Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	18-05-2023
Team ID	NM2023TMID14621
Project Name	AI enabled car parking using open CV
Maximum Marks	4 Marks

Technical Architecture:

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

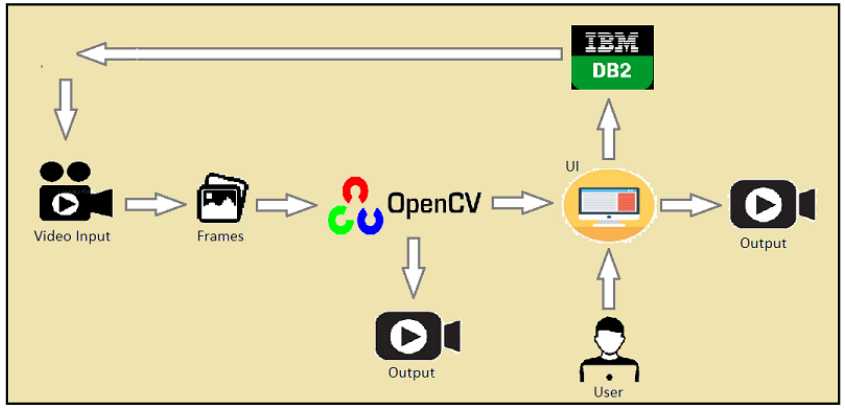

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	The interface through which users interact with the system	Web UI: HTML, CSS, JavaScript; Mobile App: Native or Hybrid frameworks; Chatbot: Natural Language Processing (NLP) platforms like Dialogflow, IBM Watson Assistant, etc.
2.	Object Detection	Identifying and tracking vehicles in the camera feed	OpenCV, Deep learning models (YOLO, SSD, Faster R-CNN, etc.
3.	Parking Space Detection	Detecting available parking spaces in the video feed	OpenCV, Image processing algorithms, Contour detection.
4.	Parking Guidance	Real-time guidance for drivers on available parking spots	LED displays, Mobile App (with map integration), Voice instructions, etc.
5.	Automated Parking	Automated steering and movement control for parking	Integration with vehicle control systems, Ultrasonic sensors, LiDAR, etc.
6.	Monitoring and Security	Surveillance and security features for the parking area	CCTV cameras, License plate recognition systems, Video analytics, etc.
7.	Database	Storage of system data	Relational databases (MySQL, PostgreSQL), NoSQL databases (MongoDB, Cassandra), etc.
8.	Cloud Infrastructure	Deployment of the system on cloud platforms	Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform (GCP), etc.
9.	Machine Learning Models	Models for object recognition, anomaly detection, etc.	Deep learning frameworks (TensorFlow, PyTorch), Custom-trained models, Pre-trained models (ResNet, MobileNet, etc.

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	OpenCV, TensorFlow, PyTorch, Flask, Django, Node.js, etc.
2.	Security Implementations	List all the security / access controls implemented, use of firewalls etc.	Encryption algorithms (AES, RSA), SSL/TLS, Hash functions (SHA-256), Firewall configurations, Identity and Access.
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-services)	Cloud platforms (AWS, Azure, GCP), Containerization (Docker, Kubernetes), Serverless architecture, Load balancing technologies (Nginx, HAProxy), Database sharding or replication, Horizontal scaling, etc.
4.	Availability	Justify the availability of application (e.g. use of load balancers, distributed servers etc.)	Load balancers (Nginx, HAProxy), Distributed server architecture, High availability configurations (redundancy, failover mechanisms), Cloud platforms (AWS, Azure, GCP), Auto-scaling, Monitoring and alerting systems (Prometheus, Nagios), etc.
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	Caching mechanisms (Redis, Memcached), Content Delivery Networks (CDNs), Optimized algorithms and data structures, Asynchronous processing, Efficient database queries and indexing, Performance testing and optimization tools (Apache JMeter, Locust), etc.