Creating a Smart Parking project for ESP32 on the Wokwi platform involves using the ESP32 microcontroller to detect and manage parking spaces, and then visualizing the data on a virtual interface provided by Wokwi. Here's a step-by-step guide on how to create such a project:

**Components Needed:**
1. ESP32 development board
2. Ultrasonic distance sensors (HC-SR04) for each parking space
3. Breadboard and jumper wires
4. Wokwi virtual simulator (https://wokwi.com/)

**Project Steps:**

1. **Hardware Setup:**

a. Connect the HC-SR04 ultrasonic sensors to your ESP32 board. You will need one sensor per parking space.

b. Wire the HC-SR04 sensors as follows:
  - VCC to 5V on ESP32
  - GND to GND on ESP32
  - Trig to a digital GPIO pin on ESP32 (e.g., GPIO2)
  - Echo to another digital GPIO pin on ESP32 (e.g., GPIO4)

c. Connect all the sensors in the same way, one for each parking space you want to monitor.

2. **Programming:**

a. Write an Arduino sketch for the ESP32 that reads the distance data from the ultrasonic sensors.

```cpp
#include <Ultrasonic.h>

Ultrasonic sensor1(GPIO_TRIGGER1, GPIO_ECHO1);
Ultrasonic sensor2(GPIO_TRIGGER2, GPIO_ECHO2);
// Add more sensors if needed

void setup() {
  Serial.begin(115200);
}

void loop() {
  long distance1 = sensor1.read();
  long distance2 = sensor2.read();
  // Read distances from more sensors if needed

  // Process distance data and manage parking spaces here

  delay(1000); // Delay for better readability
}
```

b. In the loop function, process the distance data from each sensor to determine whether a parking space is occupied or vacant. You can set a threshold distance to decide when a space is occupied.

c. You may want to use a data structure to keep track of the parking space status, e.g., an array of boolean values.

3. **Visualization:**

a. Go to the Wokwi platform (https://wokwi.com/) and create an account if you haven't already.

b. Create a new project and select the ESP32 as your target board.

c. Import the Arduino sketch you created earlier into the Wokwi editor.

d. Use the virtual interface provided by Wokwi to display the parking space status. You can use LEDs or any other graphical elements to represent the parking spaces.

4. **Testing:**

a. Simulate the project on Wokwi and observe how the parking space status changes based on the simulated distance measurements.

b. Fine-tune your code and interface as needed to ensure it works correctly.

5. **Deployment:**

a. Once your Smart Parking project works as expected in the virtual simulator, you can deploy it to a physical ESP32 board and connect it to real sensors in a parking area.

6. **Enhancements:**

Depending on your project's requirements, you can add features such as mobile app integration for real-time parking updates, data logging, and alerts when parking spaces are full or vacant.

Remember to refer to the ESP32 and HC-SR04 datasheets and the Wokwi documentation for detailed information on programming and using these components in your project.