***Smart Parking Using IoT***

**INTRODUCTION:**

**1.1 Overview**

**Everyone who has ever been frustrated driving around urban areas in search of parking has wished for a solution that could quickly lead them to that elusive spot. In a recent research it has been found that a driver takes nearly 8 minutes to park his vehicle because he spend more time in searching the parking lot. This searching leads to 30 to 40% of traffic congestion. This concern attracted strategic investments from dedicated industry sectors to boost parking revenues through technology-enabled solutions. Parking industry is being revolutionized by new technologies that enable cities to reduce traffic congestion and carbon emission. The Internet of Things (IoT) permeates with the world of parking to streamline processes that deliver intelligent parking solutions, which extend and manage parking inventories. In this context, IoT uses embedded wireless sensor networks to connect physical parking space infrastructures with information and communication technologies, where cloud-based smart management services are provided. This interconnectivity shift is also driving socio-economical changes, where data unleashed from physical infrastructures is leading to productivity gains through new applications and new business models.**

**1.2 Purpose**

**Smart parking development implies an IoT-based system that sends data about free and occupied parking slots via web/mobile application. The main moto of this creation of smart parking using the Internet Of Things and random sensor values where available parking places could be displayed in a web application.**

**LITERATURE SURVEY:**

**2.1 Existing problem**

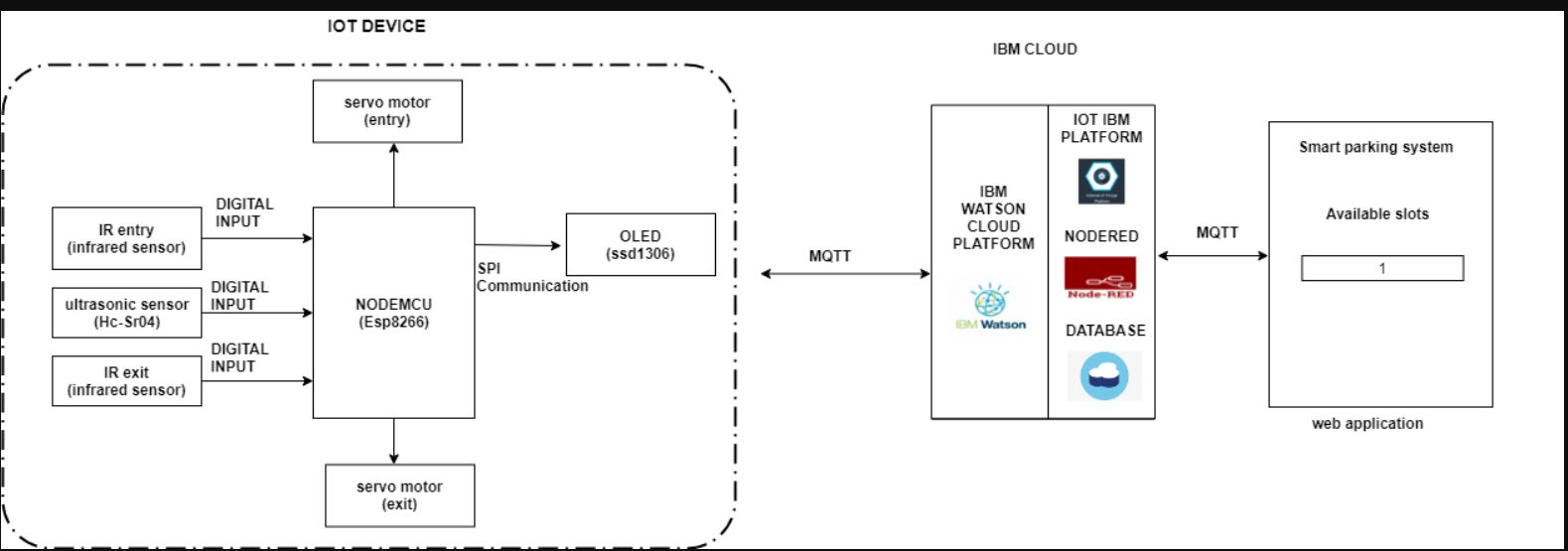
**Insufficient parking space in heavily populated areas. Inefficient use of existing parking capacity. Struggle to find an open parking slot in a large parking space Confusion in finding your vehicle in a large parking lot.**

**2.2 Proposed solution**

**Minimum Parking Requirements Increase On-Street (Curb) Parking. Subsidize Off-street Parking. Add Remote Parking Spaces. Redesign Existing Parking Facilities. Car Stackers and Mechanical Garages. Provide Parking Information to Users. To find the parking space from any where by using the mobile application.**

**THEORITICAL ANALYSIS:**

**3.1 Block diagram**

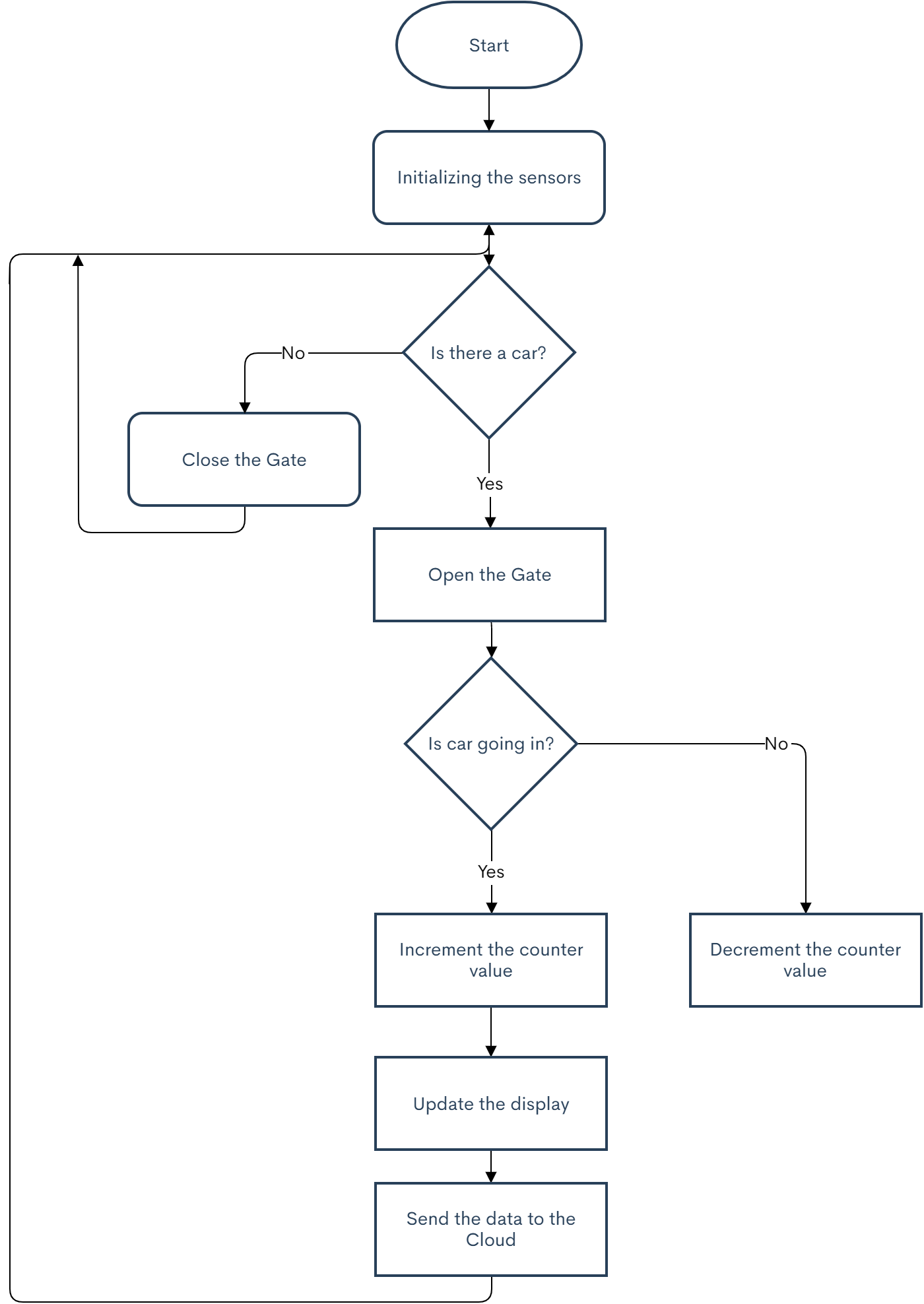


**3.2 Software design**





**FLOWCHART:**



**RESULT:**

**The project “IOT based Smart Parking system” was designed such that the status of parking slots can be known from anywhere in the users webpage. This is achieved using Wi-Fi communication.**

**ADVANTAGES:**

* **Shorter waiting time at parking place**
* **It saves fuel, money, space and time.**
* **Reduced pollution.**
* **Reduced traffic.**
* **Carbon emission is reduced.**
* **Efficiency.**

**DISADVANTAGES:**

* **Requires staff to administrate.**
* **Inappropriate for use at smaller scales.**
* **Hard to repair the underground sensors.**
* **Use of redundant systems will result in a greater cost.**
* **It may be a bit confusing for unfamiliar users.**

**APPLICATIONS:**

**The smart car parking system can be implemented in**

* **Shopping malls**
* **Restaurants**
* **Theatres**

**CONCLUSION:**

* **This project focuses on implementation of car parking place detection using Internet of Things.**
* **The system benefits of smart parking to do well by avoiding time wasting.**
* **Developing a smart parking solutions with in a city solves the pollution problem.**

**FUTURE SCOPE:**

**The future of the smart parking system is expected to be**

**significantly influenced by the arrival of automated vehicles(AVs).**

**Several cities around the world are already beginning to trial self-parking vehicles, specialized AV parking Iots, and robotic parking valets.**

**BIBLOGRAPHY:**

* **I.V.VAIBHAV, A.Ramya, A Review on Smart Parking Management System Using Vehicle Authentication, IJAREEIE 2016.**
* **Chi-Hung Chuang, Luo-Wei Tsai, “Vehicle License plate recognition using super resolution technique”, 2014 11th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS).**
* **Mingkai Chen, “A Parking Guidance and Information System based on Wireless Sensor Network”, IEEE International Conference on Information and Automation Shenzhen, China June 2011.**
* **Pahang, “Development of an Automatic Parallel Parking System for Nonholonomic Mobile Robot”, International Conference on Electrical, Control and Computer Engineering Pahang, Malaysia, June 21-22, 2011.**
* **Huang Cai-mei, He Zhi-kun, “Design of Reverse Search Car System for Large Parking Lot Based on NFC Technology”, 2014 IEEE.**
* **Bhosale Swapnali B, Kayastha Vijay S, “Feature extraction using surf algorithm for object recognition”, International Journal of Technical Research and Applications.**
* **Face recognition using principal component analysis and neural networks, at: http: //www.researchgate.net /publication/23595016.**
* **W.S. Tang, Yuan Zheng, “An Intelligent Car Park Management System based on Wireless Sensor Networks”, 2009 IEEE.**
* **Giuliano Benelli, Alessandro Pozzebon, “An Automated Payment System for Car Parks Based on Near Field Communication Technology”, University of Siena, Italy be nelli, alessandro.pozzebon.**
* **Abhirup Khanna “IoT based Smart Parking System” University of Petroleum and Energy Studies (UPES) Dehradun, Uttarakhand, 2016**

**APPENDIX:**

**The growth of Internet of Things have given rise to New possibilities in terms of smart cities. Smart parking facilities and traffic management systems have always been at the core of constructing smart cities. In this project, we address the issue of parking and present an IoT based web application smart parking system. The system that we propose provides real time information regarding availability of parking slots in a parking area. Users from any locations could book a parking slot for them by the use of our web application. The efforts made in this project are indented to improve the parking facilities of a city and thereby aiming to enhance the quality of life of its people. In our system user can view the real view of parking slot of nay register buildings, mall, hospitals, colleges and may more public parking areas. Due to this user has a choice that in what transport system he should use to visit that place**

**Source code:**

**# -\*- coding: utf-8 -\*-**

**import datetime**

**import ibm\_boto3**

**from ibm\_botocore.client import Config, ClientError**

**import cv2**

**import numpy as np**

**import sys**

**import ibmiotf.application**

**import ibmiotf.device**

**import random**

**import time**

**from cloudant.client import Cloudant**

**from cloudant.error import CloudantException**

**from cloudant.result import Result, ResultByKey**

**#Provide your IBM Watson Device Credentials**

**organization = "yj35v1"**

**deviceType = "project"**

**deviceId = "7777"**

**authMethod = "token"**

**authToken = "123456789"**

**def myCommandCallback(cmd):**

**print("Command received: %s" % cmd.data)**

**print(cmd.data['command'])**

**if(cmd.data['command']=="open"):**

**print("gate open")**

**if(cmd.data['command']=="close"):**

**print("gate close")**

**if(cmd.data['command']=="entry"):**

**print("entry")**

**if(cmd.data['command']=="exit"):**

**print("exit")**

**if(cmd.data['command']=="filled slots"):**

**print("filled slots")**

**if(cmd.data['command']=="empty slots"):**

**print("empty slots")**

**try:**

**deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken}**

**deviceCli = ibmiotf.device.Client(deviceOptions)**

**#..............................................**

**except Exception as e:**

**print("Caught exception connecting device: %s" % str(e))**

**sys.exit()**

**# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting" 10 times**

**deviceCli.connect()**

**car\_classifier=cv2.CascadeClassifier("cars.xml")**

**#It will read the first frame/image of the video**

**video=cv2.VideoCapture('cars2.mp4')**

**COS\_ENDPOINT = "https://s3.jp-tok.cloud-object-storage.appdomain.cloud" # Current list avaiable at https://control.cloud-object-storage.cloud.ibm.com/v2/endpoints**

**COS\_API\_KEY\_ID = "tD4oM6k1dLctb5ZW6yr103quV\_5hpuD18XQGw6D1zPYx" # eg "W00YiRnLW4a3fTjMB-oiB-2ySfTrFBIQQWanc--P3byk"**

**COS\_AUTH\_ENDPOINT = "https://iam.cloud.ibm.com/identity/token"**

**COS\_RESOURCE\_CRN = "crn:v1:bluemix:public:cloud-object-storage:global:a/3f309106247d4e799cc270a23f1bde40:1fb7171d-ba67-4ee9-9b7c-b767eab3a7f4::"**

**client = Cloudant("d613fe38-fcff-4aa6-af10-9bbad53806c9-bluemix", "8f01736ba3c010370fe6a4e6c4f851b6c2da83ef4bf65412aa292019e3c5e72b", url="https://d613fe38-fcff-4aa6-af10-9bbad53806c9-bluemix:8f01736ba3c010370fe6a4e6c4f851b6c2da83ef4bf65412aa292019e3c5e72b@d613fe38-fcff-4aa6-af10-9bbad53806c9-bluemix.cloudantnosqldb.appdomain.cloud")**

**client.connect()**

**database\_name = "doorbell"**

**# Create resource**

**cos = ibm\_boto3.resource("s3",**

**ibm\_api\_key\_id=COS\_API\_KEY\_ID,**

**ibm\_service\_instance\_id=COS\_RESOURCE\_CRN,**

**ibm\_auth\_endpoint=COS\_AUTH\_ENDPOINT,**

**config=Config(signature\_version="oauth"),**

**endpoint\_url=COS\_ENDPOINT**

**)**

**def multi\_part\_upload(bucket\_name, item\_name, file\_path):**

**try:**

**print("Starting file transfer for {0} to bucket: {1}\n".format(item\_name, bucket\_name))**

**# set 5 MB chunks**

**part\_size = 1024 \* 1024 \* 5**

**# set threadhold to 15 MB**

**file\_threshold = 1024 \* 1024 \* 15**

**# set the transfer threshold and chunk size**

**transfer\_config = ibm\_boto3.s3.transfer.TransferConfig(**

**multipart\_threshold=file\_threshold,**

**multipart\_chunksize=part\_size**

**)**

**# the upload\_fileobj method will automatically execute a multi-part upload**

**# in 5 MB chunks for all files over 15 MB**

**with open(file\_path, "rb") as file\_data:**

**cos.Object(bucket\_name, item\_name).upload\_fileobj(**

**Fileobj=file\_data,**

**Config=transfer\_config**

**)**

**print("Transfer for {0} Complete!\n".format(item\_name))**

**except ClientError as be:**

**print("CLIENT ERROR: {0}\n".format(be))**

**except Exception as e:**

**print("Unable to complete multi-part upload: {0}".format(e))**

**while True:**

**#capture the first frame**

**check,frame=video.read()**

**gray=cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)**

**#detect the cars from the video using detectMultiScale function**

**cars=car\_classifier.detectMultiScale(gray,1.1,1)**

**#drawing rectangle boundries for the detected car**

**for(x,y,w,h) in cars:**

**cv2.rectangle(frame, (x,y), (x+w,y+h), (127,0,255), 2)**

**cv2.imshow('cars detection', frame)**

**picname=datetime.datetime.now().strftime("%y-%m-%d-%H-%M")**

**picname=picname+".jpg"**

**pic=datetime.datetime.now().strftime("%y-%m-%d-%H-%M")**

**cv2.imwrite(picname,frame)**

**car=5**

**my\_database = client.create\_database(database\_name)**

**multi\_part\_upload("cloud-object-storage-dsx-cos-standard-jbp",picname,pic+".jpg")**

**if my\_database.exists():**

**print("'{database\_name}' successfully created.")**

**json\_document = {**

**"\_id": pic,**

**"link":COS\_ENDPOINT+"/cloud-object-storage-dsx-cos-standard-jbp/"+picname**

**}**

**new\_document = my\_database.create\_document(json\_document)**

**if new\_document.exists():**

**print("Document '{new\_document}' successfully created.")**

**time.sleep(1)**

**f=34**

**e=45**

**data = {"d":{ 'filledslots' : f, 'emptyslots': e, 'car': car}}**

**#print data**

**def myOnPublishCallback():**

**print ("Published data to IBM Watson")**

**success = deviceCli.publishEvent("Data", "json", data, qos=0, on\_publish=myOnPublishCallback)**

**if not success:**

**print("Not connected to IoTF")**

**time.sleep(1)**

**deviceCli.commandCallback = myCommandCallback**

**car=0**

**#waitKey(1)- for every 1 millisecond new frame will be captured**

**Key=cv2.waitKey(1)**

**if Key==ord('q'):**

**#release the camera**

**video.release()**

**#destroy all windows**

**cv2.destroyAllWindows()**

**break**

**deviceCli.disconnect()**