

UNIVERSIDAD ORT
FACULTAD DE INGENIERÍA



IoT-PARKING

INFORME DE PROYECTO INTEGRADOR II PRESENTADO POR



JUAN I. SCAFFO - 257565
ING. ELECTRÓNICA



CARLOS CASCO - 251942
ING. TELECOMUNICACIONES



RONNY SUAREZ - 238874
ING. TELECOMUNICACIONES

PROFESORES

EMILIANO ESPÍNDOLA GONZÁLEZ
JUAN PEDRO SILVA GALLINO

MONTEVIDEO, 13 DE DICIEMBRE DE 2022

Resumen

Imagina que tienes una cita médica un día laboral cualquiera y para poder asistir debes pedir autorización en tu centro de trabajo. Lo que menos deseas al llegar a la clínica o centro de salud es tener que estar buscando donde estacionar tu auto ya que tienes el tiempo limitado. Lamentablemente, hoy día es difícil encontrar lugar de estacionamiento en zonas concurridas y no existen sistemas que te indiquen específicamente donde puedes dejar tu vehículo. Eso trae consigo que pierdas tiempo dando vueltas en la zona preguntándote: ¿Dónde estaciono? Sabemos que tus picos de estrés están elevados y por eso te presentamos una solución.

IoT-Parking es un sistema de estacionamiento inteligente basado en IoT (Internet of Things) que te permite saber dónde estacionar antes de salir para tu destino. Solo debes tener acceso a Internet a través de un Smartphone o una PC y nuestro sistema te asignará un lugar de estacionamiento que quedará reservado para ti durante el horario que indiques.

Palabras clave: Estacionamiento, IoT

Índice general

Resumen	III
1. Introducción	1
1.1. ¿Cómo surge la idea?	2
1.2. ¿Qué proponemos?	2
1.3. Objetivos	2
1.4. Caso de Uso	3
2. Descripción	5
2.1. Ecosistema IoT	5
2.2. Elementos que componen IoT-Parking	6
2.2.1. Sensores y actuadores	7
2.2.2. Conectividad	11
2.2.3. Nube IoT	12
2.2.4. Análisis y gestión de datos	13
2.2.5. Dispositivos e interfaz	13
2.3. Prototipo de la maqueta	13
3. Manos a la obra	15
3.1. Pruebas de concepto	15
3.1.1. Mecanismo de barrera	15
3.1.2. Flujo de datos a la nube	16
3.1.3. Mostrar lugar para estacionar	17
3.1.4. Sistema de reserva	18
3.1.5. Ruta hacia el lugar de estacionamiento	18
3.2. Maqueta	20
4. Conclusiones	27
5. Trabajo Futuro	29
Referencias	31

Capítulo 1

Introducción

Seguramente muchos se preguntarán: ¿Qué significa IoT? El sitio web Internet Society define IoT (Internet of Things) como “escenarios en los que la conectividad de red y la capacidad de cómputo se extienden a objetos, sensores y artículos de uso diario que habitualmente no se consideran computadoras, permitiendo que estos dispositivos generen, intercambien y consuman datos con una mínima intervención humana. Sin embargo, no existe ninguna definición única y universal” (K.Rose, S.Eldridge, L.Chapin, 2015)

Tal como indica la definición, IoT nos permite conectar “cosas” a través de una red para que intercambien información. En nuestro caso, necesitamos que los elementos que componen el sistema de IoT-Parking interactúen entre ellos a través de Internet y se compartan los datos que posteriormente serán analizados y procesados para tomar decisiones.

Habiendo entendido el concepto de IoT y los beneficios que trae a nuestras vidas traemos a la mesa un problema real que nos aqueja. Hoy día en muchas ciudades se hace extremadamente tedioso y estresante estacionar, ya sea en la calle o en un estacionamiento, y esto es porque empleamos mucho tiempo intentando encontrar un lugar donde dejar el auto. Pero ¿qué pasa si creamos un sistema que nos diga anticipadamente dónde estacionar? Gracias a IoT esto es posible dado que si dotamos al sistema de inteligencia podemos saber a través de sensores que lugares hay disponibles. Y si instalamos cámaras que lean la matrícula del auto podemos saber el puesto que le corresponde a ese vehículo. Además, las barreras sabrán cuando deben abrirse para un usuario específico. Todo esto orquestado en la nube de Azure y Thingsboard.

Parece interesante esta solución y desafiante para nuestro equipo, pero confiamos que será del agrado del lector el trabajo que presentamos a continuación. Vamos a describir un escenario real tomando como referencia la Universidad ORT y como con IoT-Parking podemos ayudar a los estudiantes, profesores y trabajadores a reducir el tiempo que emplean para estacionar.

1.1. ¿Cómo surge la idea?

Imagina que un estudiante termina su jornada laboral a las 18 horas y el tiempo que le toma en llegar a la Facultad de Ingeniería de la ORT es aproximadamente 20 minutos. Mientras va conduciendo hacia su destino va pensando en la odisea que será para él encontrar un lugar donde estacionar su auto y que probablemente no llegue en tiempo a su clase de las 18:30. Esto genera un pico de estrés que afecta, entre otras cosas, el proceso de aprendizaje. A un profesor le puede ocurrir lo mismo y no es correcto tener esperando a un grupo de estudiantes la llegada de su profesor debido a que no ha logrado estacionar.

1.2. ¿Qué proponemos?

Ahora imaginemos el mismo escenario del estudiante saliendo de su centro laboral a las 18 horas, pero con una particularidad. Un rato antes de salir inició sesión en la web de IoT-Parking y reservó un lugar de estacionamiento durante su estancia en la ORT. Ya no tendrá que preocuparse por buscar un lugar para estacionar, simplemente llega a las instalaciones de IoT-Parking¹, su matrícula será leída, se habilitará su acceso y será guiado a su puesto de estacionamiento. Se terminaron los eternos 10 minutos buscando donde estacionar y ahora sí podrá aprovechar al máximo su horario de clases.

1.3. Objetivos

Nuestro objetivo principal al enfrentar nuestro proyecto fue brindar una alternativa a los estacionamientos tradicionales dotando nuestro sistema de inteligencia para que sea eficiente, práctico y atractivo al cliente a la hora de estacionar su vehículo. Adicionalmente describimos otros objetivos que tuvimos en cuenta y hacen nuestra propuesta interesante frente a otras opciones similares:

- Crear una página web fácil e intuitiva para cualquier tipo de usuario.
- El sistema de reservas permite al usuario poder seleccionar un lugar específico, fecha y hora estimada de ingreso/egreso del parking.
- El ingreso al Parking debe ser lo más rápido posible para evitar congestión en la entrada.
- Establecer un reconocimiento de matrícula a partir de una imagen utilizando un OCR (Optical Recognition System).
- Instalar tiras leds que guiarán al usuario hasta el lugar asignado.

¹Se supone existe un Parking en las inmediaciones de la ORT

1.4. Caso de Uso

Todo comienza cuando un usuario escucha sobre IoT-Parking. Un conocido le comenta que existe un sistema de estacionamiento inteligente que le permite reservar un lugar en un parking cercano el tiempo que desee y nadie ocupará ese puesto durante el horario reservado.

El usuario busca en Google la página web de IoT-Parking y como es nuevo en la plataforma debe registrarse como Usuario Nuevo. Todo el proceso transcurre de la siguiente manera:

Reserva del lugar

1. Cada usuario nuevo deberá registrarse en la página web e ingresar los datos que sean solicitados.
2. Iniciar sesión en su cuenta.
3. Dirigirse a la sección “Reservar” y elegir el puesto de estacionamiento que desee.
4. Ingresar fecha y hora de entrada y estimada de salida.

Llegada al estacionamiento

5. El auto llega a la entrada principal del estacionamiento.
6. Se detecta con un sensor su presencia y se toma una imagen utilizando una cámara.
7. La imagen es enviada a un servidor que utilizará OCR para descifrar la matrícula del auto.
8. Se abre la barrera del portón principal y se muestra en un display el lugar que le corresponde.
9. El auto ingresa en el estacionamiento y se ilumina un camino de leds hacia el lugar asignado.
10. La barrera del lugar asignado se abre para el ingreso del vehículo.
11. El auto se retira del estacionamiento.
12. Cuando el sensor ubicado en cada puesto de estacionamiento detecta que no hay auto, envía esa información al sistema para que la barrera de ese lugar se baje nuevamente.

Capítulo 2

Descripción

Nuestra idea es preparar una maqueta que simule un estacionamiento y poder mostrar las funcionalidades de IoT-Parking. Pretendemos hacer un recorrido por todas las etapas que atravesamos en la confección y puesta en práctica de la solución. En este capítulo describiremos las partes principales que componen IoT-Parking y la interacción entre ellas.

2.1. Ecosistema IoT

Antes de hablar sobre nuestro proyecto hay que entender qué es y como se compone un ecosistema IoT. Según el artículo “¿Qué es un ecosistema IoT?” (D. Hattingh, 2022) los componentes principales que conforman un ecosistema IoT son:

- **Sensores y actuadores.**
Esta capa la conforman los dispositivos (sensores, cámaras, servomotores) que recopilan datos y envían a las capas superiores para su procesamiento. Reciben instrucciones y ejecutan las tareas que fueron enviadas por el usuario o sistema automatizado.
- **Conectividad.**
Permite el intercambio de información entre los sensores o dispositivos actuadores y la nube de IoT a través de protocolos inalámbricos como Wi-Fi, Bluetooth, NB-IoT, etc. Otros protocolos que permiten el correcto flujo de datos a través de esta pasarela IoT son HTTP, HTTPS y MQTT.¹
- **Nube IoT.**
En la nube los datos serán almacenados, procesados y posteriormente analizados con el objetivo de tomar decisiones lo más rápido posible.

¹<https://mqtt.org/>

- **Análisis y gestión de datos.**

Se gestiona y analiza todo el volumen de datos para convertirlos en información fácil de procesar. Podemos ver reportes, alarmas, enviar datos de interés por e-mail, etc.

- **Dispositivos e interfaz.**

Permite administrar los dispositivos y ofrece características críticas para mantener el estado, la conectividad y la seguridad del sistema a lo largo de todos sus ciclos de vida.

En algunos textos se indican 4 capas en lugar de 5 y sus nombres no son exactamente iguales. El propósito es entender a grandes rasgos la funcionalidad de cada capa y el rol que juega cada elemento que la compone en la infraestructura IoT.

2.2. Elementos que componen IoT-Parking

Habiendo descrito en la sección anterior la arquitectura de IoT pasaremos a definir los elementos que componen la solución de IoT-Parking así como sus características principales y sus funciones dentro del ecosistema.

La Figura 2.1 muestra los elementos que conforman cada capa y la interacción entre ellos.

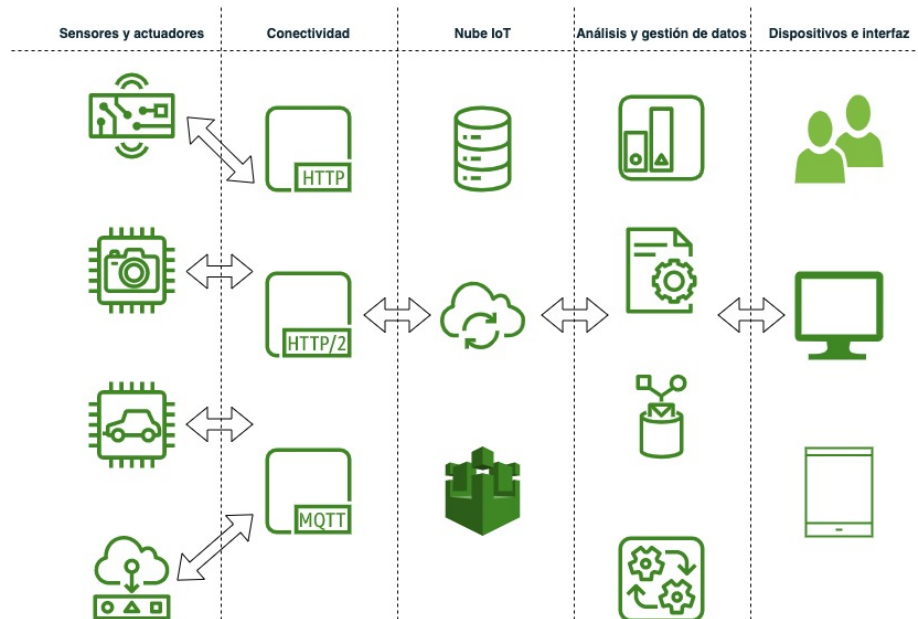


Figura 2.1: Arquitectura de IoT

2.2.1. Sensores y actuadores

En esta capa contamos con los siguientes elementos. Haremos un breve resumen de sus principales características.

■ Sensores de posición.

Los sensores de posición (Figura 2.2) se encargan de detectar cuando el auto esté ubicado en el lugar asignado con el objetivo de enviar esa información a las capas superiores y tener un control de cuáles son lugares disponibles u ocupados.

Sus principales características son:²

- Detectar la presencia de obstáculos entre 2 a 50 cm.
- Ángulo de detección 35°.
- Comparador LM393 on-board.
- Orificio de instalación para facilitar su uso.
- Indicador de alimentación (LED rojo).
- Indicador de salida digital (LED verde).
- Conexión de 3 hilos.
- Tensión de alimentación: 3.3v a 5v



Figura 2.2: Sensor MK0434

■ Servo motor SG90.

El Servomotor SG90 (Figura 2.3) es el encargado de abrir y cerrar la barrera. Este es un componente actuador, solo recibirá la indicación cuando

²Sitio web electronica.uy(Sensor MK0434, 2022)

debe subir/bajar la barrera.

Sus principales características son:³

- Micro Servo Tower-pro
- Velocidad: 0.10 sec/60° @ 4.8V
- Torque: 1.8 Kg-cm @ 4.8V
- Voltaje de funcionamiento: 3.0-7.2V
- Temperatura de funcionamiento: -30 °C 60 °C
- Ángulo de rotación: 180°
- Ancho de pulso: 500-2400 µs
- Longitud de cable de conector: 24.5cm



Figura 2.3: Motor servo micro SG90

■ NodeMCU.

El NodeMCU (Figura 2.4) es el módulo que permite conectar con los sensores y servo motores. Cuenta con un chip ESP8266 WiFi que permite conectarse a una red WiFi a través de la cual viajaran los datos en Internet. Sus principales características son:⁴

- Interfaz: CH340G
- Expansión: GPIO D0 D8, SD0 SD3, AD0
- Conectividad: WiFi b/g/n
- Alimentación: 5V via MicroUSB
- Compatibilidad: AT, Lua, Arduino
- Chipset: ESP8266
- Indicadores: LED de actividad en el módulo ESP8266
- Antenas: Integrada
- Frecuencias: 2,4GHz
- Velocidad de transferencia: 110 460800 bps

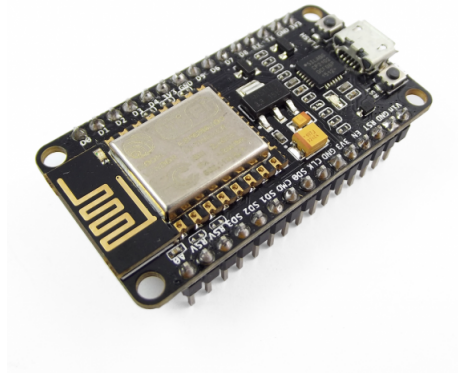


Figura 2.4: NodeMCU V3 ESP8266 WiFi

■ ESP32-CAM

La ESP32-CAM (Figura 3.2) es un módulo que se le puede incorporar una cámara OV2640. Es indispensable para el reconocimiento de la matrícula. Su función es tomar una foto de la matrícula del auto cuando se le indique. Inmediatamente la foto será subida a un servidor web que validará en una base de datos y se permitirá el acceso si el usuario reservó su lugar previamente.

Sus principales características son:⁵

- Voltaje de alimentación: 5VDC
- Voltaje entradas/salidas(GPIO): 3.3VDC
- SoM: ESP-32S (Ai-Thinker)
- SoC: ESP32 (ESP32-D0WDQ6)
- Wifi 802.11b/g/n, Bluetooth 4.2
- Antena PCB, también disponible conexión a antena externa
- 520KB SRAM interna, 4MB SRAM externa
- Cámara OV2640
- Resolución fotos: 1600 x 1200 pixels
- Óptica de 1/4"
- Dimensiones: 27*40.5*6 mm

³Sitio web **Eneka**(Servo motor SG90, 2022)

⁴Sitio web **Eneka**(NodeMcu, 2022)

⁵Sitio web **Electrónica.uy**(ESP32-CAM, 2022)

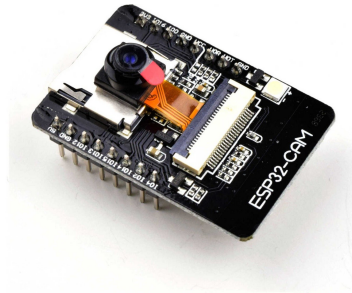


Figura 2.5: ESP32-CAM

■ Display OLED

El Display (Figura 2.6) mostrará el lugar que fue asignado al usuario. Sus principales características son:⁶

- Tamaño: 0.96"
- Color: Azul
- Ángulo de visión: más de 160 grados
- Plataformas compatibles: Arduino, 51 series, MSP430 series, STIM32/2, SCR chips
- Bajo consumo de energía: 0,04W durante el funcionamiento normal
- Soporte de voltaje: 3,3V - 5V DC
- Temperatura de trabajo: -30^o - 80^o
- Volumen: 27MM * 27MM * 4,1MM
- Driver IC: SSD1306

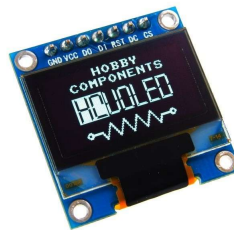


Figura 2.6: Display OLED 0.96

⁶Sitio web **Eneka**(Display OLED 0.96, 2022)

■ Tira LED WS2812B

La tira LED (Figura 2.7) tiene como función guiar al cliente hasta su lugar de estacionamiento.

Sus principales características son:⁷

- Alimentación: 5V
- Potencia: 18W
- Leds: WS2812B (5050 SMD led)
- Niveles de brillo de cada canal de color: 256 (8 bits)
- Anchura: 10mm
- Protección contra el agua
- Se puede cortar en cualquier separación entre leds.



Figura 2.7: LED WS2812B

2.2.2. Conectividad

Gracias a los módulos Wi-Fi que tienen las placas ESP8266 la comunicación entre la capa **Sensores y actuadores** 2.2.1 y las capas superiores se hace a través de una red inalámbrica. Cada placa obtiene una dirección IP que provee el router Wi-Fi a través del protocolo DHCP y le permite acceder a recursos en Internet.

Luego de garantizar conectividad IP necesitamos enviar información a dos sitios que hemos definido para nuestro proyecto: Servidor web alojado en la nube de Azure y la plataforma de gestión IoT **Thingsboard**. Para ello nos apoyamos en las ventajas de los protocolos HTTP y FTP.

A continuación una breve descripción de cada uno de ellos:

⁷Sitio web **e-ika**(LED Ws2812B, 2022)

■ HTTP

Según el sitio web Wikipedia podemos definir brevemente a HTTP (Hypertext Transfer Protocol) como “el protocolo de comunicación que permite las transferencias de información a través de archivos (XML, HTML...) en la World Wide Web”.(HTTP, 2022)

Podemos hablar mucho de HTTP y sus diferentes versiones pero no es el objetivo de esta documentación.

A efectos prácticos relacionados con nuestro proyecto mostraremos un pequeño ejemplo de como utilizamos el protocolo HTTP en IoT-Parking.

Cuando un usuario desea reservar un lugar para estacionar debe acceder mediante su Smartphone o PC al sitio web de IoT-Parking. El servidor alojado en la nube de Azure está “escuchando” conexiones en el puerto 80/TCP, puerto utilizado por defecto por HTTP para la transferencia de datos entre el cliente, en este caso el usuario, y el servidor web.

■ FTP

Según el sitio web Wikipedia podemos definir brevemente a FTP (File Transfer Protocol) como “un protocolo de red para la transferencia de archivos entre sistemas conectados a una red basado en la arquitectura cliente-servidor”.(FTP, 2022)

En IoT-Parking hacemos uso de este protocolo para la comunicación entre la ESP32-CAM y nuestro servidor en la nube de Azure utilizando el puerto 21/TCP para recibir las imágenes con la matrícula del auto.

2.2.3. Nube IoT

Como nuestro sistema tiene la particularidad de contar con un sistema de reservas debíamos tener un servidor con el rol de Web-Server. Este servidor también nos ayudaría con el servicio FTP que debíamos tener en disponible para la transferencia de imágenes. Y así fuimos viendo que si configurábamos un servidor en la nube de Azure⁸ podíamos aprovechar las ventajas que este nos brinda para alojar todos los servicios que necesitábamos. Entre las grandes ventajas que tiene la nube de Azure es que te permite crear un servidor web con el sistema operativo de Microsoft que elijas y las características de hardware que necesites de manera gratuita por un plazo de tiempo determinado. Para nuestro proyecto esta opción era factible.

Los principales roles que tiene este servidor son los siguientes:

■ Web Server

Este servicio permite que los usuarios accedan al sitio de reservas y gestionen el tiempo que van a estar en el estacionamiento.

■ SQL Server

Este servicio nos ayuda a manejar una base de datos que contiene la in-

⁸<https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-is-azure/>

formación de cada cliente como la matrícula del auto, usuario, contraseña de acceso a la plataforma.

- **FTP Server**

Con este servicio podemos recibir las imágenes desde la ESP32-CAM con la matrícula de cada auto.

2.2.4. Análisis y gestión de datos

En esta capa creamos un código que nos permite tomar la imagen ya almacenada en el servidor y convertirla en letras y números, específicamente el formato de matrícula de auto. Esa matrícula será insertada en la base de datos y utilizada para manejar las reservas.

Luego que se asigne un puesto de estacionamiento, esa información será enviada a los Servo Motores para que abran la barrera del lugar que se reservó al usuario.

2.2.5. Dispositivos e interfaz

La página web será la interfaz que ayudará al usuario a reservar su puesto de estacionamiento. Utilizamos Node.js para levantar el servidor web y la página está en lenguaje JavaScript. Más adelante especificamos el formato de la página.

2.3. Prototipo de la maqueta

Para comenzar nuestro proyecto debíamos diseñar en primer lugar un diagrama que nos permitiera visualizar como debía quedar la maqueta a implementar. Luego de varias discusiones en el grupo, definimos que quedaría de la siguiente manera (Figura 2.8).

En el proceso de armado de la maqueta tuvimos que hacer algunos ajustes por cuestiones de practicidad y si bien el resultado final no fue idéntico a como inicialmente se planeó nos sirvió de mucha utilidad este diseño como guía para empezar el armado.

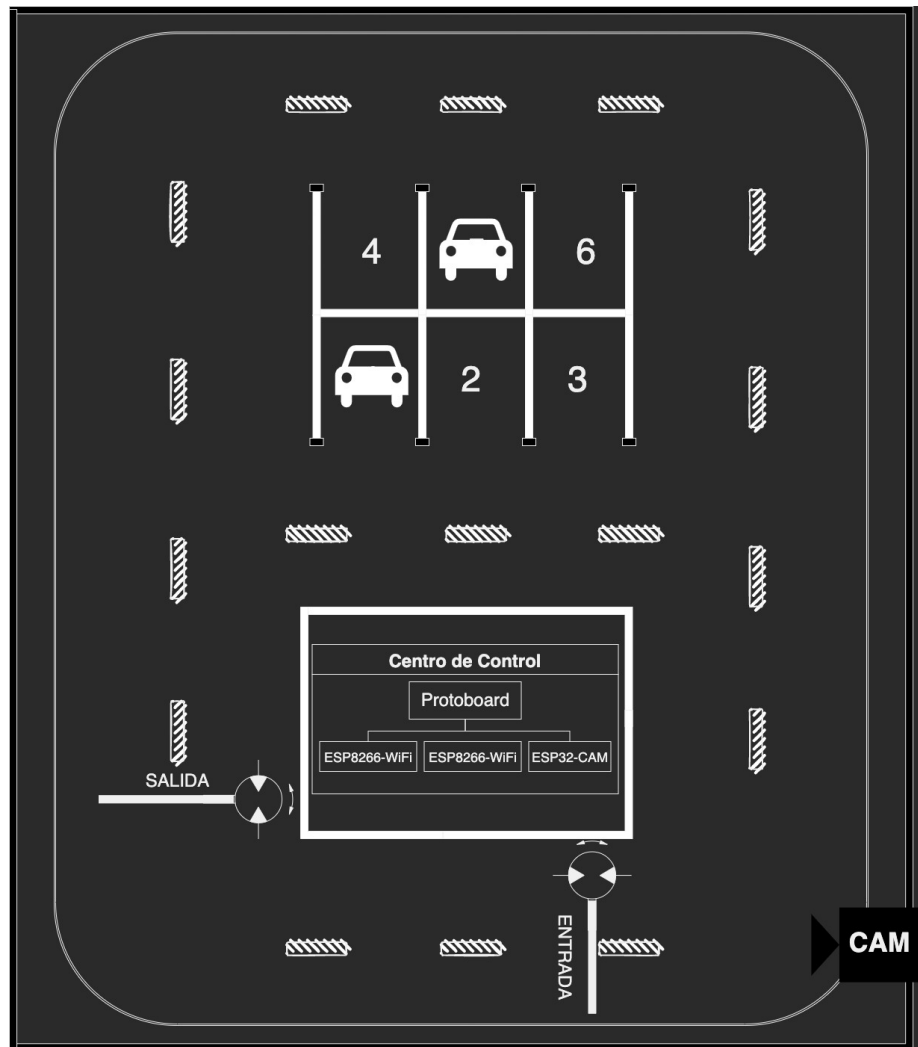


Figura 2.8: Diseño de maqueta

Capítulo 3

Manos a la obra

Llegó la parte interesante y desafiante para el equipo. Llevar a la realidad todo lo planificado e imaginado durante varias semanas.

Los primeros pasos luego de haber investigado y determinado la idea central fue hacer un Anteproyecto que nos sirvió de referencia para “bajar a tierra” todo lo conversado con el grupo. Toda la documentación del Anteproyecto y los archivos de código que creamos para las pruebas de concepto y en definitiva para el proyecto en sí se encuentra disponible en el repositorio de GitHub del equipo: <https://github.com/Conectando-Cosas-2022/IoTParking>¹

3.1. Pruebas de concepto

Para desarrollar la planificación del plan de trabajo tuvimos en cuenta diferentes prácticas de gestión de proyectos que nos permitió acercarnos a cumplir los objetivos en un tiempo adecuado.

Luego establecimos pruebas de concepto para dividir las tareas y lograr el objetivo final. Estas fueron las principales pruebas que planteamos en un inicio y el resultado de cada una.

3.1.1. Mecanismo de barrera

Prueba

Esta prueba de concepto tuvo como objetivo diseñar y probar el proceso mecánico de abrir o cerrar el portón principal y los lugares de estacionamiento. Probamos varios Servo Motores conectados de manera simultánea y logramos elegir cuál de las barreras se abrían y cerraban.

¹Para no cargar la documentación de código, decidimos indicar el link al repositorio de GitHub. En ese link estará disponible el código de cada una de las pruebas de concepto

Resultado

El resultado fue el esperado. Al inicio costó un poco de trabajo porque no giraba lo suficiente para abrir y cerrar la barrera. Luego investigamos y logramos el ángulo de 90° que estábamos buscando.

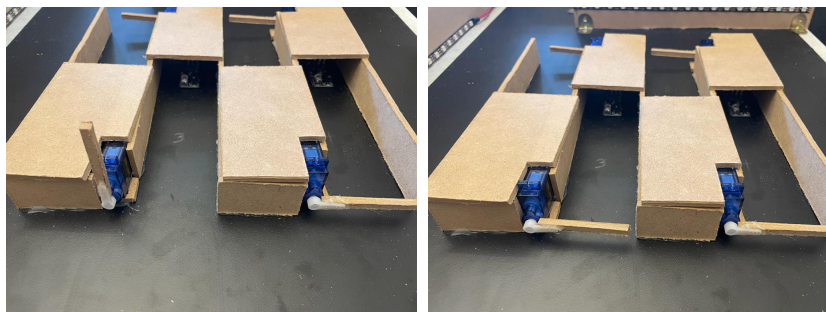
Los componentes de esta prueba de concepto fueron:

- Micro servo motor SG90.
- ESP8266.

Todos estos componentes fueron descriptos en la sección 2.2.1.

Enlace al código del prototipo: <https://github.com/Conectando-Cosas-2022/IoTParking/tree/main/Sweep>

Imágenes



(a) Barrera arriba.

(b) Barrera bajo.

Figura 3.1: Estados de la barrera.

3.1.2. Flujo de datos a la nube

Prueba

El objetivo de esta prueba fue el envío correcto de datos al servidor. Necesitábamos que la ESP32-CAM enviara una foto de la matrícula del auto. Una vez la imagen estuviera almacenada en el servidor, debíamos procesarla y tomar de ella el formato de matrícula, por ejemplo, **ABC1234**.

Resultado

El principal problema que hubo al realizar esta prueba fue la obtención del archivo de imagen en el servidor Web. Al principio comenzamos utilizando base64 para transmitir los datos binarios al servidor a través de un request HTTP. Sin embargo este string era demasiado largo, de todas maneras, se encontró una opción para permitir mensajes http más largos pero las imágenes a veces

contaban con corrupción debido a la velocidad del envío. La solución a este problema fue utilizar un servidor FTP de intermediario, la ESP32 sube el código al servidor FTP y una vez lo logra subir realiza un request HTTP al server para avisar que la imagen esta pronta. Los componentes de esta prueba de concepto fueron:

- ESP32-CAM
- Web Server
- FTP Server

Todos estos componentes fueron descriptos en la sección 2.2.1 y 2.2.3 Link al código del prototipo: <https://github.com/Conectando-Cosas-2022/IoTParking/tree/main/camaraESP32CAM>

3.1.3. Mostrar lugar para estacionar

Prueba

El propósito general de esta prueba de concepto debe ser el procesamiento de los múltiples datos captados por los sensores de los lugares del estacionamiento, principalmente se deberá calcular a partir de los mismos el próximo lugar para asignar y mostrarlo en el display.

Resultado

El resultado fue el esperado. Logramos mostrar la imagen en el display. Se ve un poco pequeña en la prueba de concepto pero en la maqueta logramos agrandar la letra.

Link al código del prototipo: https://github.com/Conectando-Cosas-2022/IoTParking/tree/main/ssd1306_128x64_spi

Imagen



Figura 3.2: Prueba de concepto - Display

3.1.4. Sistema de reserva

Prueba

El sistema de reservas será una capa adicional que se incorporará por encima del aviso de la asignación de lugares y el flujo de datos. En esta prueba de concepto, es importante que se pueda seleccionar, lugar, fecha y hora desde la aplicación web. Si el auto con la matrícula de la reserva ingresa en el estacionamiento, se debe preguntar al servidor cuál lugar está reservado. Si un auto nuevo ingresa en el estacionamiento, no se le debe otorgar un lugar reservado.

Resultado

Hubo ciertos factores que no habíamos contemplado en el Anteproyecto que fueron necesarios concretar una vez comenzamos con el sistema de reservas.

Uno de estos factores fue como se considerarán los lugares que están libres y tienen una reserva muy cercana a la hora actual. Por ejemplo, una persona llega al estacionamiento a las 3:10, el sistema le da el lugar 1 que esta libre pero resulta que este tiene una reserva a las 3:15, cuando la persona que había reservado ese lugar llega a esa hora, ya no tiene el lugar reservado. Nos parecía un sistema muy injusto y que podía generar varios conflictos por ese motivo se solucionó de la siguiente forma.

El sistema solo entregará lugares que estén libres y no tengan ninguna reserva en la próxima hora, esto implica que las personas que llegan al estacionamiento sin reserva, tienen como mínimo 1 hora de estacionamiento, luego de esto se deberán retirar si hay una reserva en ese mismo lugar, de lo contrario son libres de permanecer hasta que falte 1 hora para la próxima reserva.

Imágenes

Los componentes de esta prueba de concepto fueron:

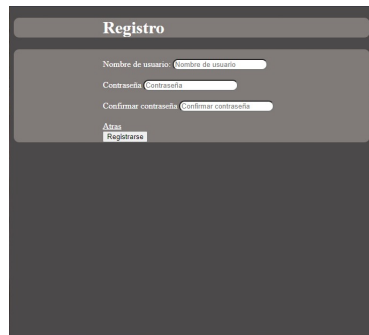
- Display
- Web Server
- SQL Server

Link al código de la prueba: <https://github.com/Conectando-Cosas-2022/IoTParking/tree/main/Node%20server>

3.1.5. Ruta hacia el lugar de estacionamiento

Prueba

Esta prueba pretende iluminar con luces led el camino hacia el lugar de estacionamiento asignado.



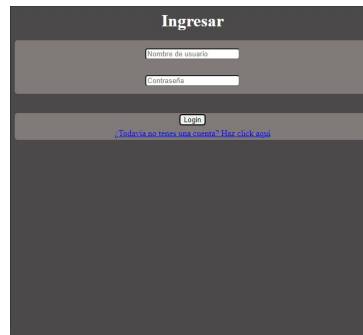
Registro

Nombre de usuario:

Contraseña:

Confirmar contraseña:

(a) Registro de Usuarios



Ingresar

Nombre de usuario:

Contraseña:

[¿Te olvidó de crear una cuenta? Haz click aquí](#)

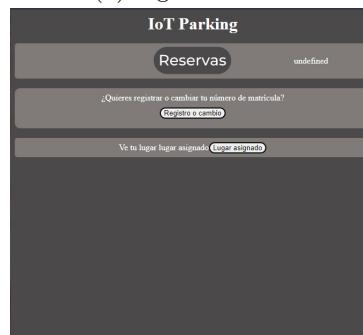
(b) Ingreso de usuarios



Registro de matrícula

Ingrese su matrícula:

(c) Registro matrícula



IoT Parking

Reservas undefined

¿Quieres registrar o cambiar tu número de matrícula?

Vé tu lugar lugar asignado:

(d) Página de reserva



Reserva de lugar

Indique los datos de la reserva

Escuche la matrícula: Seleccione la fecha a reservar: Seleccione la hora:

Escuche la duración (minutos):

Seleccione su lugar

1 2

3 4

(e) Reserva de lugar



Notificaciones

Indique si desea subir o bajar la barrera

Matrículas

Matricula: Lugar:

(f) Notificaciones

Figura 3.3: Página web de reservas.

Resultado

Intentamos generar una función que recibiera como parámetro una tira y ejecute el mismo código para lograr hacerlo de manera más prolija. No fue posible utilizando un parámetro tira en lugar de tira1 y tira2. La solución fue dejar el código con dos variables, una para cada tira. Otro problema que encontramos fue la necesidad de concretar la construcción de la base de la maqueta para

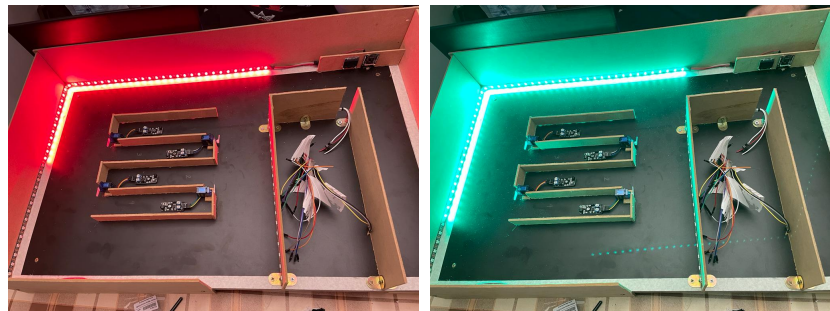
probar correctamente hasta que punto se deben prender las leds por lo que no pudimos realizar la prueba final.

Los componentes de esta prueba de concepto fueron:

- ESP32-CAM
- Tiras LED

Link a código de prueba: https://github.com/Conectando-Cosas-2022/IoTParking/tree/main/Pruebas_Leds

Imágenes



(a) Puesto estacionamiento 3

(b) Puesto estacionamiento 4

Figura 3.4: Guía para cada puesto de estacionamiento con luces LED.

3.2. Maqueta

A continuación mostraremos el proceso de armado de la maqueta en imágenes.



Figura 3.5: Base de la maqueta

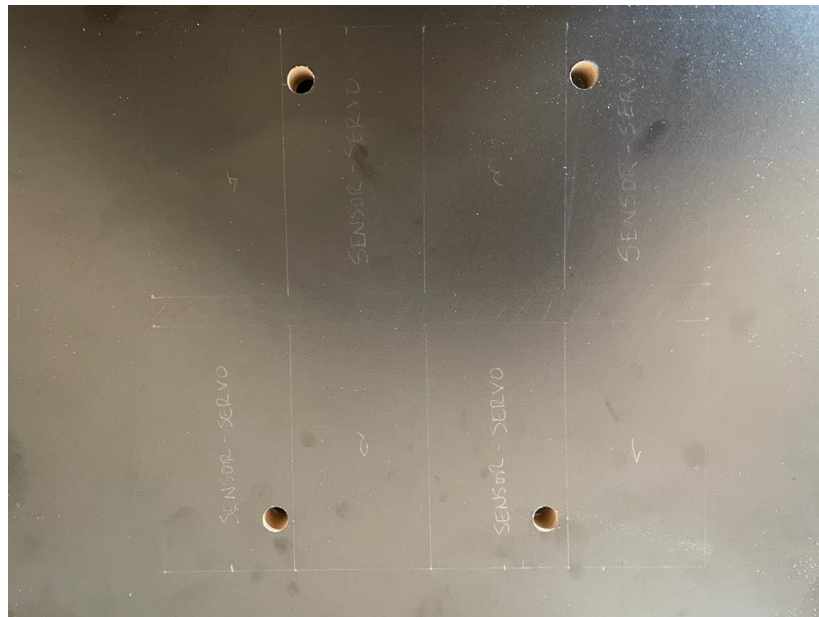


Figura 3.6: Agujeros para pasar los cables

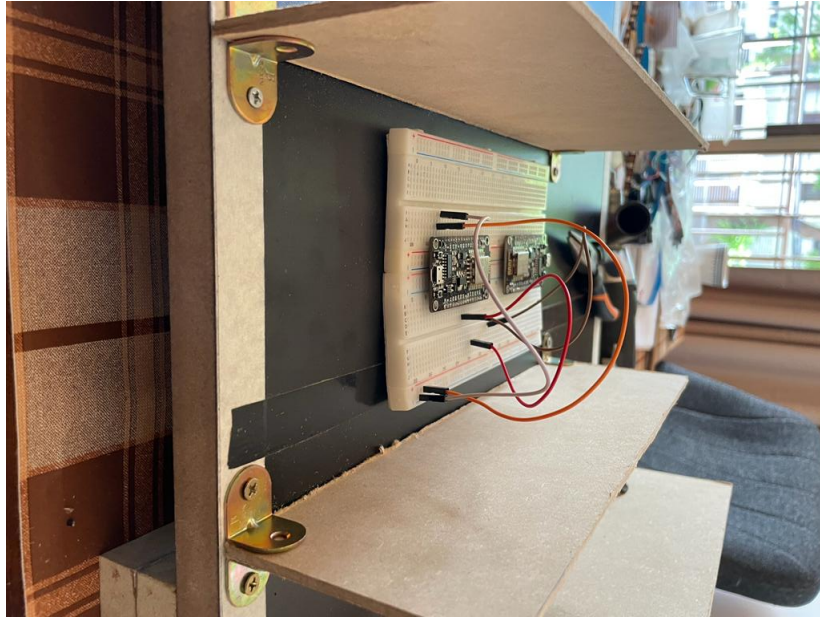


Figura 3.7: Centro de Control

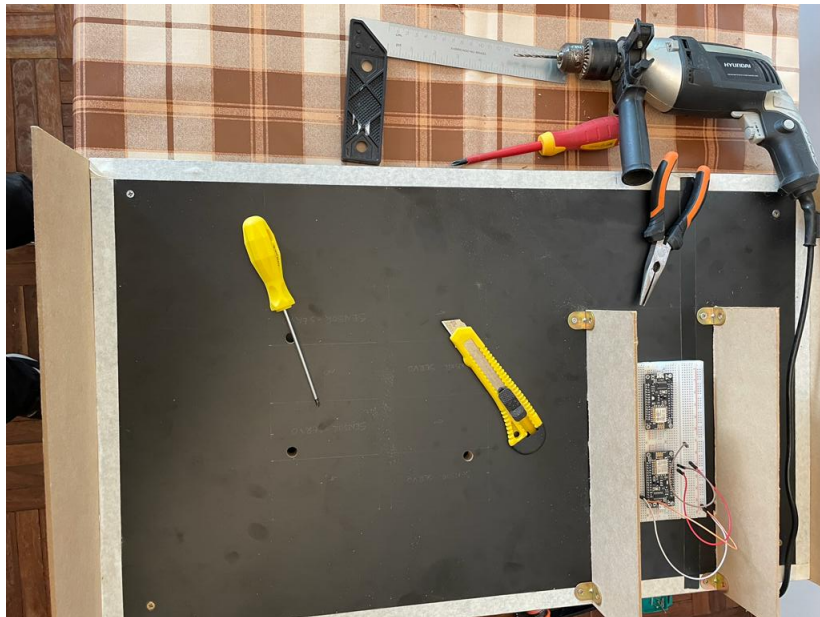


Figura 3.8: Vista Aérea - Distribución del espacio



Figura 3.9: Sensores, Servo Motores y Cables para Centro de Control



Figura 3.10: Organización de cables por debajo de la base



Figura 3.11: Identificación de cables en Centro de Control

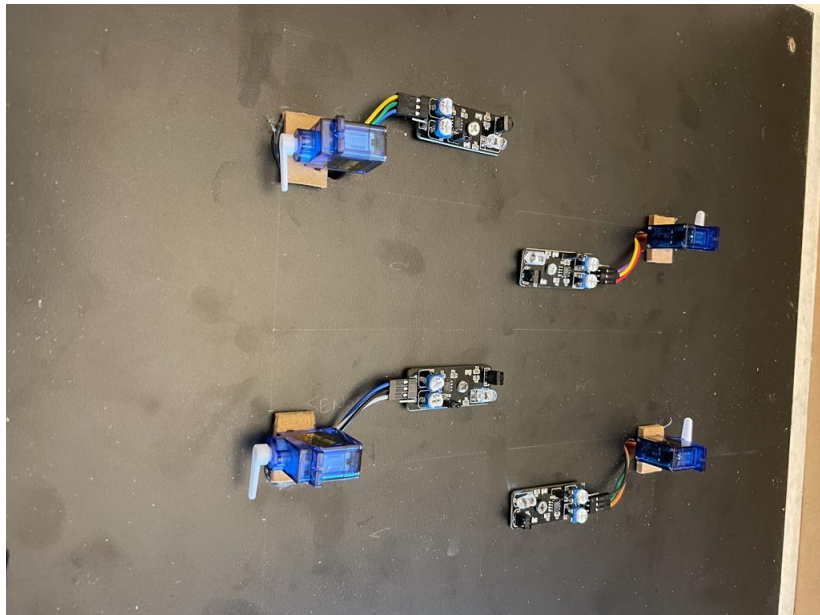


Figura 3.12: Ubicación de Sensores y Servo Motores

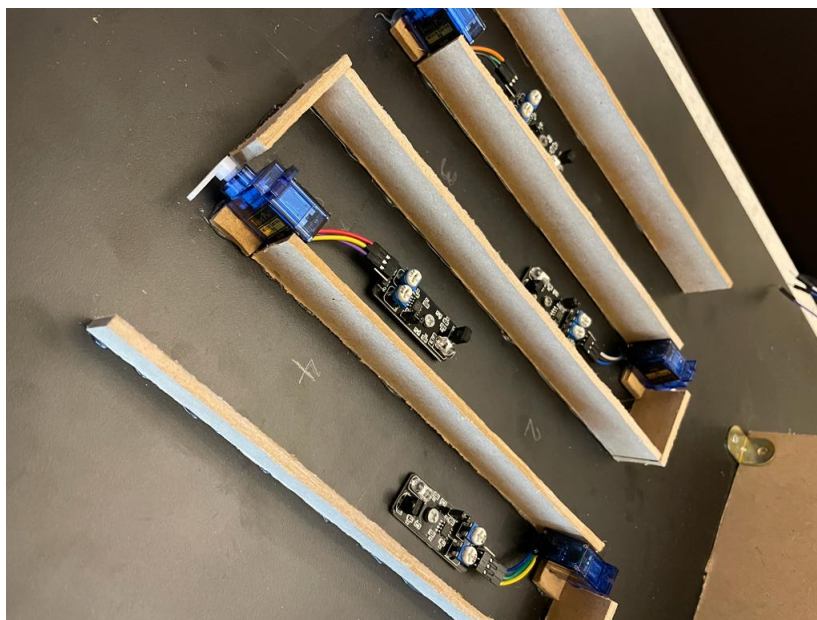


Figura 3.13: División de puestos de estacionamiento

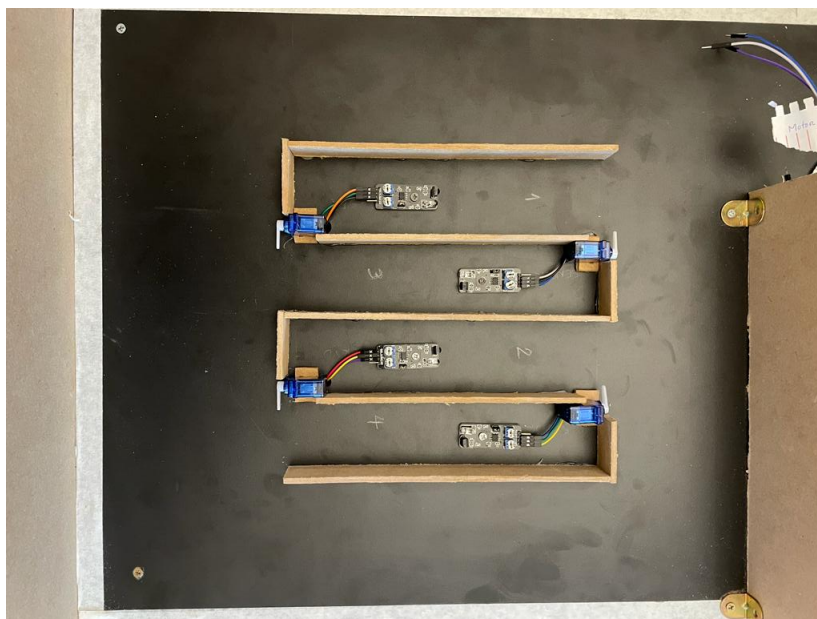


Figura 3.14: Vista Aérea - División de puestos

Capítulo 4

Conclusiones

Como cierre de nuestro proyecto queremos destacar aspectos importantes que nos ayudaron a lograr el objetivo final.

En un inicio nos dividimos el trabajo en pequeñas tareas y asignamos a cada uno la ejecución de las mismas. Eso nos permitió que cada uno se especializara en temas particulares y el resultado fuera mejor de lo esperado.

Desarrollamos un sistema que cumple con los objetivos planteados e implementamos una solución útil para los clientes de IoT-Parking.

Estamos confiados que la solución marca una diferencia ante otras soluciones presentes actualmente ya que nos destacamos en los siguientes aspectos:

- **Página web:** Sin dudas la página web juega un papel importante debido a que te permite reservar con antelación tu estancia en el estacionamiento.
- **Puesto reservado:** Sabes que si reservas un lugar, lo tendrás disponible para ti gracias al sistema de barreras con que cuenta cada lugar.

Cuando se inicia un proyecto se hace una planificación que nos sirve como guía y disminuye los problemas en gran medida. Sin embargo, como cada proyecto, tuvimos imprevistos que ocasionaron cierto estrés en el equipo y nos hizo destinar tiempo con el cual no contábamos en su resolución. Pero lo más importante a señalar en cuanto a oportunidades de mejoras son:

- Priorizar las tareas.
- Tratar de cumplir con los plazos establecidos.
- Mantener la calma en situaciones de estrés.

Capítulo 5

Trabajo Futuro

Durante el desarrollo del proyecto vimos oportunidades de mejora que sin duda harían el proyecto más atractivo. Por cuestiones de tiempo principalmente no lo implementamos pero podría servir de utilidad para proyectos futuros. Estas son algunas de las ideas que recomendamos:

- Diseño de una aplicación móvil para Smartphone y Tablet.
- En el momento de reservar, no se pueden ver las reservas realizadas para el lugar que se busca reservar, esto genera que el usuario ingrese toda la información y recién cuando va a reservar se entera que ese lugar ya está reservado, la manera de solucionar esto sería mostrar las reservas realizadas para cada lugar en el momento de reservar.
- Problemas de seguridad. El código es vulnerable a inyección de SQL por lo que un usuario malicioso podría eliminar, modificar la información de usuarios, reservas y matrículas.
- La verificación del usuario ingresado se hace a través de Cookies, estas son muy vulnerables a ataques y se pueden modificar fácilmente, se debería utilizar criptografía para generar una key y guardarlo en la cookie y la base de datos.

Referencias

- D. Hattingh. (2022). *¿qué es un ecosistema iot?* <https://telecoms.adaptit.tech/es/blog/what-is-an-iot-ecosystem/>.
- Display OLED 0.96. (2022). <https://www.eneka.com.uy/componentes/optoelectronica/displays/oled-0.96-128x64-7-pines-i2c-spi-blanco-detail.html>.
- ESP32-CAM. (2022). <https://www.electronica.com.uy/robotica/tarjetas-acesorias-de-desarrollo/mdulo-esp-32-wifi-bluetooth-con-cmra-detalle.html>.
- FTP. (2022). https://es.wikipedia.org/wiki/Protocolo_de_transferencia_de_archivos.
- HTTP. (2022). https://es.wikipedia.org/wiki/Protocolo_de_transferencia_de_hipertexto.
- K.Rose, S.Eldridge, L.Chapin. (2015). *La internet de las cosas — una breve reseña*. <https://www.internetsociety.org/es/resources/doc/2015/iot-overview/>.
- LED Ws2812B. (2022). <https://www.e-ika.com/tira-de-led-rgb-ws2812b-5v-1metro-60-leds>.
- NodeMcu. (2022). <https://www.eneka.com.uy/robotica/modulos-comunicacion/mdulo-nodemcu-v3-esp8266-wifi-detail.html>.
- Sensor MK0434. (2022). <https://www.electronica.com.uy/robotica/sensores/mdulo-sensor-de-distancias-hc-sr04-7391-7538-detalle.html>.
- Servo motor SG90. (2022). <https://www.eneka.com.uy/robotica/motores/servomotores/motor-servo-micro-sg90-detail.html>.