INTERNET OF THINGS -GROUP 4

SMART PARKING

PHASE-3

STUDENT NAME : AKILRAJ.R

COLLEGE CODE : 8226

COLLEGE NAME : ARIFA INSTITUTE OF TECHNOLOGY, NAGAPATTINAM

NM ID : au822621106005

REGISTER NUMBER: 822621106005

EMAIL ID : akilrajakilraj0@gmail.com

RELATED WORKS :

- The sensors used in IoT based smart parking systemstores and accesses data from remote locations with the help of the cloud these factors give raise to cloud of things (COT). The nodes could be monitored and controlled from any location the system that we propose provides information regarding the availability of the parking slots with the help of the mobile application the users from the remote location can book the parking slots.
- An algorithm is used to increase efficiency of cloud-based parking system and network architecture technology is used. This algorithm is used to find the lowest cost parking space. Considering the number of parking space available and also considering the distance of the parking space from the user. The user can directly access the cloud-based server and find the information on the parking space. The user can also install an application in their mobile phones to access this information. With the help of this algorithm, waiting time of the user to find a parking space can be minimised. Security aspects are not included in this paper.
- A wireless sensor node along with smart phone application is being used to find the parking space.Since, wireless technology is used here the system has high accuracy and efficiency. In this system, onboard units are used to communicate with other vehicles. The user parks his vehicle in any one of the several bays available a mechanical lift lifts the vehicle out.ticket key and id are given to the user and it is only known to the user which is used to retrieve the vehicle. The user need not carry any paper ticket since anRfid card is given to the user. The technology used here is economical. Security features must be improved to protect the user’s privacy.

- The author of smart parking system the survey has divided detector system and vehicle sensors into two math categories as intrusive sensors and non - intrusive sensors. Intrusive sensors are installed in holes on the road surface by tunnelling under the road. Nonintrusive sensors do not affect the surface of the road and it can be easily installed and maintained. Smart parking system helps us to resolve the grounding problems of the traffic congestion and it also reduces the emission from a car.
- A paper proposes efficient way to unfold the issue of parking availability in the real time scenario and to reduce the time consumption. In this, the data is sent locally with devices which filters the data. This signal is transmitted over the cloud for the process as well as for evaluation which uses machine learning algorithms. This paper uses mobile phone application that connects the user with the real time traffic status via Google API. Thus ,avoiding traffic congestion. This paper does not provide the reservation facility for the car parking.
- Smart parking using IoT technology helps to designs and develops a real smart parking system which provides information for vacant spaces and also helps the user to locate the nearest availability. This paper uses a computer vision to detect vehicle number plate in order to enhance the security. The user can pay for the parking space prior to the entry of the car through mobile payment. Thus, insuring the reservation of the parking. The user is notified about the parking location, number of slots available and all other relevant information. The paper uses efficient algorithms and techniques for extracting license plate text. An algorithm operates on the ultrasonic sensor detection of the vehicle entering into the parking slot and calculates the minimum cost for the user.
- Smart parking system based on reservation allows the reservation of a vacant space which involves smart parking system based on reservation (SPSR).This consists of host parking database

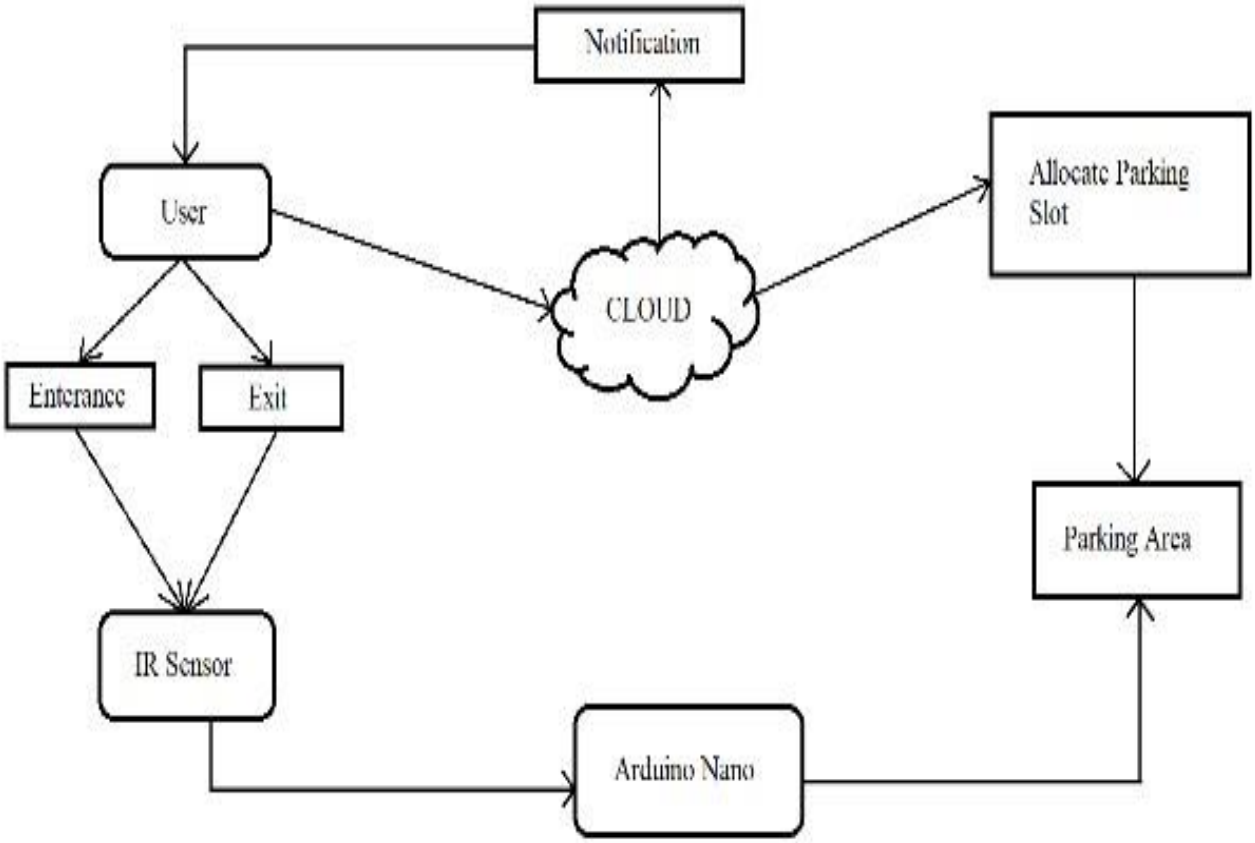
management which collects and stores data about the driver's identity and parking location. When the parking reservation time is about to expire a notification will be sent to the user through the web service that has been provided to the user by the admin. The main drawback is that some other user can occupy a reserved parking space to avoid this QR scanners are used to identify the user.

- It helps us to propose a way in which the user can reserve his parking space by mentioning the destination and the vehicle type with the help of mobile applications .The booking details will be stored in the cloud which finds the shortest path from the user to the parking space , the location of the user is updated regularly in the cloud with the help of GPS . When the user reaches the car parking the Rfid is scanned and the user is allowed into the parking space.The billing is done by the cloud server. The main disadvantage is that the car parking space must be registered in the smart parking system for the user to use it.
- This paper describes the implementation of wireless sensor networks (WSN) used in a car parking system with the help of a server which is using xbeezigbee. The car parking system can detect the car which is parked in the parking slot. The aim of this project is to make it cost effective and user friendly. Car parking system helps the user to sustain the data with 90% of accuracy.
- Smart car parking system provides a comprehensive parking solution for the user as well as admin of the parking area. It provides the feature for a reserved parking slot and identify reserved user. In this, user can navigate to the nearest parking area depending upon the size of the vehicle. The user can reserve parking slot based on hourly, daily, weekly or monthly basis. An algorithm is designed to identify the nearest parking according to the size. The mobile application provided to the user is used to reserve and pay-as-you go service.

SYSTEM ARCHITECTURE

A. Proposed System

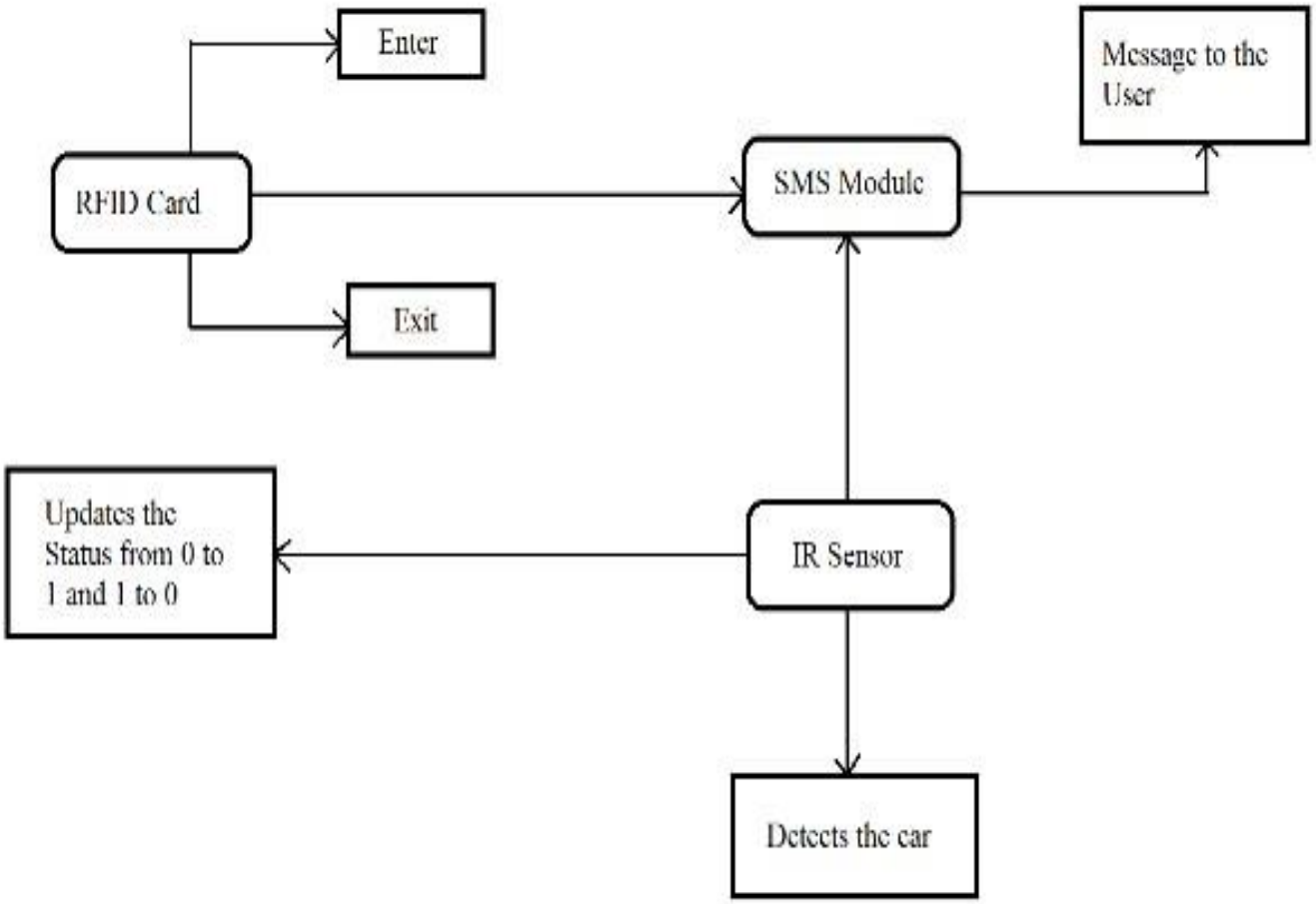
It consists of three sections: first section is the parking area which includes Arduino devices along with IR sensor. The user interacts with the parking area with the help of these devices. The user cannot enter the parking area without the help of RFID card. The second section contains the cloud-based web services which acts a mediator between the user and parking area. The cloud is updated depending upon the availability of the parking area. The admin administers the cloud services and it can also be viewed by the user for checking the availability. The third section is the user side. The user gets notification on the basis of the availability via SMS through GSM module.



B. Hardware

The three main hardware components used are GMS module, RFID card, IR sensors. A user is allowed inside a parking space only if the user has a RFID card. RFID card contains the information of the registered user. As the car enters the parking slot, reader module scans the registered user's RFID tag. The data is sent to the ardunio for checking the availability of the car parking and simultaneously, the user is notified through SMS about the status of the parking area. The GSM

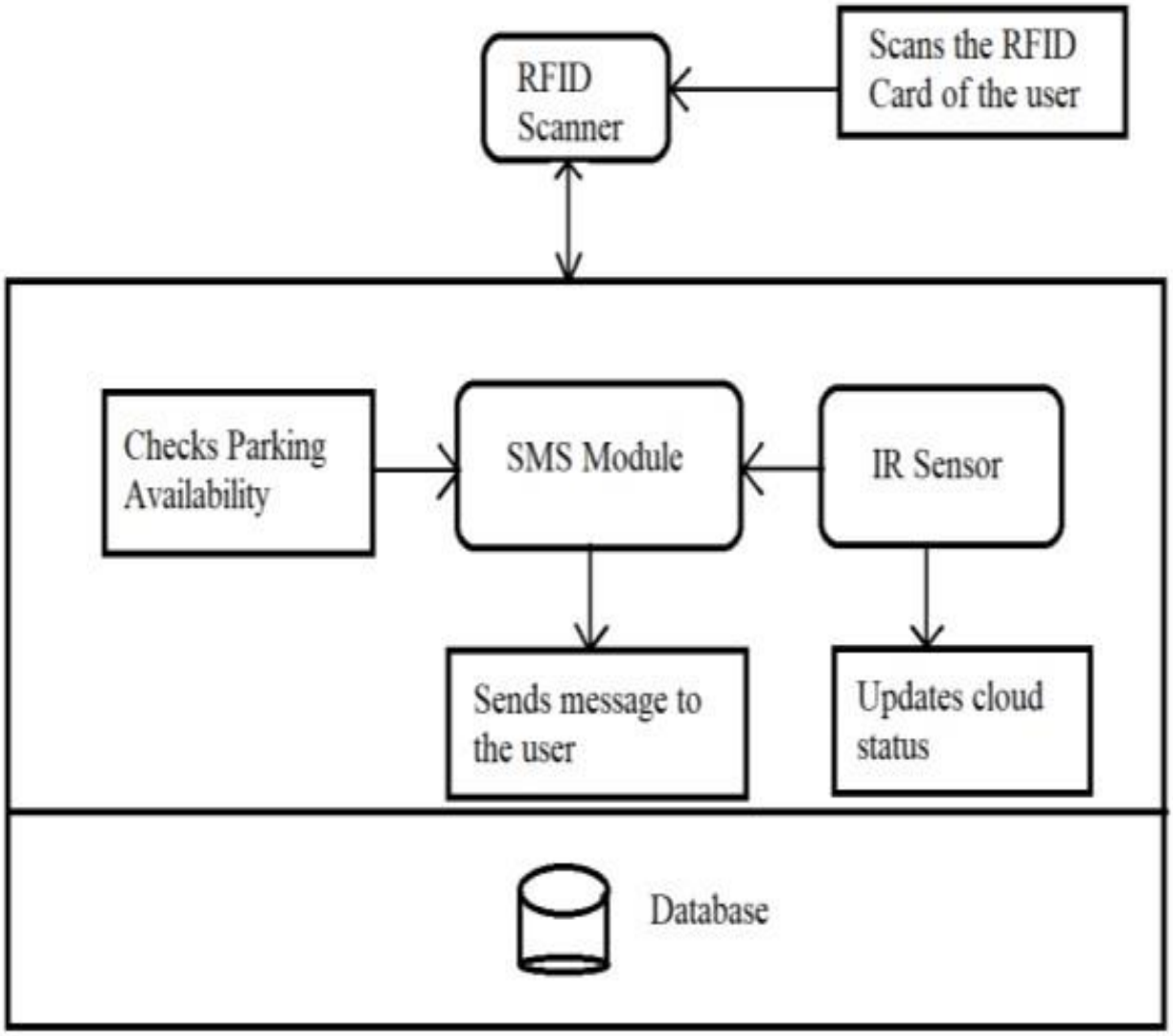
module sends the message according to the availability. IR sensor sends the signals according to the presence of the vehicle.



C. Software

The cloud server acts as a mediator between the modules. The cloud server is connected to the Wi-Fi module. The user receives messagesthrough the SMS module while the car enters and exits the parking area using RFID card. The messages sent by the SMS module are managed by the cloud. As soon as the IR sensor detects the car, the

status of the cloud will be updated from 0 to 1 and when the car leaves the parking area the status of the car will be updated from 0 to 1.



DETAILS OF THE MODULE A. GSM Module

The GSM module is a circuit which is used to setup communication between mobile phones and microcontroller. It is used to send SMS, MMS and voice messages through mobile network. GPRS extension in GSM allows high data transmission. GSM uses time division multiple access approach for transmission.

C. RFID Card

RFID tags are made up of integrated circuit (IC), an antenna, and a substrate. It is an identification badge or credit card that transfers its contents about an object to the reader module. RFID tag transfers data about an object through radio waves.When RFID tags are attached to devices they can also be used for tracking.

D. READER Module

This module is a device which scans and gathers the information from the RFID Card. This card can be used to track objects. As the car enters the parking area, the user scans the RFID card and all the information stored in card is transferred to the admin through this module.

E. Servo Motor

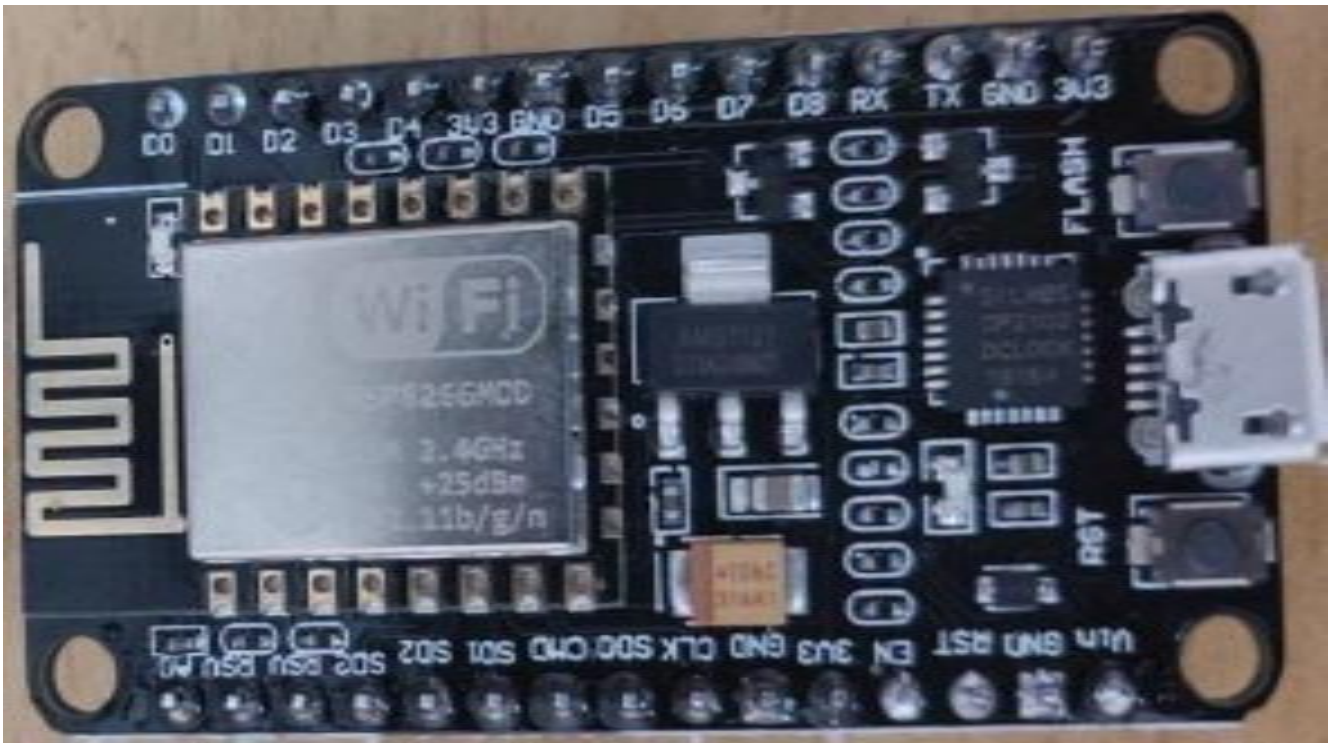
It is a rotator device that allows the control of angular as well as linear motion. A servo motor is used for the opening and closing of the gate.Servo drivetransmits electrical signals to the servo motor for producing motion.

F. Arduino Nano

It is a compact board which can be used in various devices and various field. It has overall 22 input/output pins out of which 14 pins are digital pins. It has a flash memory of about 32 kb. These pins can control the operations of digital pins as well as analogy pins. This module is a breadboard-friendly board which can be easily used anywhere.

G. WIFI Module

It is used to send data from embedded system to the internet using URL by HTTP POST method using TCP/IP protocol. It is developed by espressif systems. It is a 32 bit microcontroller with 80kb user data. It contains 16 gpio pins.



Source Code:

```
import colorama


from termcolor import colored


options_message = """
Choose:
1. To park a vehicle
2. To remove a vehicle from parking
```

```
3. Show parking layout
4. Exit
"""



class Vehicle:

    def _init_(self, v_type, v_number):
        self.v_type = v_type
        self.v_number = v_number
        self.vehicle_types = {1: 'c', 2: 'b', 3: 't'}

    def _str_(self):
        return self.vehicle_types[self.v_type]



class Slot:

    def _init_(self):
        self.vehicle = None

    @property
    def is_empty(self):
        return self.vehicle is None
```

```
class Parking:

    def _init_(self, rows, columns):
        self.rows = rows
        self.columns = columns
        self.slots = self._get_slots(rows, columns)

    def start(self):
        while True:
            try:
                print(options_message)

                option = input("Enter your choice: ")

                if option == '1':
                    self._park_vehicle()

                if option == '2':
                    self._remove_vehicle()

                if option == '3':
                    self.show_layout()
```

```
            if option == '4':
                break

        except ValueError as e:
            print(colored(f"An error occurred: {e}. Try again.", "red"))

    print(colored("Thanks for using our parking assistance system", "green"))

def _park_vehicle(self):
    vehicle_type = self._get_safe_int("Available vehicle types: 1. Car\t2. Bike\t3.
Truck.\nEnter your choice: ")

    if vehicle_type not in [1, 2, 3]:
        raise ValueError("Invalid vehicle type specified")

    vehicle_number = input("Enter vehicle name plate: ")
    if not vehicle_number:
        raise ValueError("Vehicle name plate cannot be empty.")
    vehicle = Vehicle(vehicle_type, vehicle_number)

    print('\n')
    print(colored(f"Slots available: {self._get_slot_count()}\n", "yellow"))
    self.show_layout()
    print('\n')
```

```
        col = self._get_safe_int("Enter the column where you want to park the vehicle: ")
        if col <= 0 or col > self.columns:
            raise ValueError("Invalid row or column number specified")

        row = self._get_safe_int("Enter the row where you want to park the vehicle: ")
        if row <= 0 or row > self.rows:
            raise ValueError("Invalid row number specified")

        slot = self.slots[row-1][col-1]
        if not slot.is_empty:
            raise ValueError("Slot is not empty. Please choose an empty slot.")

        slot.vehicle = vehicle

    def _remove_vehicle(self):
        vehicle_number = input("Enter the vehicle number that needs to be removed from parking slot: ")
        if not vehicle_number:
            raise ValueError("Vehicle number is required.")

        for row in self.slots:
            for slot in row:
                if slot.vehicle and slot.vehicle.v_number.lower() == vehicle_number.lower():
                    vehicle: Vehicle = slot.vehicle
```

```
                slot.vehicle = None
                print(colored(f"Vehicle with number '{vehicle.v_number}' removed from parking", "green"))
                return
        else:
            raise ValueError("Vehicle not found.")

    def show_layout(self):
        col_info = [f'<{col}>' for col in range(1, self.columns + 1)]
        print(colored(f"|{".join(col_info)}|columns", "yellow"))

        self._print_border(text="rows")

        for i, row in enumerate(self.slots, 1):
            string_to_printed = "|"
            for j, col in enumerate(row, 1):
                string_to_printed += colored(f"[{col.vehicle if col.vehicle else ' '}]",
                                             "red" if col.vehicle else "green")
            string_to_printed += colored(f"|<{i}>", "cyan")
            print(string_to_printed)

        self._print_border()

    def _print_border(self, text=""):
        print(colored(f"|{'-' * self.columns * 3}|{colored(text, 'cyan')}", "blue"))
```

```
def _get_slot_count(self):
    count = 0
    for row in self.slots:
        for slot in row:
            if slot.is_empty:
                count += 1
    return count

@staticmethod
def _get_slots(rows, columns):
    slots = []
    for row in range(0, rows):
        col_slot = []
        for col in range(0, columns):
            col_slot.append(Slot())
        slots.append(col_slot)
    return slots

@staticmethod
def _get_safe_int(message):
    try:
        val = int(input(message))
        return val
    except ValueError:
```

```
            raise ValueError("Value should be an integer only")


def main():
    try:
        print(colored("Welcome to the parking assistance system.", "green"))
        print(colored("First let's setup the parking system", "yellow"))
        rows = int(input("Enter the number of rows: "))
        columns = int(input("Enter the number of columns: "))

        print("Initializing parking")
        parking = Parking(rows, columns)
        parking.start()

    except ValueError:
        print("Rows and columns should be integers only.")

    except Exception as e:
print(colored(f"An error occurred: {e}", "red"))



if _name_ == '_main_':
    colorama.init()  # To enable color visible in command prompt
    main()
```

ArduinoProperties, Schematic Diagram, and Power

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560. It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button.

Specifications and Features of Arduino Mega 2560

Some of the detailed technical specification is mentioned below:- • Operating voltage of 5 V.

• 16 MHz clock speed.

• The number of digital I/O's pins is 54.

• The number of analog input pins is 16.

• 4 hardware serial ports.

• Flash Memory: 32 KB

• SRAM: 2 KB

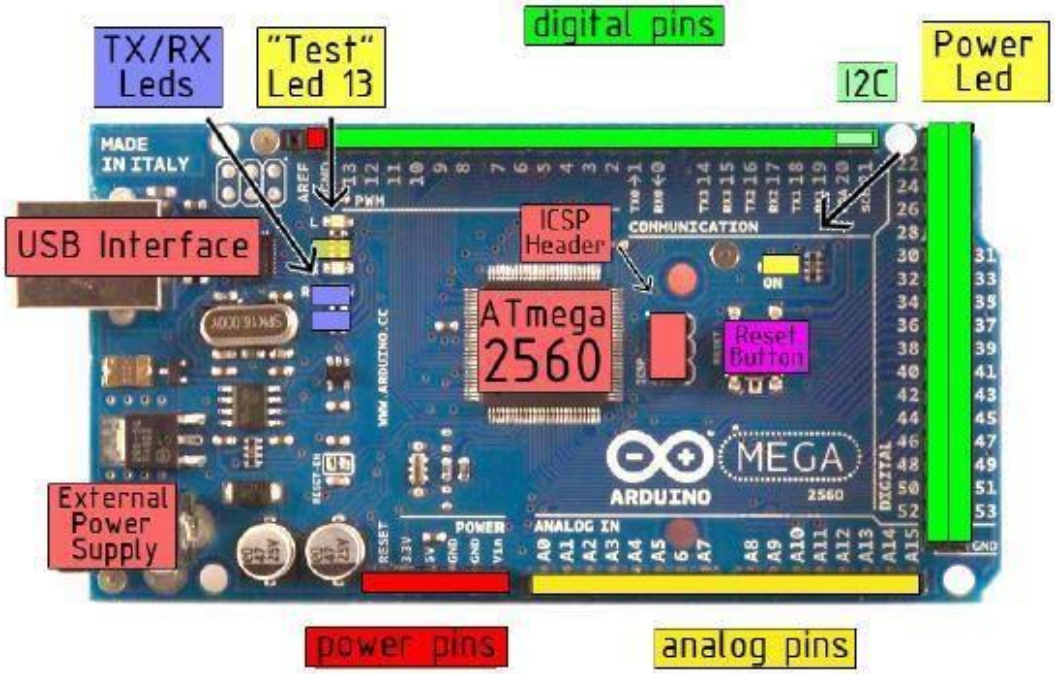
• EPROM: 1 KB

• Clock Speed: 16 MHZ

What sets a microcontroller apart from other processors are special circuits to deal with the needs of real time applications. There is an important feature of Arduino which is Software Serial library that allows for serial communication on any of the Mega‘s digital pins. The Arduino 2560 has a host of such features intended to maximize system

reliability, minimize cost through elimination of external components, provide power saving operating modes and offer code protection. These features as illustrated in figure 2.1 next pageare:

- Reset button
- Digital I/O
- Power pins
- Analog inputs
- Voltage regulator
- In-circuit serial programming
- FTDI USB chip and USB jack
- Power jack
- ICSP header



The Arduino can be used to develop stand-alone interactive objects or it can be connected to a computer to retrieve or send data to the Arduino and then act on that data. The Arduino is an amazing device. Users can us to make many things from interactive works of art to robots. With a little enthusiasm to learn how to program the Arduino and make it interact with other components a well as a bit of imagination.Users can build many things.TheArduino can be connected to LED Matrix displays, RFID readers, buttons, switches,motors, temperature sensors, pressure sensors, distance sensors, webcams, printers, GPS receivers, and Ethernet modules. The Arduino board is made of an Atmel AVR Microprocessor, a crystal or oscillator (basically a crude clock that sends time pulses to the microcontroller to enable it to operate at the correct speed) and a 5-volt linear regulator. USB connector is used to connect to a PC or Mac to upload or retrieve data. The board exposes the microcontroller I/O (Input/Output) pins to enable you to connect those pins to other circuits or to sensors, etc.