

**INTERNET OF THINGS [SMART PARKING]**

**PHASE 2: INNOVATION**

Creating a smart parking system involves several components, including hardware (such as sensors and cameras) and software (for data processing and user interfaces). Below is a simplified example of a Python program that simulates a basic smart parking system using a command-line interface. This program doesn't include actual sensor integration but demonstrates the logic.

**CODE:**

class ParkingSpace:

def \_\_init\_\_(self, space\_id):

self.space\_id = space\_id

self.is\_occupied = False

class SmartParkingSystem:

def \_\_init\_\_(self, num\_spaces):

self.spaces = [ParkingSpace(space\_id) for space\_id in range(1, num\_spaces + 1)]

def park\_car(self, space\_id):

if 1 <= space\_id <= len(self.spaces):

if not self.spaces[space\_id - 1].is\_occupied:

self.spaces[space\_id - 1].is\_occupied = True

print(f"Car parked in space {space\_id}")

else:

print(f"Space {space\_id} is already occupied.")

else:

print(f"Invalid space {space\_id}.")

def leave\_car(self, space\_id):

if 1 <= space\_id <= len(self.spaces):

if self.spaces[space\_id - 1].is\_occupied:

self.spaces[space\_id - 1].is\_occupied = False

print(f"Car left space {space\_id}")

else:

print(f"Space {space\_id} is already empty.")

else:

print(f"Invalid space {space\_id}.")

def display\_status(self):

for space in self.spaces:

status = "Occupied" if space.is\_occupied else "Empty"

print(f"Space {space.space\_id}: {status}")

def main():

num\_spaces = 10 # Number of parking spaces

parking\_system = SmartParkingSystem(num\_spaces)

while True:

print("\nSmart Parking System")

print("1. Park Car")

print("2. Leave Car")

print("3. Display Parking Status")

print("4. Exit")

choice = input("Enter your choice: ")

if choice == "1":

space\_id = int(input("Enter the space ID to park the car: "))

parking\_system.park\_car(space\_id)

elif choice == "2":

space\_id = int(input("Enter the space ID to leave the car: "))

parking\_system.leave\_car(space\_id)

elif choice == "3":

parking\_system.display\_status()

elif choice == "4":

print("Exiting the program.")

break

else:

print("Invalid choice. Please try again.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

This Python program creates a basic smart parking system with a defined number of parking spaces. Users can park and leave cars in parking spaces and view the parking status. Remember that this is a simplified simulation, and a real-world smart parking system would require hardware integration and more complex software

**SMART PARKING SYSTEM:**

A smart car parking system using IoT can address many issues and tasks. For example, a driver can view available parking slots directly from their smartphone with such a solution. Companies, in turn, can supervise their parking spaces more efficiently. And most importantly, they can do it remotely.

Here are a few tasks an IoT intelligent parking system can tackle.

**Access Control and Management**

* Parking lot owners can use access control and management
* systems to allow only authorized users to enter a parking area. Here’s how it works.
* The car parking system using IoT takes a user authorization mechanism through a mobile app or license plate scanning. At the same time, the controller on the barrier or gate may allow or refuse drivers to park their cars according to the set parameters.
* WebbyLab’s project [Propuskator is an example of such an access control and management system](https://webbylab.com/cases/acs-remote-controller/). It uses a controller connected to the gate or barrier and paired with a [2Smart Cloud mobile application](https://webbylab.com/cases/2smart/) to grant or restrict access to the territory.
* Using a 2Smart Cloud mobile application for access control and management.

**Parking Monitoring**

* Another task that an IoT Smart parking management system solves is monitoring the movement of people and cars in a parking lot.
* Integrating video surveillance into the system can improve parking administration and troubleshoot potential problems, such as traffic accidents or theft, by controlling which cars enter the area.

**Parking Reservation**

* An IoT-based smart parking system also solves the problem of overcrowded parking lots. It allows drivers to reserve parking spaces in advance through a mobile application or web interface.
* It’s also possible to use individual solutions like parking locks for reservations. These are devices installed directly on parking slots. When a parking space owner leaves the territory, a parking lock will block entry for other cars.

**Parking Management**

Parking administrators can use an IoT-based parking system to supervise all processes at the facility. In particular, this solution monitors parking space availability and facilitates the billing process.

**IoT Smart Parking System: Working Principles & Architecture**

A smart parking system uses IoT devices and sensors to collect real-time data on parking lot occupancy and transmits this information to the cloud or local network. It also involves [building IoT apps](https://webbylab.com/blog/building-iot-apps-in-2022-technologies-costs-tips/) for end-users, like parking administrators and drivers. They can adopt this mobile or web application and access the necessary data on available parking spaces, pricing, etc.

As a rule, an IoT-based parking system comprises the following components.

**Hardware**

* IoT-based smart parking system deployment requires integrating various devices, sensors, and microcontrollers.
* For example, it can be a microcontroller transmitting data to the cloud environment or a Bluetooth beacon. With its help, consumers can control parking locally. Such systems also integrate with video surveillance cameras, video recorders, or automatic number plate recognition (ANPR) cameras.
* Generally, there are several options for IoT devices and sensors you can use. It all depends on your business needs.

**Cloud-Based IoT Services**

* An IoT-based car parking system usually requires cloud-based services like AWS IoT, AWS Lambda, or Microsoft Azure IoT Hub for data collection and further transmission to the user application. This process involves the following data path:

1.The sensors send information to the microcontroller

2.The microcontroller transfers data to the cloud.

**Network Protocols**

* A parking system also requires protocols to ensure IoT devices’ and sensors’ connectivity in the parking lot. These can be MQTT, LoRaWAN, [Zigbee protocol for wireless IoT networks](https://webbylab.com/blog/what-is-zigbee-protocol-in-wireless-iot-networks/), or else.
* Such a system also requires video transmission protocols if it uses video surveillance. For example, WebbyLab used RTCP for our [Propuskator](https://webbylab.com/cases/acs-remote-controller/) project. This way, we provided a real-time video stream users can monitor through the application.
* The microcontrollers for a Propuskator access control and management system.

**User Interface**

* A mobile or web application is the final component of an IoT-based smart parking system. As a rule, such apps ensure parking management, time tracking, reservation, billing tools, data logging, remote video surveillance, guest passes, and driver authorization.
* After connecting those components, the system will work the following way:

**The architecture of an IoT-based smart parking system.**

**IoT Sensors Used to Create a Smart Parking System**

[Smart parking solution development](https://webbylab.com/iot/iot-solutions-for-smart-home-devices/) involves various sensors. Let’s consider the most common options.

**Ultrasonic Sensors**

The integration of ultrasonic sensors, which measure the distance between objects using ultrasonic waves, allows for precise parking. However, such devices have one drawback — the sensor might get blocked with dirt.

**Electromagnetic Field Sensors**

An electromagnetic field sensor detects and measures changes in the magnetic field. This way, it reacts to the approach of metal objects.

**Infrared Sensors**

Infrared (IR) sensors emit an infrared signal and catch the reflection of this signal from the environment. Integrating such a device enables measuring the temperature or detecting movement.

The smart parking development process. It consists of the following stages.

**1. Concept**

First, you come up with the concept of your IoT-based parking solution. At this stage, consider which sensors, microcontrollers, cloud-based services, and network protocols to use. You also decide which app to create: mobile, web, or both.

**2. Prototype**

Next, you engage the Internet of Things experts to [build an IoT prototype](https://webbylab.com/blog/how-much-does-it-cost-to-build-iot-prototype/). It’s the initial version of your smart parking system. You can test and analyze the prototype to suggest further improvements.

**3. Design**

At this stage, your experts create the user interface and select features for the MVP. The main goal is to create a user-friendly app suitable for all major operating systems.

**4. Development**

This step involves mobile, front-end, and back-end, hardware developers. They implement the technical side of your IoT-based smart parking system MVP and connect it with ready-made design solutions.

**5. Testing**

Once every component of your IoT smart parking system is connected, it’s time to test it. First, the hired experts examine the MVP for flaws and limitations and check if it achieves the goals set. Then, your specialists fix bugs and offer updates for your system.

**Smart Parking Solution Development**

Why wait until your competitors implement IoT smart parking systems? Build and apply your IoT parking system solution and start managing your parking lots efficiently. Our skilled team will help you with that, using extensive experience in IoT solutions development. Feel free to [contact WebbyLab](https://webbylab.com/contact-us/) to build a decent smart parking system.

