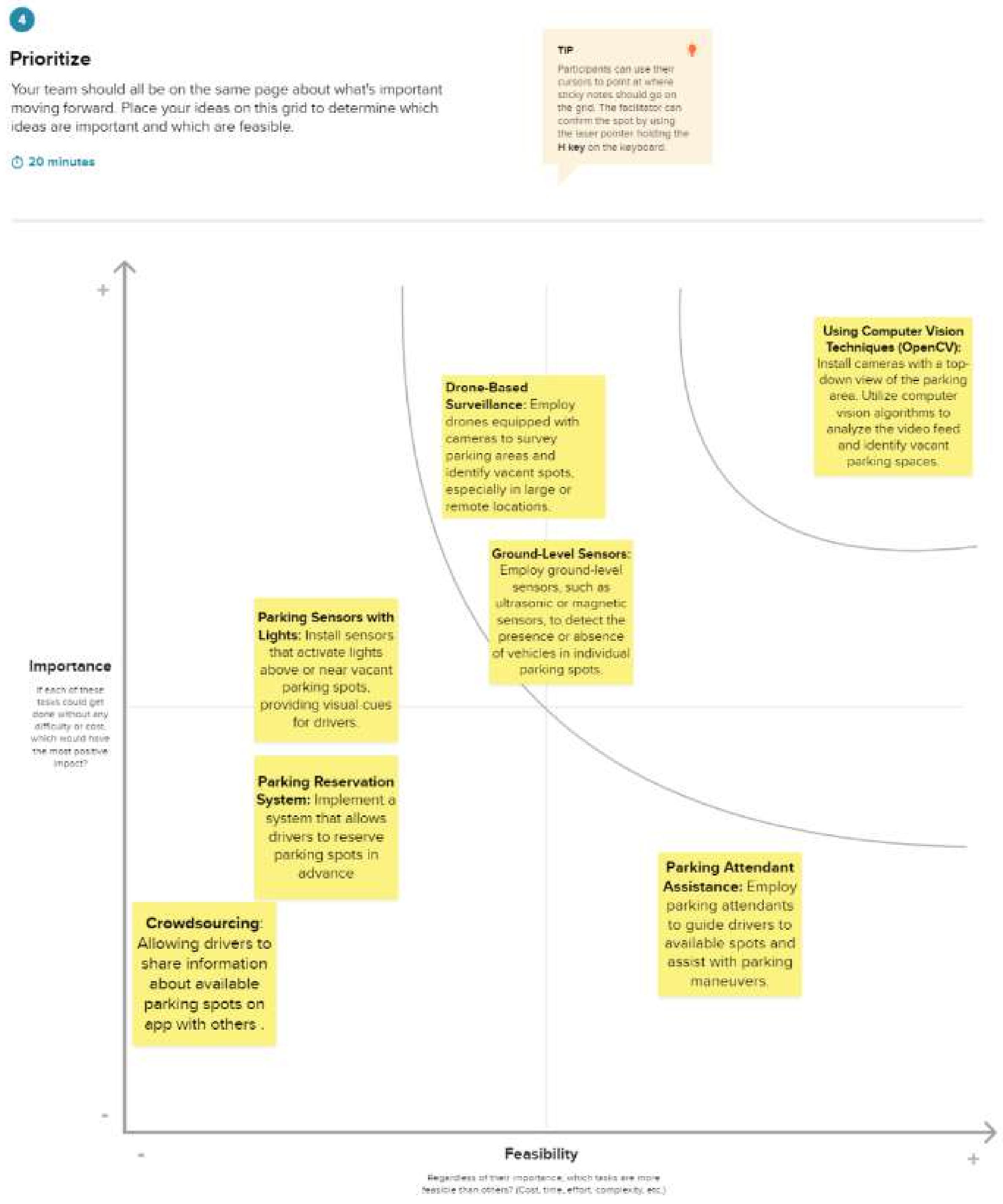
Drone-Based Surveillance: Employ drones equipped with cameras to survey parking areas and identify vacant spots, especially in large or remote locations.

Parking Data Analytics: Analyze parking data to identify usage patterns and occupancy enabling data-driven parking management decisions.

Using Computer Vision Techniques (OpenCV): Install cameras with a top-down view of the parking area. Utilize computer vision algorithms to analyze the video feed and identify vacant parking spaces.

Idea Prioritization

Idea prioritization is the process of ranking or assessing ideas based on specific criteria (in this case importance and feasibility) to determine which ideas should be implemented or pursued first.



4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the **H key** on the keyboard.

Using Computer Vision Techniques (OpenCV): Install cameras with a top-down view of the parking area. Utilize computer vision algorithms to analyze the video feed and identify vacant parking spaces.

Drone-Based Surveillance: Employ drones equipped with cameras to survey parking areas and identify vacant spots, especially in large or remote locations.

Ground-Level Sensors: Employ ground-level sensors, such as ultrasonic or magnetic sensors, to detect the presence or absence of vehicles in individual parking spots.

Parking Sensors with Lights: Install sensors that activate lights above or near vacant parking spots, providing visual cues for drivers.

Parking Reservation System: Implement a system that allows drivers to reserve parking spots in advance.

Crowdsourcing: Allowing drivers to share information about available parking spots on app with others .

Parking Attendant Assistance: Employ parking attendants to guide drivers to available spots and assist with parking maneuvers.

+
Importance
If each of these tasks could get done without any difficulty or cost, which would have the most positive impact?
-
-
Feasibility
Regardless of their importance, which tasks are more feasible than others? (Cost, time, effort, complexity, etc.)
+

From the prioritization process, we can see that an AI-enabled car parking system using OpenCV is a compelling choice and would be the best approach.

Firstly, this idea addresses a significant and prevalent issue in urban areas: parking congestion and inefficiency. By leveraging artificial intelligence and computer vision, we can optimize parking space utilization, reduce driver frustration, and improve overall traffic flow. This means that drivers in urban settings will see a substantial reduction in time and effort spent trying to find parking spots.

Moreover, OpenCV is a well-established and widely used open-source library for computer vision applications. This provides a robust and accessible foundation for developing our parking system, reducing development costs and time to market.

Additionally, this project aligns with our commitment to smart city initiatives and sustainable transportation solutions. By streamlining parking processes, we can reduce vehicle emissions from unnecessary searching, minimize congestion-related pollution, and promote a more efficient transportation infrastructure.

Lastly, this initiative presents an opportunity for innovation and technological advancement. By developing an AI-powered parking system, we not only address a pressing urban challenge but also position ourselves as leaders in the field of smart parking solutions.

In conclusion, the selection of an AI-enabled car parking system using OpenCV as our project is a strategic decision based on its high impact potential, feasibility, and alignment with our smart city goals. We are confident that this choice will contribute to a more efficient, sustainable, and user-friendly urban transportation landscape.

Ideation Phase
Empathize & Discover

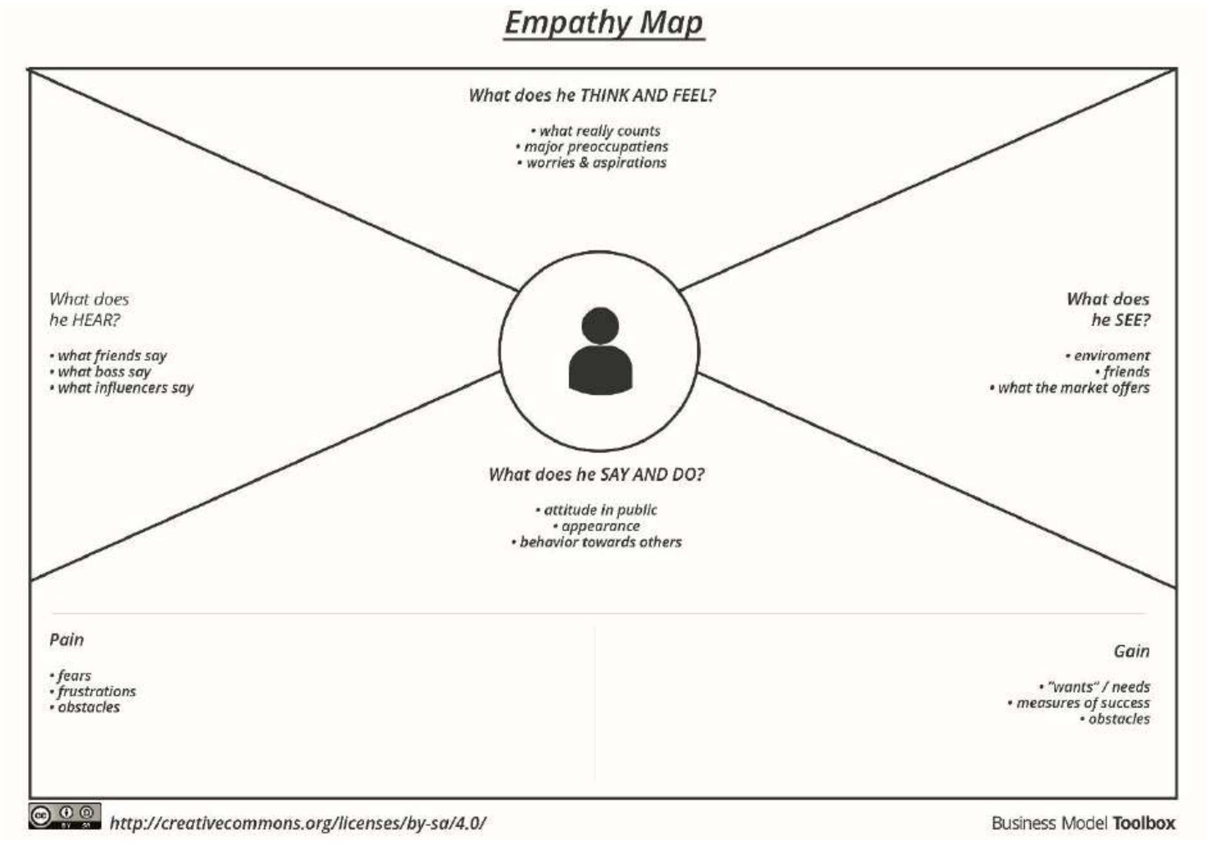
Date	19 September 2022
Team ID	PNT2022TMIDxxxxxx
Project Name	Project - xxx
Maximum Marks	4 Marks

Empathy Map Canvas:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes.

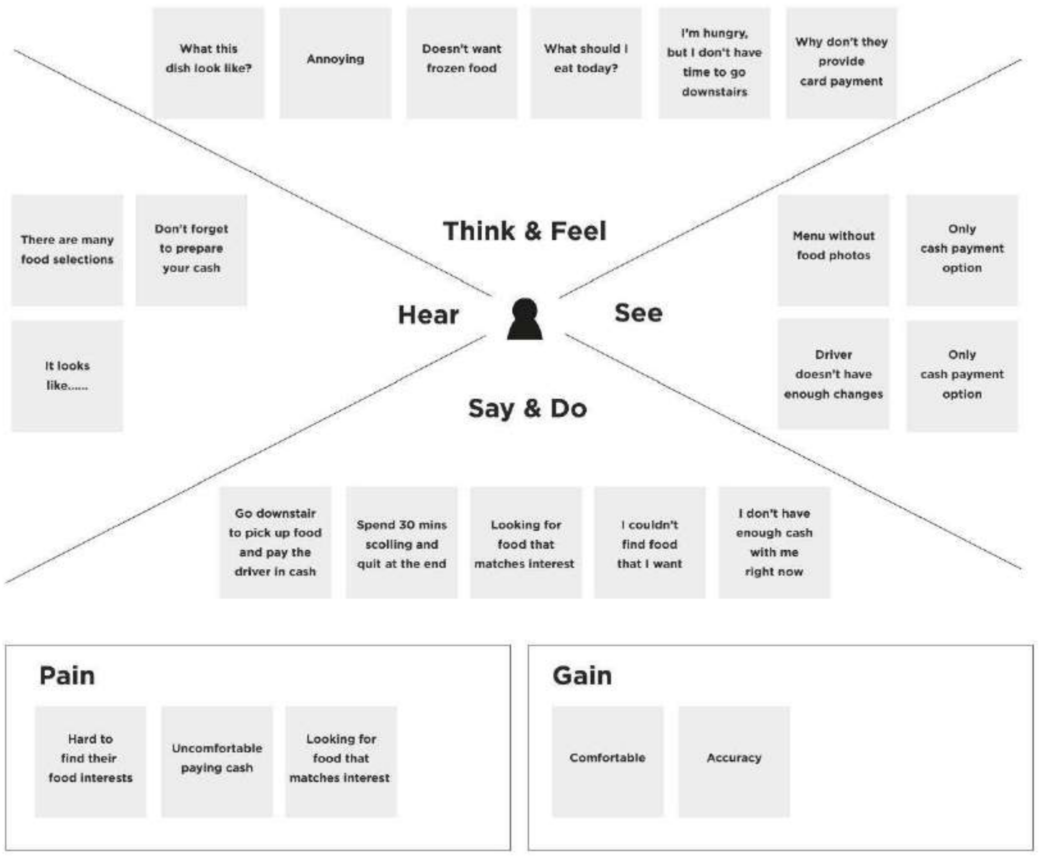
It is a useful tool to helps teams better understand their users.
Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.

Example:



Reference: https://www.mural.co/templates/empathy-map-canvas

Example: Food Ordering & Delivery Application


What this dish look like?
Annoying
Doesn't want frozen food
What should I eat today?
I'm hungry, but I don't have time to go downstairs
Why don't they provide card payment
Think & Feel
There are many food selections
Don't forget to prepare your cash
It looks like......
Hear
Menu without food photos
Only cash payment option
Driver doesn't have enough changes
Only cash payment option
See
Say & Do
Go downstair to pick up food and pay the driver in cash
Spend 30 mins scolling and quit at the end
Looking for food that matches interest
I couldn't find food that I want
I don't have enough cash with me right now
Pain
Hard to find their food interests
Uncomfortable paying cash
Looking for food that matches interest
Gain
Comfortable
Accuracy

Ideation Phase

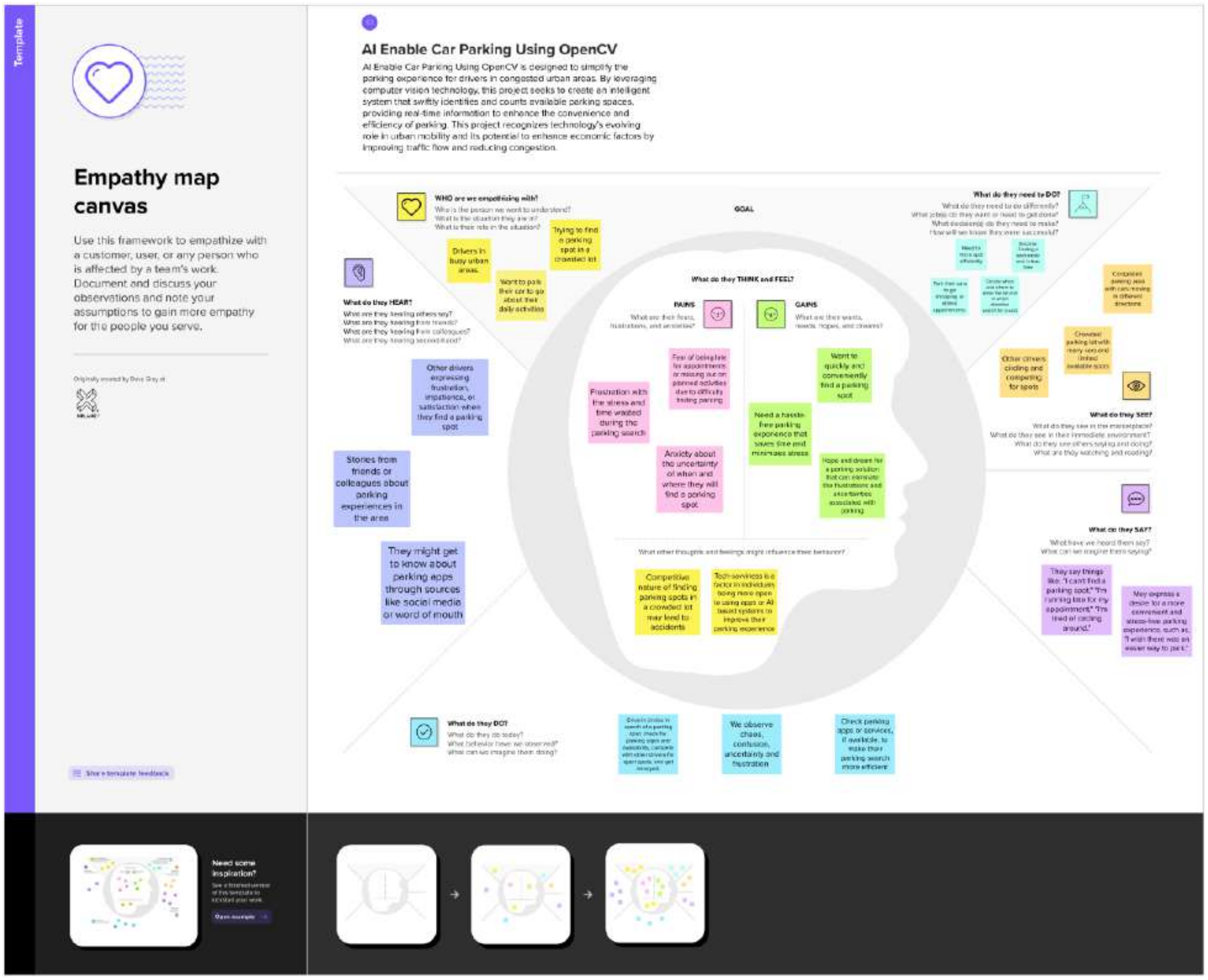
Empathize & Discover

Date	18 October 2023
Team ID	Team-593009
Project Name	AI Enable car parking using OpenCV
Maximum Marks	5 Marks

Empathy Map Canvas:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviors and attitudes. It is a useful tool to helps teams better understand their users.

Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.



Template

Empathy map canvas

Use this framework to empathize with a customer, user, or any person who is affected by a team's work. Document and discuss your observations and note your assumptions to gain more empathy for the people you serve.

AI Enable Car Parking Using OpenCV

AI Enable Car Parking Using OpenCV is designed to simplify the parking experience for drivers in congested urban areas. By leveraging computer vision technology, this project seeks to create an intelligent system that swiftly identifies and counts available parking spaces, providing real-time information to enhance the convenience and efficiency of parking. This project recognizes technology's evolving role in urban mobility and its potential to enhance economic factors by improving traffic flow and reducing congestion.

WHO are we empathizing with?
Who is the person we want to understand?
What is the situation they are in?
What is their role in the situation?

- Drivers in busy urban areas
- Want to park their car to go about their daily activities
- Trying to find a parking spot in a crowded lot

GOAL

What do they need to DO?
What do they need to do differently?
What job(s) do they want or need to get done?
What decision(s) do they need to make?
How will we know they were successful?

What do they HEAR?
What are they hearing others say?
What are they hearing from friends?
What are they hearing from colleagues?
What are they hearing second-hand?

- Other drivers expressing frustration, impatience, or satisfaction when they find a parking spot
- Stories from friends or colleagues about parking experiences in the area
- They might get to know about parking apps through sources like social media or word of mouth

What do they THINK and FEEL?

PAINS
What are their fears, frustrations, and anxieties?

- Fear of being late for appointments or missing out on planned activities due to difficulty finding parking
- Frustration with the stress and time wasted during the parking search
- Anxiety about the uncertainty of when and where they will find a parking spot

GAINS
What are their wants, needs, hopes, and dreams?

- Want to quickly and conveniently find a parking spot
- Need a hassle-free parking experience that saves time and minimizes stress
- Hope and dream for a parking solution that can eliminate the frustrations and uncertainties associated with parking

What other thoughts and feelings might influence their behavior?

- Competitive nature of finding parking spots in a crowded lot may lead to accidents
- Tech-savviness is a factor in individuals being more open to using apps or AI based systems to improve their parking experience

What do they SEE?
What do they see in the marketplace?
What do they see in their immediate environment?
What do they see others saying and doing?
What are they watching and reading?

- Congested parking area with cars moving in different directions
- Crowded parking lot with many cars and limited available spots
- Other drivers circling and competing for spots

What do they SAY?
What have we heard them say?
What can we imagine them saying?

- They say things like: "I can't find a parking spot." "I'm running late for my appointment." "I'm tired of circling around."
- May express a desire for a more convenient and stress-free parking experience, such as, "I wish there was an easier way to park."

What do they DO?
What do they do today?
What behavior have we observed?
What can we imagine them doing?

- We observe chaos, confusion, uncertainty and frustration
- Check parking apps or services, if available, to make their parking search more efficient

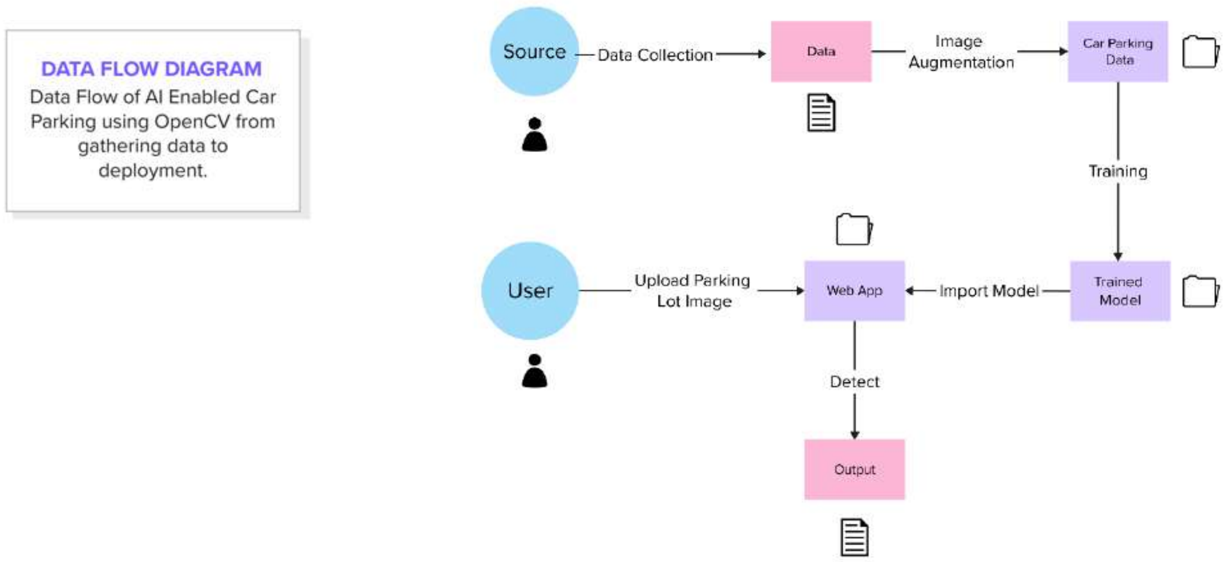
Project Design Phase-II

Data Flow Diagram & User Stories

Date	23-10-2023
Team ID	Team-593009
Project Name	AI Enable car parking using OpenCV
Maximum Marks	4 Marks

Data Flow Diagram

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance Criteria	Priority	Release
Car Parking Operators and Service Providers	Project setup & Infrastructure	USN-1	**Set up the development environment** with the required tools and frameworks to develop and deploy the AI-enabled car parking solution using OpenCV.	Successfully configure the development environment with all necessary tools and frameworks.	High	Sprint 1
Municipalities and Local Governments	Data collection	USN-2	**Gather a diverse dataset** of parking lot images captured via cameras in various urban settings to train the AI model effectively.	Collect a diverse dataset with images of different parking scenarios, including crowded lots, empty lots, and various lighting conditions.	High	Sprint 1
Project team	Data preprocessing	USN-3	**Preprocess the collected dataset** by resizing images, normalizing pixel values, and splitting it into training and validation sets.	Successfully preprocess the dataset, ensuring it's ready for model training.	High	Sprint 2

Project team	Model development	USN-4	**Explore and evaluate different computer vision models** using OpenCV for parking spot detection and classification to determine the most suitable model for the project.	Experiment with various OpenCV-based models to choose the most accurate and efficient one.	High	Sprint 2
Project team	Model development	USN-5	**Train the selected model** using the preprocessed dataset and monitor its performance on the validation set.	Train the model and verify its accuracy and effectiveness in detecting empty parking spots.	High	Sprint 3
Project team	Model development	USN-6	**Implement real-time video processing** for continuous parking spot detection and tracking using OpenCV.	Successfully process live video footage to identify and count available parking spots.	Medium	Sprint 3
Project team	Model deployment & Integration	USN-7	**Develop a Flask-based web application** to deploy the trained model. Integrate the model's output into the web app, allowing users to access the available parking spot count	Successfully develop and deploy a web application using Flask that integrates the trained model and provides parking spot information to users.	Medium	Sprint 4

			through a user-friendly interface.			
Testing & quality assurance	Testing & quality assurance	USN-8	**Conduct extensive testing and quality assurance** of the model and the web application to identify and report any issues, bugs, or inaccuracies. Optimize model hyperparameters based on user feedback and testing results.	Conduct thorough testing, ensure the model's accuracy, and fine-tune it based on user feedback and testing results.	Medium	Sprint 5

Project Design Phase-I
Solution Architecture

Date	23 October 2023
Team ID	Team-593009
Project Name	AI-enabled car parking system using OpenCV
Maximum Marks	4 Marks

Solution Architecture:

The AI-enabled car parking solution leverages computer vision and OpenCV to automate the parking process, making it seamless and efficient. It identifies empty parking spaces and provides real-time information on the availability of parking spots, improving the user experience and optimizing parking operations. By harnessing computer vision and OpenCV, this AI-enabled car parking solution enhances parking management efficiency, streamlines the parking experience, and provides valuable insights for optimizing parking lot operations.

1. Data Gathering:

- Utilizes strategically placed cameras within the parking lot to capture a live video feed.
- The system continuously collects video data, ensuring a real-time view of the parking lot.

2. Image Preprocessing:

- Preprocesses each frame from the video feed to enhance image quality and extract key features.
- Techniques like resizing, noise reduction, and contrast adjustment are employed to improve image clarity.

3. Model Building:

- Develops a robust machine learning model, possibly using convolutional neural networks (CNNs), to analyze the preprocessed images.
- The model is trained on labeled data to recognize parking spaces and accurately identify occupied and unoccupied spots.

4. Empty Parking Space Detection:

- The model systematically processes each frame to identify parking spaces.
- It classifies parking spaces as either 'occupied' or 'empty' based on the presence of vehicles.
- Object detection algorithms are employed to precisely locate vehicles within parking spots.

5. Real-Time Analysis:

- The system provides continuous, real-time analysis of the live video feed, actively monitoring parking space status.
- Updates are made to the count of available parking spots in real-time.

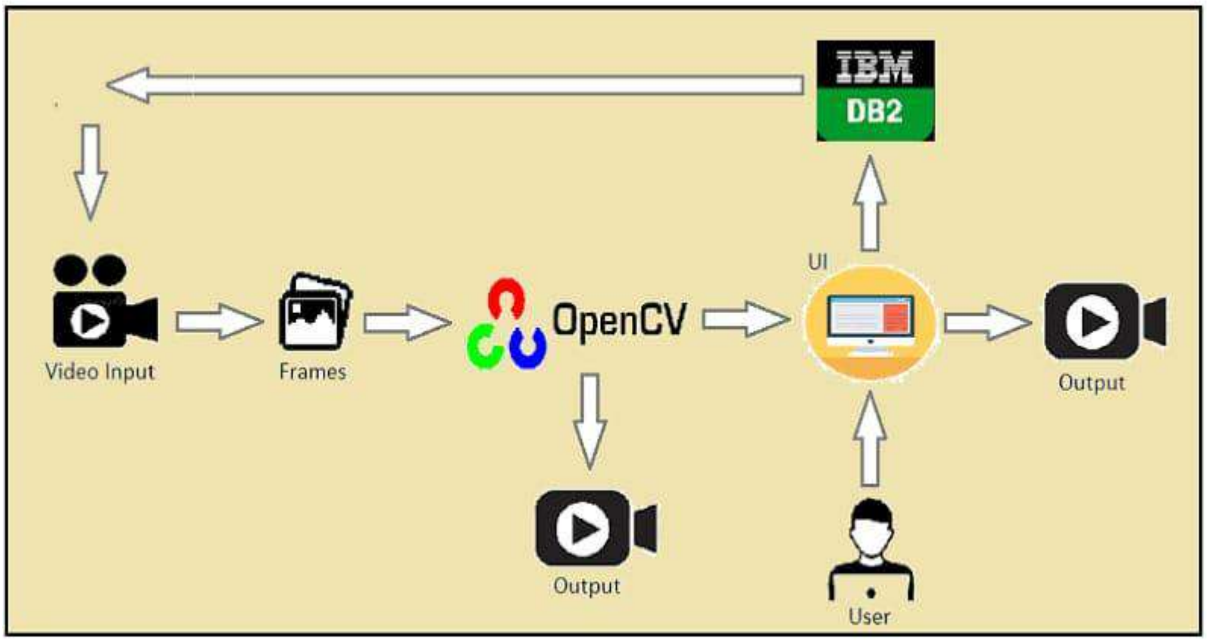
6. User Interface:

- Offers a user-friendly interface, such as a mobile app or a digital display at the parking lot entrance.
- Provides an up-to-the-minute count of available parking spots, ensuring drivers can quickly locate parking spaces.

7. Alerts and Notifications:

- Generates alerts or notifications for parking management personnel and drivers when specific conditions are met, such as the parking lot reaching a certain capacity or detecting unauthorized parking.

Example - Solution Architecture Diagram:



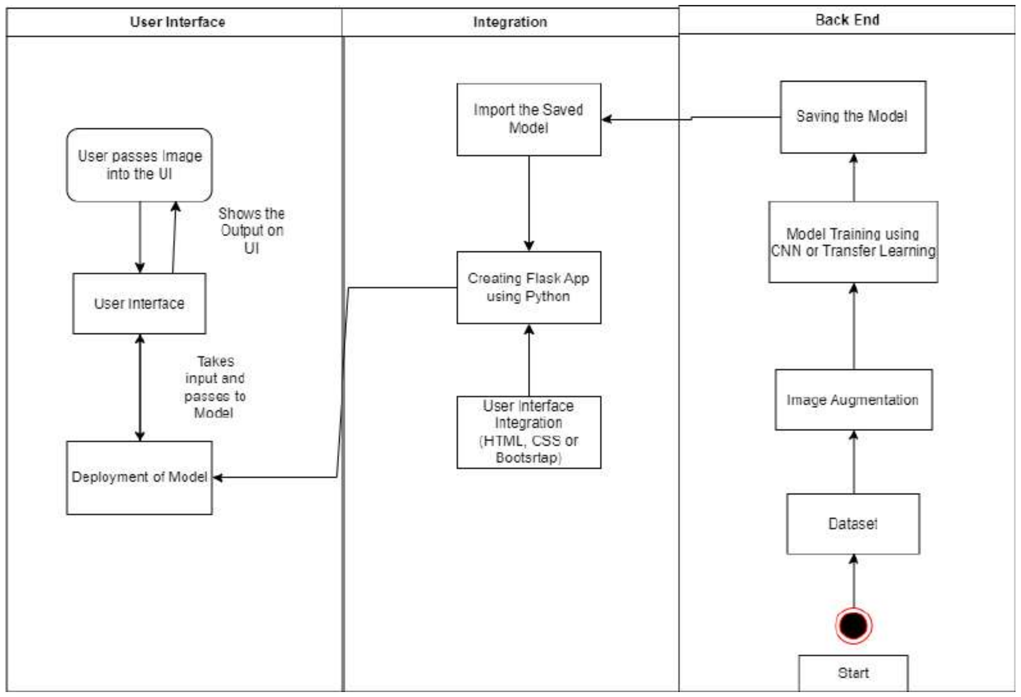
Project Design Phase-II

Technology Stack (Architecture & Stack)

Date	27-10-2023
Team ID	Team-593009
Project Name	AI Enable car parking using OpenCV
Maximum Marks	4 Marks

Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2


User Interface
Integration
Back End
User passes image into the UI
Shows the Output on UI
User Interface
Takes input and passes to Model
Deployment of Model
Import the Saved Model
Creating Flask App using Python
User Interface Integration (HTML, CSS or Bootsrtap)
Saving the Model
Model Training using CNN or Transfer Learning
Image Augmentation
Dataset
Start

Guidelines:

1. Include all the processes (As an application logic / Technology Block)
2. Provide infrastructural demarcation (Local / Cloud)
3. Indicate external interfaces (third party API's etc.)
4. Indicate Data Storage components / services
5. Indicate interface to machine learning models (if applicable)

Table-1 : Components & Technologies:

S.No.	Component	Description	Technology
1.	User Interface	How user interacts with application e.g., Web UI	HTML, CSS, JavaScript, Bootstrap
2.	Application Logic-1	Logic for a process in the application	Python
3.	Database	Collect the Dataset Based on the Problem Statement	File Manager
4.	File Storage/ Data	File storage requirements for Storing the dataset	Local System, Google Drive
5.	Frame Work	Used to Create a web Application, Integrating Frontend and Back End	Python Flask
6.	Deep Learning Model	Purpose of Model	OpenCV, CNN
7.	Infrastructure	Application Deployment on Local System	Local Server

Table: Application Characteristics

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Python’s Flask
2.	Security Implementations	List all the security / access controls implemented,use of firewalls etc.	e.g., SHA-256, Encryptions, IAM Controls, OWASP etc.

3.	Scalable Architecture	Justify the scalability of architecture (3 – tier,Micro-services)	Technology used
4.	Availability	Justify the availability of application (e.g., use ofload balancers, distributed servers etc.)	Technology used
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	Technology used

S.No	**Characteristics**	**Description**	**Technology**
1.	Open-Source Frameworks	Python's Flask or any basic web framework	Python's Flask
2.	Security Implementations	Basic encryption for data transmission, simple access controls	Basic Encryption (e.g., HTTPS), Basic Access Control
3.	Scalable Architecture	Simple, non-distributed architecture	Single-Server Setup
4.	Availability	Local server setup for development and testing	Local Server
5.	Performance	Basic performance considerations for local use	Basic Caching Strategies, Local Testing

Project Planning Phase

Project Planning Template (Product Backlog, Sprint Planning, Stories, Story points)

Date	27-10-2023
Team ID	Team-593009
Project Name	AI Enabled car parking using OpenCV
Maximum Marks	20 Marks

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Project setup & Infrastructure	USN-1	Install and configure cameras and hardware components in the parking lot for real-time video feed.	1	High	Abha
Sprint-1	development environment	USN-2	Set up the development environment with the required tools and frameworks to start the car parking project	2	High	Shraddha
Sprint-2	Object Detection	USN-3	Develop an object detection model to identify cars in real-time using OpenCV and a pre-trained CNN architecture (e.g., YOLO).	5	High	Aryan
Sprint-2	Parking Space Detection	USN-4	Implement a parking space detection model to identify vacant and occupied parking spaces using OpenCV	5	High	Parth
Sprint-3	Real-time Parking Lot Occupancy Detection	USN-5	Integrate the object detection and parking space detection models into the OpenCV pipeline for real-time analysis of parking lot occupancy	6	High	Shraddha
Sprint-3	Updating model to keep count of vacant parking spaces	USN-6	Develop an algorithm to maintain a count of available parking spaces based on real-time data	3	medium	Parth
Sprint-4	Testing & quality assurance	USN-7	Conduct extensive testing under different lighting and weather conditions to ensure system accuracy	5	medium	Aryan
Sprint-5	Model deployment and integration	USN-8	Deploy the AI-enabled car parking system in a real-world parking lot and monitor its performance.	4	medium	Abha

Project Tracker, Velocity & Burndown Chart: (4 Marks)

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	3	2 Days	27 oct 2023	28 oct 2023	10	28 oct 2023
Sprint-2	10	3 Days	29 oct 2023	31 oct 2023		
Sprint-3	9	6 Days	1 nov 2023	6 nov 2023		
Sprint-4	5	3 Days	6 nov 2023	8 nov 2023		
Sprint-5	4	2 Days	8 nov 2023	9 nov 2023		

Velocity:
Imagine we have a 29-days sprint duration, and the velocity of the team is 20 (points per sprint). Let’s calculate the team’s average velocity (AV) per iteration unit (story points per day)

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

AV= 16/10 = 1.6

Burndown Chart:

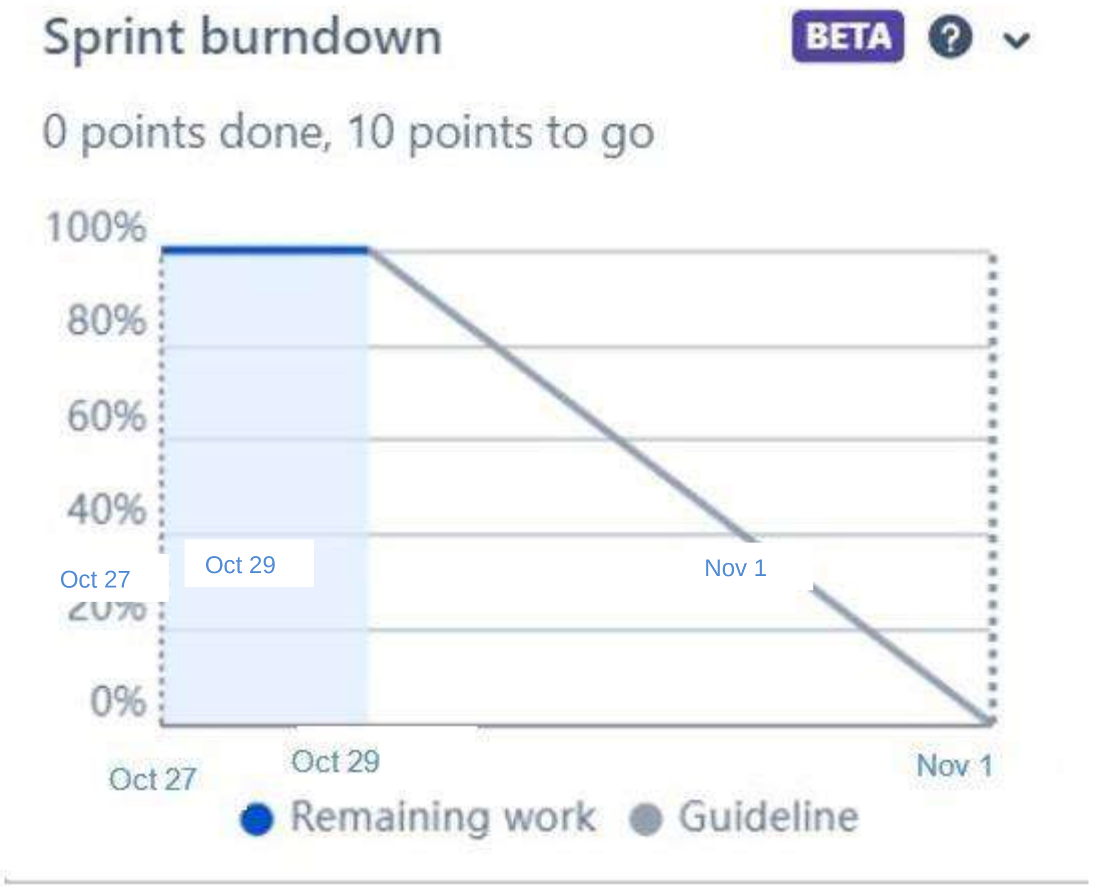
A burndown chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

https://www.visual-paradigm.com/scrum/scrum-burndown-chart/
https://www.atlassian.com/agile/tutorials/burndown-charts

Reference:

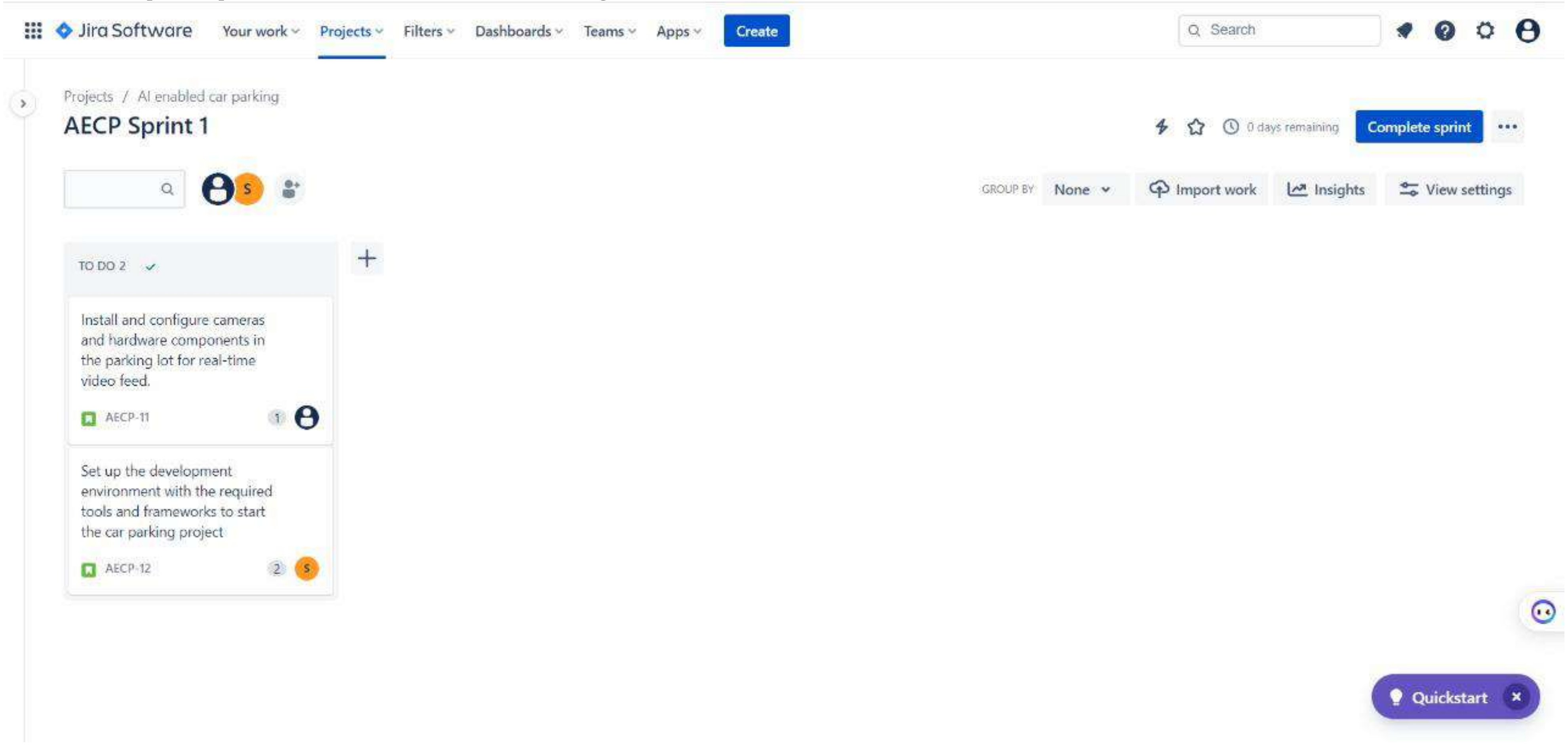
https://www.atlassian.com/agile/project-management
https://www.atlassian.com/agile/tutorials/how-to-do-scrum-with-jira-software
https://www.atlassian.com/agile/tutorials/epics
https://www.atlassian.com/agile/tutorials/sprints
https://www.atlassian.com/agile/project-management/estimation
https://www.atlassian.com/agile/tutorials/burndown-charts

Burndown Chart:



Board section.

We have completed sprint 1 and 2. So we can see the remaining tasks on board.



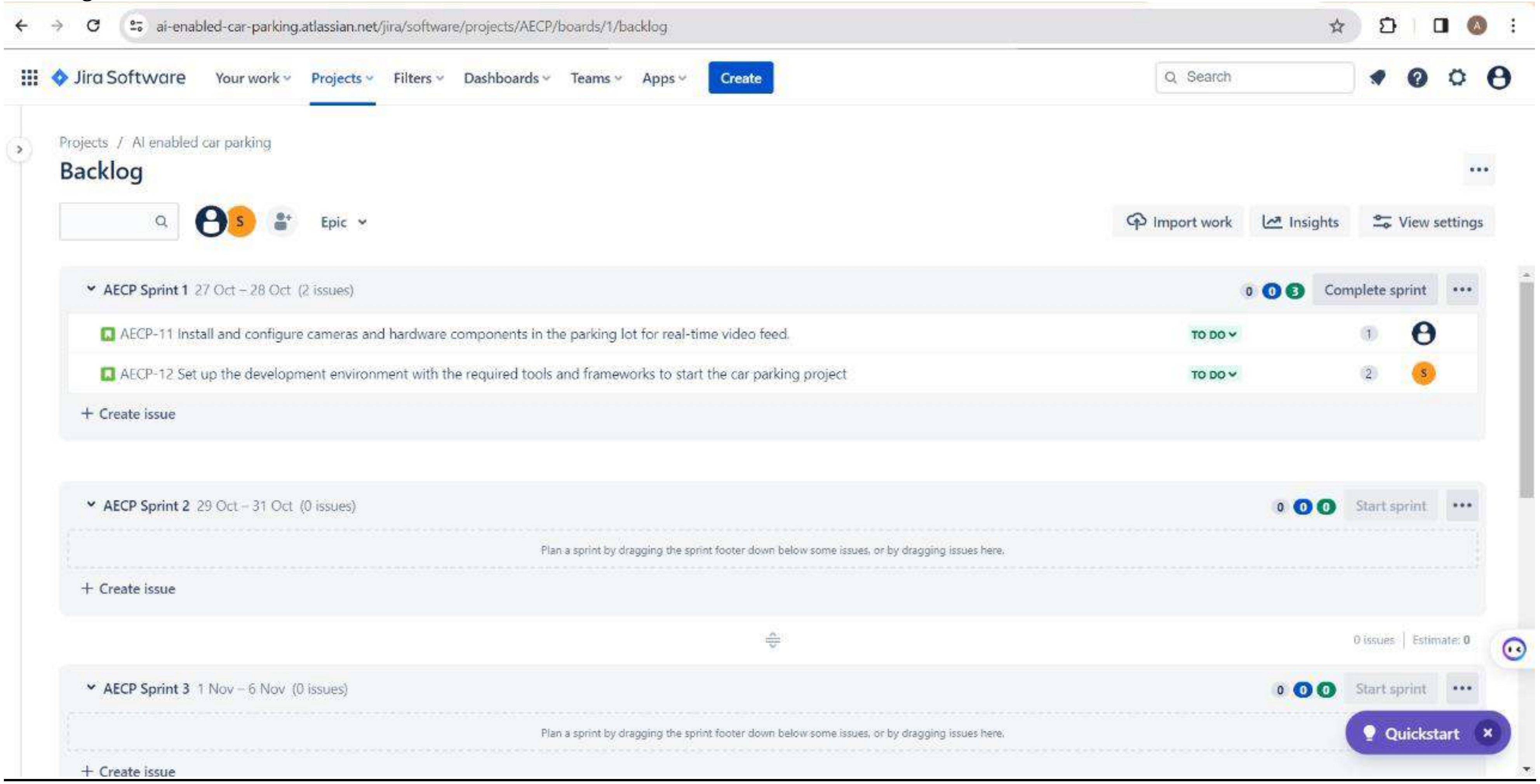
AECP Sprint 1

TO DO 2

Install and configure cameras and hardware components in the parking lot for real-time video feed.
AECP-11 1

Set up the development environment with the required tools and frameworks to start the car parking project
AECP-12 2

Backlog section



ai-enabled-car-parking.atlassian.net/jira/software/projects/AECP/boards/1/backlog

Projects / AI enabled car parking

Backlog

AECP Sprint 1 27 Oct – 28 Oct (2 issues) 0 0 3 Complete sprint

AECP-11 Install and configure cameras and hardware components in the parking lot for real-time video feed. TO DO 1

AECP-12 Set up the development environment with the required tools and frameworks to start the car parking project TO DO 2

+ Create issue

AECP Sprint 2 29 Oct – 31 Oct (0 issues) 0 0 0 Start sprint

Plan a sprint by dragging the sprint footer down below some issues, or by dragging issues here.

+ Create issue

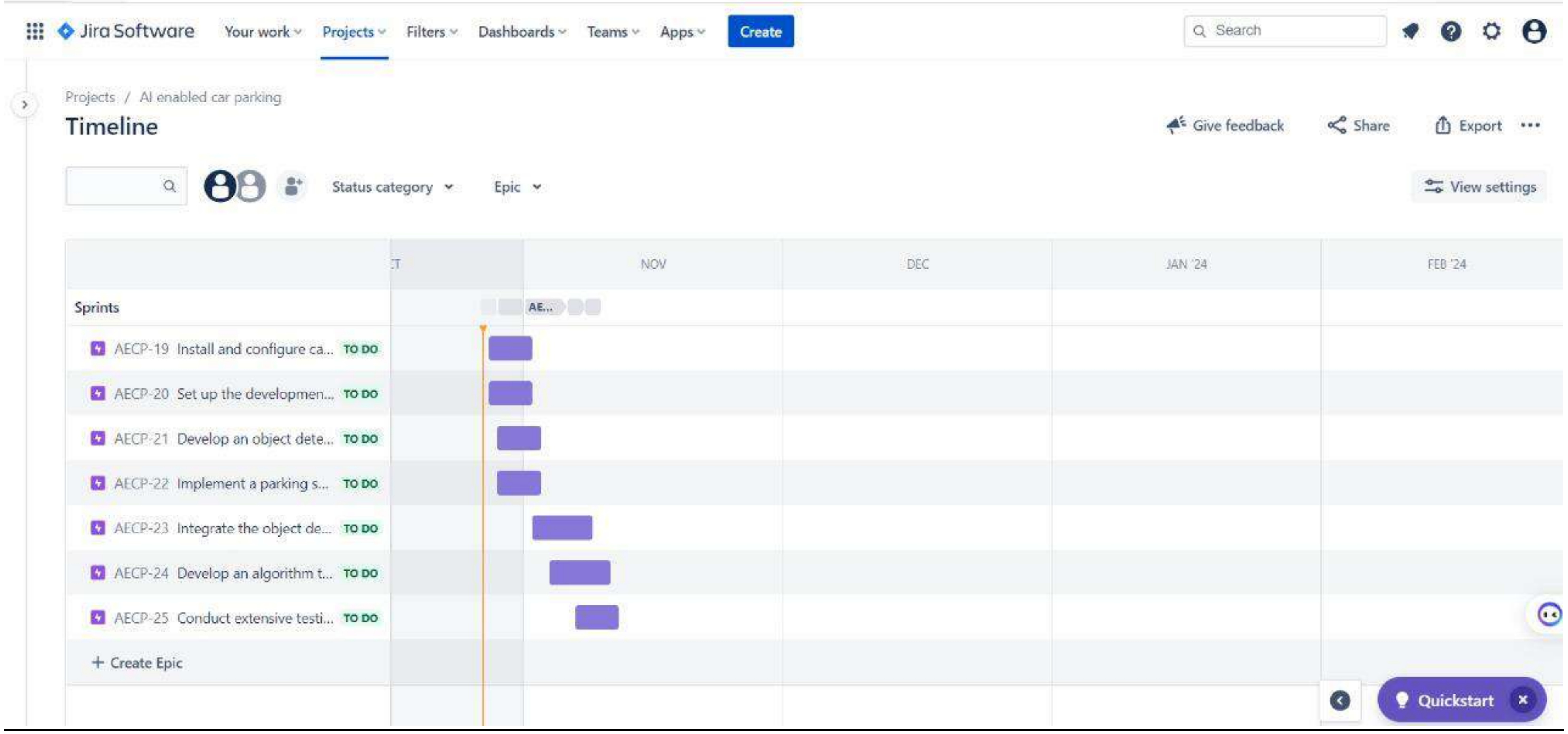
0 issues | Estimate: 0

AECP Sprint 3 1 Nov – 6 Nov (0 issues) 0 0 0 Start sprint

Plan a sprint by dragging the sprint footer down below some issues, or by dragging issues here.

+ Create issue

Timeline



Projects / AI enabled car parking

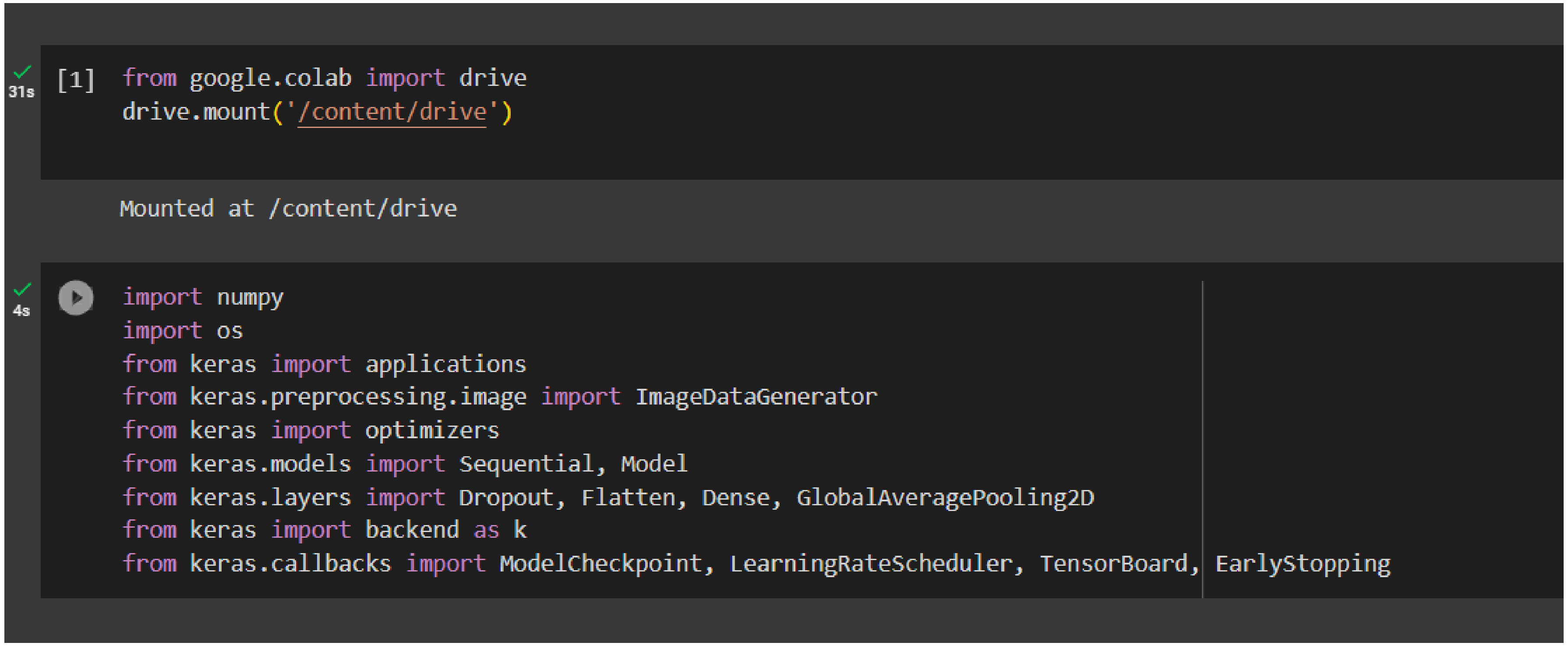
Timeline

Sprints

AECP-19 Install and configure ca...	TO DO
AECP-20 Set up the developmen...	TO DO
AECP-21 Develop an object dete...	TO DO
AECP-22 Implement a parking s...	TO DO
AECP-23 Integrate the object de...	TO DO
AECP-24 Develop an algorithm t...	TO DO
AECP-25 Conduct extensive testi...	TO DO

6. CODING & SOLUTIONING (Explain the features added in the project along with code)

The journey begins with the mounting of Google Drive, establishing a connection between our code and the vast storage capabilities of the cloud. From there, we delve into the intricacies of neural networks with the introduction of Keras, a high-level neural networks API, and the powerful VGG16 model, pre-trained on the vast ImageNet dataset.

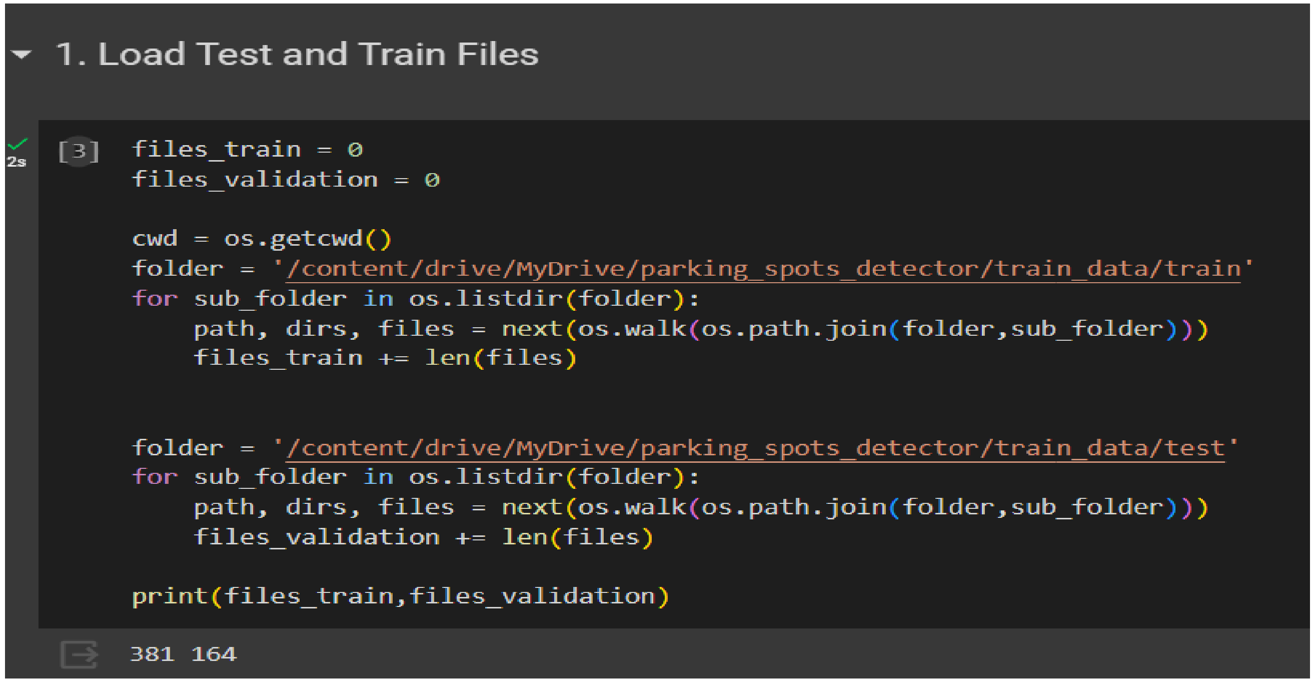


```
[1] from google.colab import drive
    drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
import numpy
import os
from keras import applications
from keras.preprocessing.image import ImageDataGenerator
from keras import optimizers
from keras.models import Sequential, Model
from keras.layers import Dropout, Flatten, Dense, GlobalAveragePooling2D
from keras import backend as k
from keras.callbacks import ModelCheckpoint, LearningRateScheduler, TensorBoard, EarlyStopping
```

As we navigate through the code, we'll notice the meticulous counting of training and validation files, a crucial step in preparing our model for learning. The VGG16 model, renowned for its prowess in image classification, forms the backbone of our system. However, rather than stopping there, we customize it to suit our specific task—detecting parking spots.

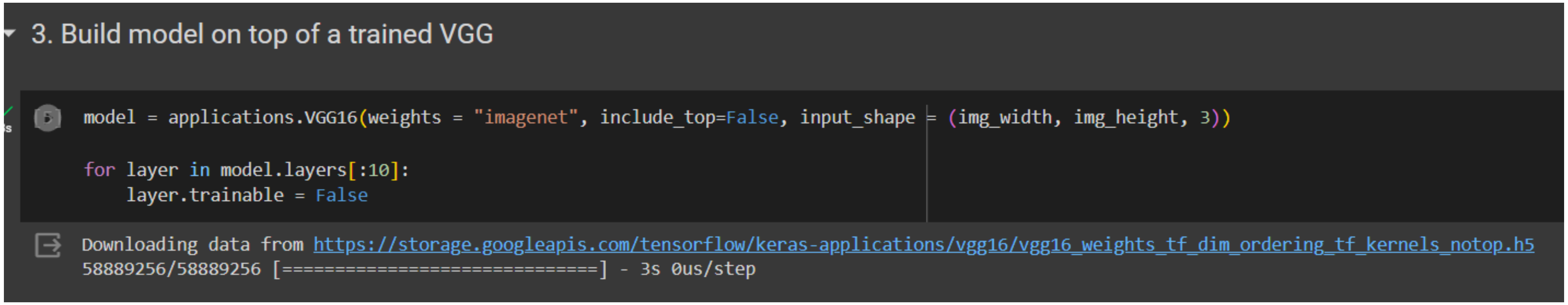


1. Load Test and Train Files

```
[3] files_train = 0
    files_validation = 0

    cwd = os.getcwd()
    folder = '/content/drive/MyDrive/parking_spots_detector/train_data/train'
    for sub_folder in os.listdir(folder):
        path, dirs, files = next(os.walk(os.path.join(folder,sub_folder)))
        files_train += len(files)


    folder = '/content/drive/MyDrive/parking_spots_detector/train_data/test'
    for sub_folder in os.listdir(folder):
        path, dirs, files = next(os.walk(os.path.join(folder,sub_folder)))
        files_validation += len(files)

    print(files_train,files_validation)
```

```
381 164
```

One notable aspect is the freezing of the initial layers. This ensures that the model capitalizes on the pre-learned features from ImageNet while adapting its later layers to our unique parking spot classification challenge. Custom layers, such as Flatten and Dense, are added to fine-tune the model's understanding of the dataset.

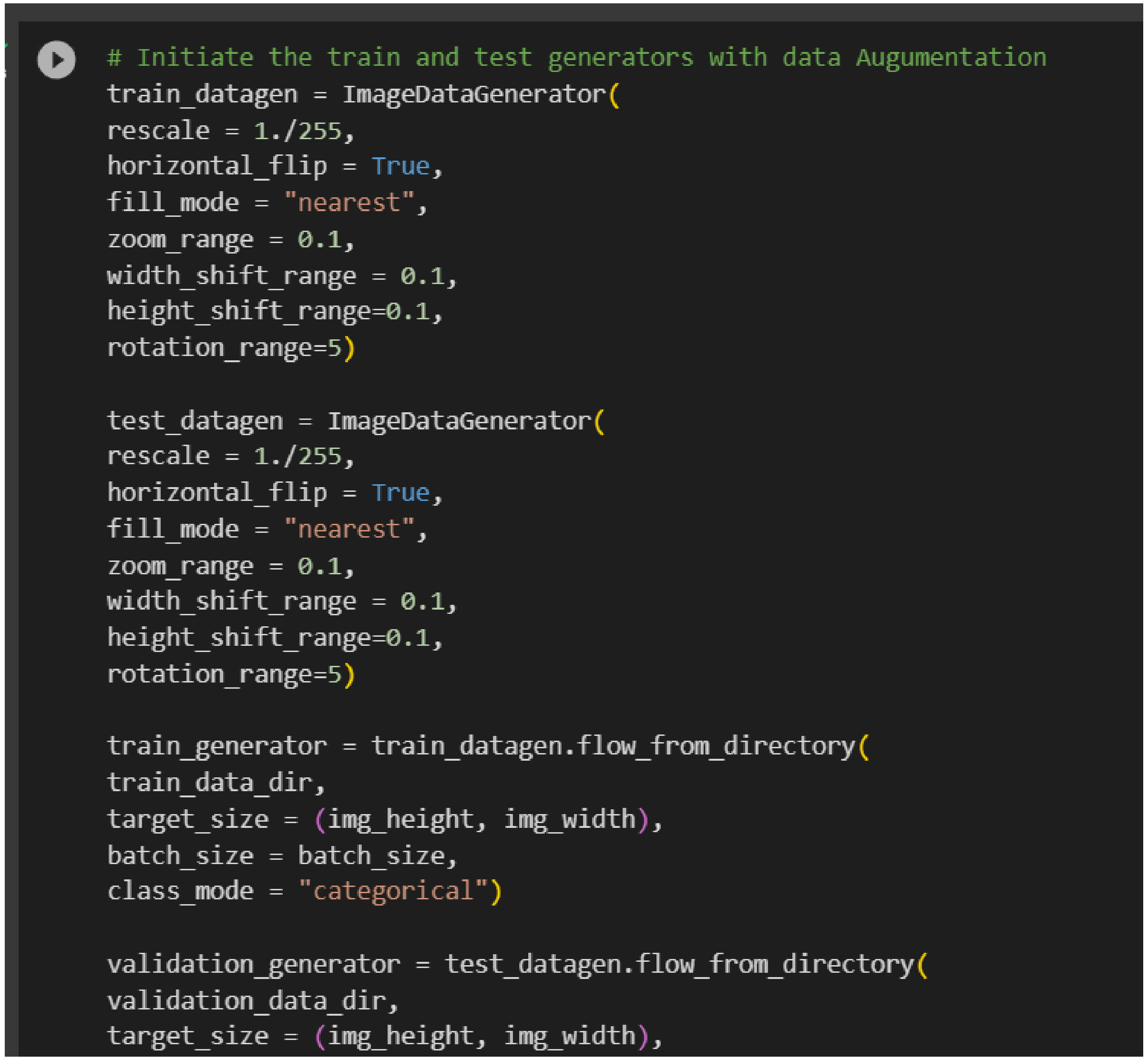


3. Build model on top of a trained VGG

```
model = applications.VGG16(weights = "imagenet", include_top=False, input_shape = (img_width, img_height, 3))

for layer in model.layers[:10]:
    layer.trainable = False
```

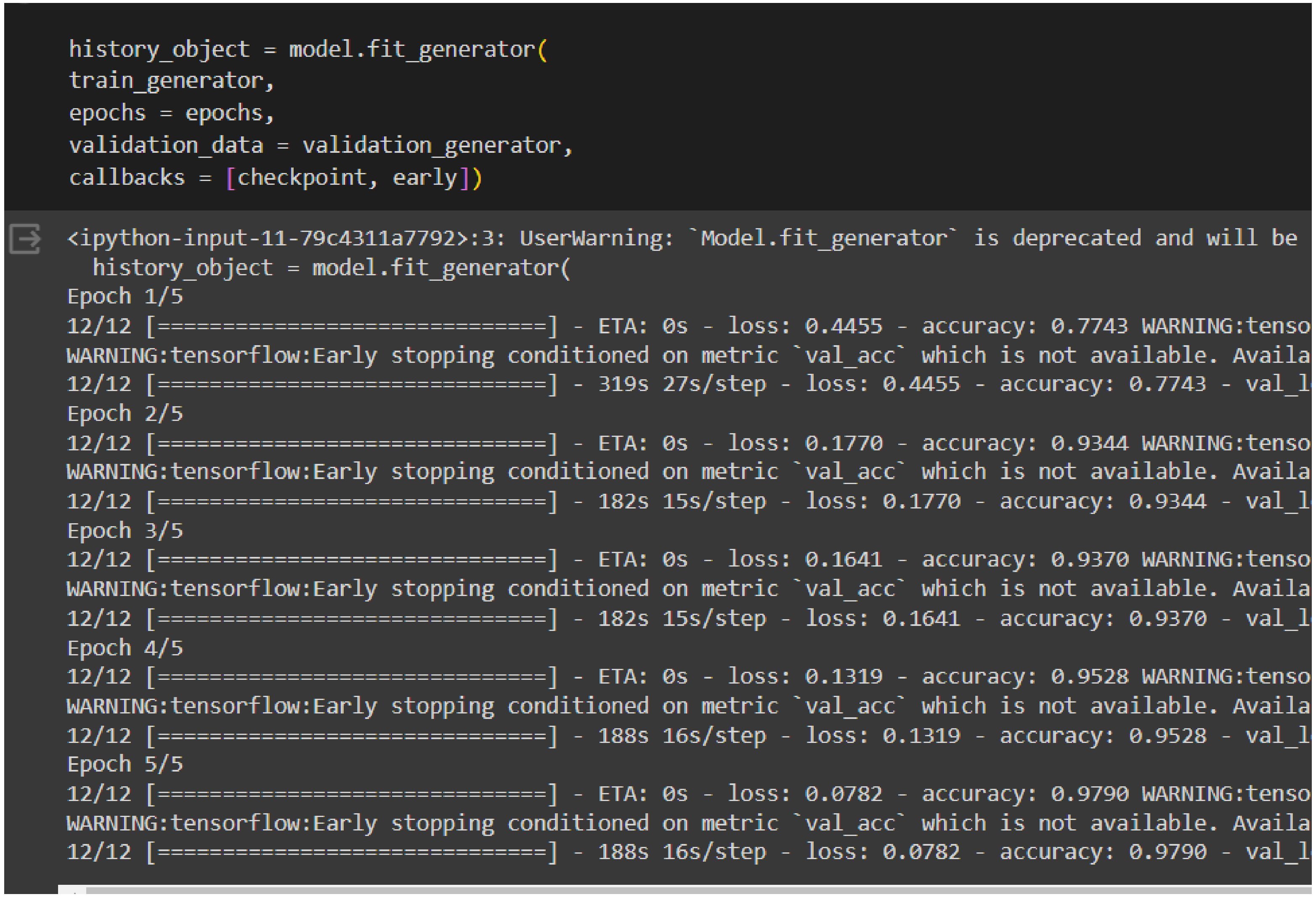
```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg16/vgg16_weights_tf_dim_ordering_tf_kernels_notop.h5
58889256/58889256 [==============================] - 3s 0us/step
```

Data augmentation, a technique vital for training robust models, is introduced. By applying transformations like flipping, zooming, and rotation to our images, we simulate diverse scenarios, enabling the model to generalize better to real-world situations.



```
# Initiate the train and test generators with data Augumentation
train_datagen = ImageDataGenerator(
rescale = 1./255,
horizontal_flip = True,
fill_mode = "nearest",
zoom_range = 0.1,
width_shift_range = 0.1,
height_shift_range=0.1,
rotation_range=5)

test_datagen = ImageDataGenerator(
rescale = 1./255,
horizontal_flip = True,
fill_mode = "nearest",
zoom_range = 0.1,
width_shift_range = 0.1,
height_shift_range=0.1,
rotation_range=5)

train_generator = train_datagen.flow_from_directory(
train_data_dir,
target_size = (img_height, img_width),
batch_size = batch_size,
class_mode = "categorical")

validation_generator = test_datagen.flow_from_directory(
validation_data_dir,
target_size = (img_height, img_width),
```

Our model's training is carefully monitored through epochs, and checkpoints are set up to save the best version of the model—akin to capturing its "Aha!" moments during the learning process. We witness the convergence of artificial and human intelligence as the model learns to distinguish parking spots with increasing accuracy over time.


```
history_object = model.fit_generator(
train_generator,
epochs = epochs,
validation_data = validation_generator,
callbacks = [checkpoint, early])
```

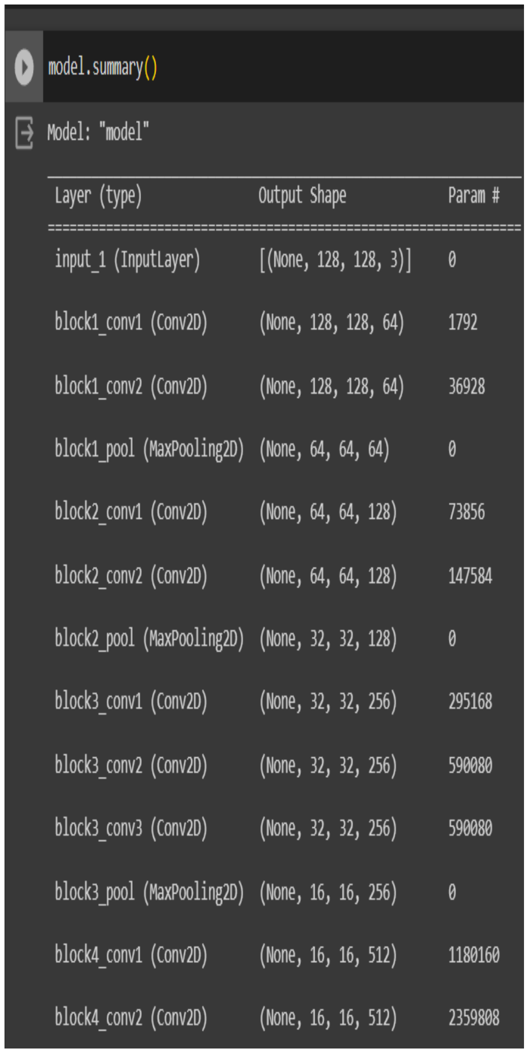
```
<ipython-input-11-79c4311a7792>:3: UserWarning: `Model.fit_generator` is deprecated and will be
  history_object = model.fit_generator(
Epoch 1/5
12/12 [==============================] - ETA: 0s - loss: 0.4455 - accuracy: 0.7743 WARNING:tenso
WARNING:tensorflow:Early stopping conditioned on metric `val_acc` which is not available. Availa
12/12 [==============================] - 319s 27s/step - loss: 0.4455 - accuracy: 0.7743 - val_l
Epoch 2/5
12/12 [==============================] - ETA: 0s - loss: 0.1770 - accuracy: 0.9344 WARNING:tenso
WARNING:tensorflow:Early stopping conditioned on metric `val_acc` which is not available. Availa
12/12 [==============================] - 182s 15s/step - loss: 0.1770 - accuracy: 0.9344 - val_l
Epoch 3/5
12/12 [==============================] - ETA: 0s - loss: 0.1641 - accuracy: 0.9370 WARNING:tenso
WARNING:tensorflow:Early stopping conditioned on metric `val_acc` which is not available. Availa
12/12 [==============================] - 182s 15s/step - loss: 0.1641 - accuracy: 0.9370 - val_l
Epoch 4/5
12/12 [==============================] - ETA: 0s - loss: 0.1319 - accuracy: 0.9528 WARNING:tenso
WARNING:tensorflow:Early stopping conditioned on metric `val_acc` which is not available. Availa
12/12 [==============================] - 188s 16s/step - loss: 0.1319 - accuracy: 0.9528 - val_l
Epoch 5/5
12/12 [==============================] - ETA: 0s - loss: 0.0782 - accuracy: 0.9790 WARNING:tenso
WARNING:tensorflow:Early stopping conditioned on metric `val_acc` which is not available. Availa
12/12 [==============================] - 188s 16s/step - loss: 0.0782 - accuracy: 0.9790 - val_l
```

Project Development Phase
Model Performance Test

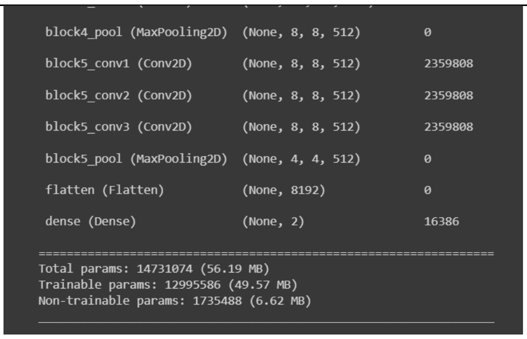
Date	9 November 2023
Team ID	Team-593009
Project Name	AI Enable car parking using OpenCV
Maximum Marks	10 Marks

Model Performance Testing:

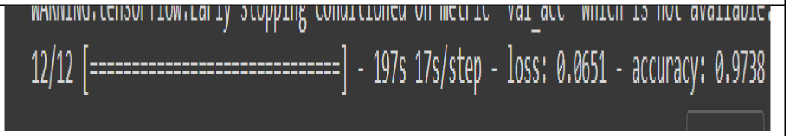
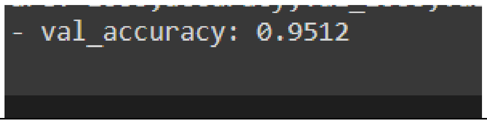
Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot
1.	Model Summary	Model: "model" ____ Layer (type) Output Shape Param # === input_1 (InputLayer) [(None, 128, 128, 3)] 0 block1_conv1 (Conv2D) (None, 128, 128, 64) 1792 block1_conv2 (Conv2D) (None, 128, 128, 64) 36928 block1_pool (MaxPooling2D) (None, 64, 64, 64) 0 block2_conv1 (Conv2D) (None, 64, 64, 128) 73856	

```
model.summary()

Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 128, 128, 3)]     0

 block1_conv1 (Conv2D)       (None, 128, 128, 64)      1792

 block1_conv2 (Conv2D)       (None, 128, 128, 64)      36928

 block1_pool (MaxPooling2D)  (None, 64, 64, 64)        0

 block2_conv1 (Conv2D)       (None, 64, 64, 128)       73856

 block2_conv2 (Conv2D)       (None, 64, 64, 128)       147584

 block2_pool (MaxPooling2D)  (None, 32, 32, 128)       0

 block3_conv1 (Conv2D)       (None, 32, 32, 256)       295168

 block3_conv2 (Conv2D)       (None, 32, 32, 256)       590080

 block3_conv3 (Conv2D)       (None, 32, 32, 256)       590080

 block3_pool (MaxPooling2D)  (None, 16, 16, 256)       0

 block4_conv1 (Conv2D)       (None, 16, 16, 512)       1180160

 block4_conv2 (Conv2D)       (None, 16, 16, 512)       2359808
```

		block2_conv2 (Conv2D) (None, 64, 64, 128) 147584 block2_pool (MaxPooling2D) (None, 32, 32, 128) 0 block3_conv1 (Conv2D) (None, 32, 32, 256) 295168 block3_conv2 (Conv2D) (None, 32, 32, 256) 590080 block3_conv3 (Conv2D) (None, 32, 32, 256) 590080 block3_pool (MaxPooling2D) (None, 16, 16, 256) 0 block4_conv1 (Conv2D) (None, 16, 16, 512) 1180160 block4_conv2 (Conv2D) (None, 16, 16, 512) 2359808 block4_conv3 (Conv2D) (None, 16, 16, 512) 2359808 block4_pool (MaxPooling2D) (None, 8, 8, 512) 0 block5_conv1 (Conv2D) (None, 8, 8, 512) 2359808	 block4_pool (MaxPooling2D) (None, 8, 8, 512) 0 block5_conv1 (Conv2D) (None, 8, 8, 512) 2359808 block5_conv2 (Conv2D) (None, 8, 8, 512) 2359808 block5_conv3 (Conv2D) (None, 8, 8, 512) 2359808 block5_pool (MaxPooling2D) (None, 4, 4, 512) 0 flatten (Flatten) (None, 8192) 0 dense (Dense) (None, 2) 16386 === Total params: 14731074 (56.19 MB) Trainable params: 12995586 (49.57 MB) Non-trainable params: 1735488 (6.62 MB) ____

		`block5_conv2`
		`(Conv2D)`
		`(None, 8, 8, 512)`
		`2359808`
		
		`block5_conv3`
		`(Conv2D)`
		`(None, 8, 8, 512)`
		`2359808`
		
		`block5_pool`
		`(MaxPooling2D)`
		`(None, 4, 4, 512)`
		`0`
		
		`flatten (Flatten)`
		`(None, 8192)`
		`0`
		
		`dense (Dense)`
		`(None, 2)`
		`16386`
		
		`===================`
		`===================`
		`===================`
		`========`
		`Total params:`
		`14731074 (56.19 MB)`
		`Trainable params:`
		`12995586 (49.57 MB)`
		`Non-trainable`
		`params: 1735488`
		`(6.62 MB)`	

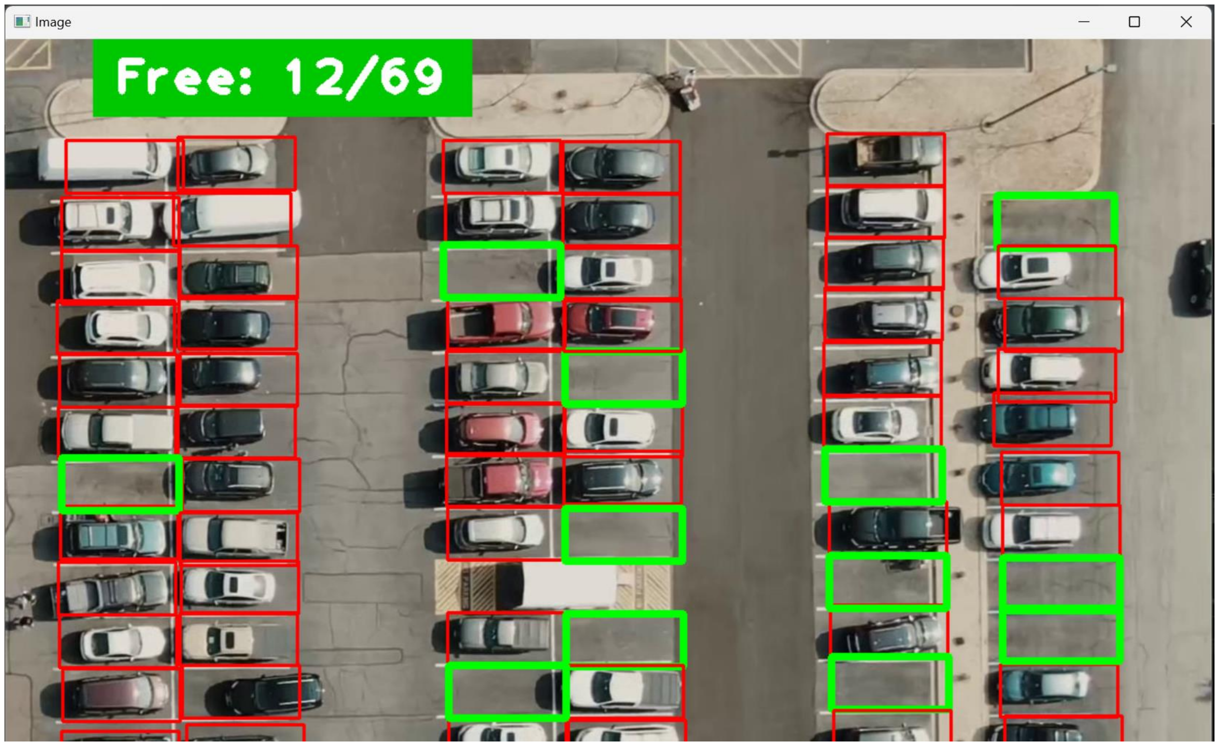
2.	Accuracy	Training Accuracy – 0.97
		
		Validation Accuracy -0.95
			
			`WARNING:tensorflow:Early stopping conditioned on metric val_acc which is not available.`
			`12/12 [==============================] - 197s 17s/step - loss: 0.0651 - accuracy: 0.9738`
			
			`- val_accuracy: 0.9512`

3.	Confidence Score (Only Yolo Projects)	Class Detected - Confidence Score -	

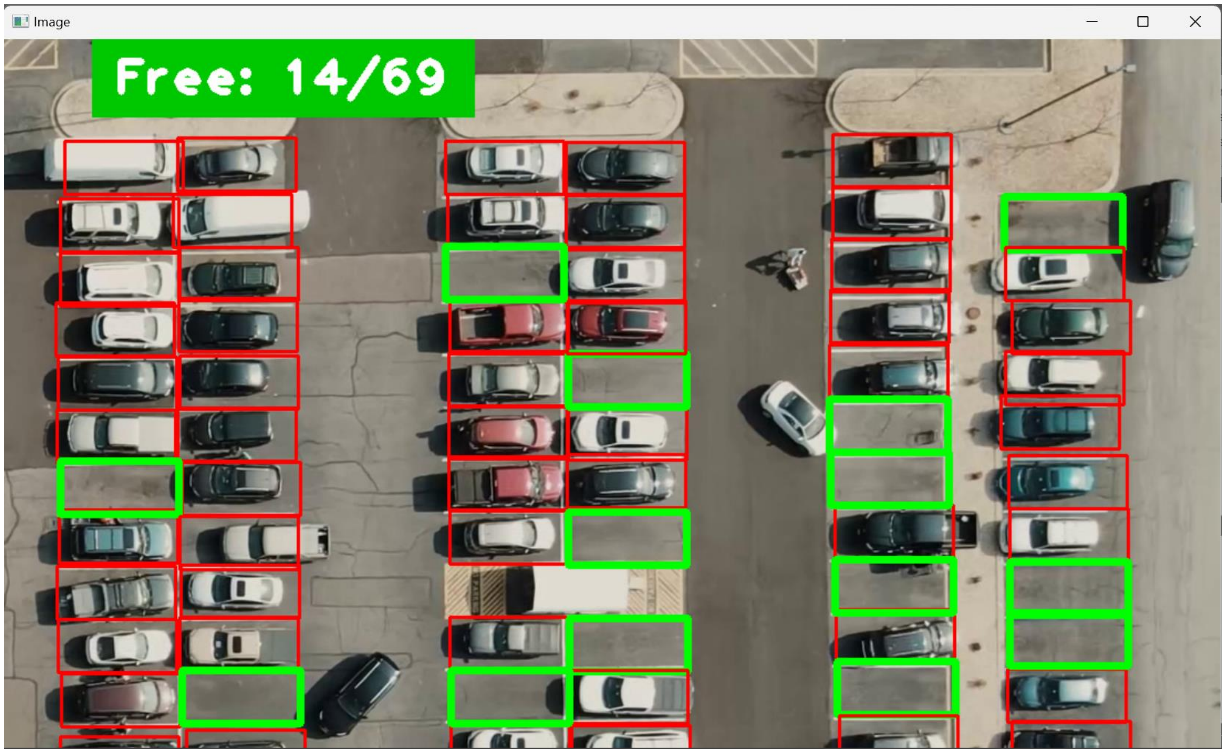
8. RESULTS

8.1 Output Screenshots:

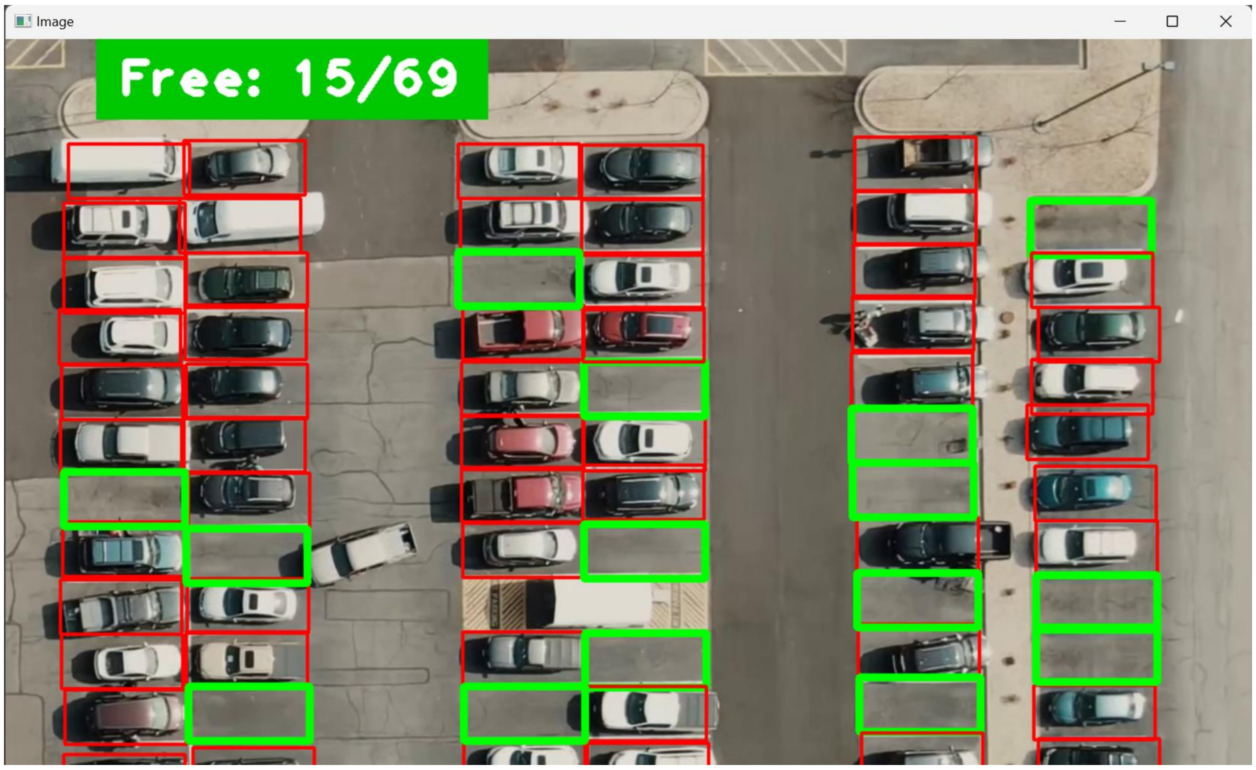
Screenshot 1:



Screenshot 2:



<u>Screenshot 3:</u>



9. ADVANTAGES & DISADVANTAGES

<u>Advantages</u>

1. Cost-Efficiency:

- **Moving Beyond Traditional Systems:**
    - The AI-enabled car parking system presents a cost-effective alternative to traditional radar and sensor-based solutions.
    - Eliminating the need for expensive hardware installations, the system relies on computer vision and AI, reducing initial setup costs.
    - Cost savings extend to maintenance, as the software-based approach minimizes wear and tear associated with physical sensors.

2. Efficient Parking Management:

- Automated parking space detection optimizes parking lot management.
- Real-time information aids drivers in finding available spaces quickly.

3. Reduced Traffic Congestion:

- Improved parking efficiency contributes to reduced traffic congestion.
- Minimizes the time spent searching for parking, leading to smoother traffic flow.

4. Environmental Impact:

- Reduced fuel consumption and emissions due to minimized search time.
- Contributes to environmental sustainability.

<u>Disadvantages</u>

1. Dependency on Technology:

- Relies on accurate functioning of computer vision and AI systems.
- Technical malfunctions may affect reliability.

2. Implementation Costs:

- Initial setup costs for implementing the system may be high.
- Maintenance and upgrades can add to the expenses.

3. Privacy Concerns:

- Collection of data for parking management may raise privacy issues.
- Proper measures must be in place to address privacy concerns.

10. CONCLUSION

In conclusion, the AI-enabled car parking system presents a promising solution for efficient urban parking management. The advantages, including improved efficiency, reduced traffic congestion, and environmental benefits, showcase the positive impact of the technology. However, considerations such as dependency on technology, implementation costs, and privacy concerns need to be addressed. The system's success lies in striking a balance between technological innovation and addressing associated challenges.

11. FUTURE SCOPE

The future scope of the AI-enabled car parking system involves continuous advancements and enhancements. Potential areas for future development include:

1. Integration with Smart Cities:

- ❖ Collaborate with smart city initiatives for seamless integration into urban infrastructure.

2. Enhanced AI Capabilities:

- ❖ Improve AI algorithms for even more accurate parking space detection.
- ❖ Explore machine learning techniques for continuous improvement.

3. User-Friendly Interfaces:

- ❖ Develop intuitive mobile applications for users to easily access parking information.
- ❖ Implement user feedback mechanisms for system optimization.

4. Security and Privacy Measures:

- ❖ Invest in robust security measures to protect data and address privacy concerns.
- ❖ Regularly update protocols to stay ahead of potential security threats.

5. Global Adoption:

- ❖ Promote the adoption of AI-enabled parking systems globally.
- ❖ Collaborate with municipalities and private entities for widespread implementation.

12. APPENDIX

Source Code:

https://drive.google.com/drive/folders/1eyiHhs5oGbC99Pc3fANYAlWqimvooeCM?usp=sharing

GitHub Repository Link:

https://github.com/smartinternz02/SI-GuidedProject-600612-1697638708

Project Demo Link
https://youtu.be/Y6_9AYtZ9dM

Thank you!