

ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

Implementación de un Sistema Integrado para Facilitar el Pago de los
Parquímetros en las Zonas Regeneradas de Guayaquil.

PROYECTO INTEGRADOR

Previo la obtención del Título de:

Ingeniero en Telemática

Presentado por:

Nécker Iván Espinosa Feijóo

Alex Antonio Mendoza Santos

GUAYAQUIL - ECUADOR

Año: 2019

DEDICATORIA

El presente trabajo desarrollado lo dedico principalmente a Dios, a mis padres, por su amor, trabajo y sacrificio en todos estos años. A mis hermanos, a mi abuelita, a Yamilett Cano por estar siempre presentes y por el apoyo moral que me brindaron a lo largo de esta etapa de mi vida y poder lograr uno de los anhelos más deseados.

Nécker I. Espinosa F.

Este logro va dedicado a todos que brindaron su contribución para que se cumpla. A mi madre, padre, hermana, tíos y resto de familia, amigos, mi enamorada y en especial a mi abuelita Rosa que hoy en día no puede compartir de la satisfacción y felicidad que ocasiona todo esto, pero fue la mejor persona para inculcarme valores, formar mi personalidad y determinación en la vida.

Alex A. Mendoza S.

AGRADECIMIENTOS

No hace falta mencionar a todas esas personas que de una u otra manera estuvieron brindando ese apoyo necesario para poder cumplir este logro, viviremos eternamente agradecido. Gracias a todos.

DECLARACIÓN EXPRESA

“Los derechos de titularidad y explotación, nos corresponde conforme al reglamento de propiedad intelectual de la institución; *Nécker Iván Espinosa Feijóo, Alex Antonio Mendoza Santos* y damos nuestro consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual”

Nécker I. Espinosa F.

Alex A. Mendoza S.

EVALUADORES

Ph.D. Rebeca L. Estrada P.
PROFESOR DE LA MATERIA

MSc. Néstor X. Arreaga A.
PROFESOR TUTOR

ÍNDICE GENERAL

ABREVIATURAS.....	IV
ÍNDICE DE FIGURAS	V
ÍNDICE DE TABLAS.....	VI
CAPÍTULO 1.....	1
1. Introducción.....	1
1.1 Antecedentes	1
1.2 Descripción del Problema.....	2
1.3 Delimitación del problema	3
1.4 Posibles Soluciones	3
1.5 Solución propuesta.....	5
1.6 Justificación.....	6
1.7 Objetivos	6
1.7.1 Objetivo General	6
1.7.2 Objetivos Específicos.....	6
CAPÍTULO 2.....	8
2. MARCO TEÓRICO.....	8
2.1 Criterio Tecnológico.....	8
2.2 Restricciones y Alcances del Sistema.....	9
2.3 Hardware	9
2.3.1 Raspberry Pi 3 B+.....	9
2.3.2 Micro SD 32 GB (clase 10)	11
2.3.3 Cámara para Raspberry PI 5 MP.....	11
2.3.4 Sensor Ultrasonido HC-SR04	12
2.3.5 Monedero Electrónico	14
2.4 Software	15
2.4.1 Python v3.5.....	15

2.4.2	OpenCV	16
2.4.3	Raspbian.....	17
2.4.4	Pytesseract v0.2.7	18
CAPITULO 3.....		19
3.	METODOLOGÍA.....	19
3.1	Descripción General	19
3.2	Diagrama de Bloques	20
3.2.1	Descripción del Bloque Sensor.....	20
3.2.2	Descripción del Bloque Input.	21
3.2.3	Descripción del Bloque Procesador.....	22
3.2.4	Descripción del bloque Procesamiento de Información.	25
3.3	Esquema de Conexiones	25
3.4	Reportería	28
CAPÍTULO 4.....		29
4.	Pruebas y analisis de resultados.....	29
4.1	Modificaciones.....	29
4.2	Hipótesis y Variables	29
4.3	Pruebas del dispositivo.....	30
4.4	Resultados	31
CAPÍTULO 5.....		34
5.	Conclusiones y recomendaciones.....	34
5.1	Conclusiones.....	34
5.2	Recomendaciones.....	34
Referencias BIBLIOGRÁFICAS.....		36
ANEXOS		38

ABREVIATURAS

ATM: Autoridad de Tránsito Municipal

OCR: Conocimiento Óptico de Caracteres

CSI: Interfaz Series para Cámaras.

AEC: Control automático de exposición

AWB: Balance de blancos automático

ABF: Filtro de banda automático

ABLC: Calibración del nivel de negro automático

DVP: Puerto de vídeo digital

OTP: Contraseña de un solo uso

ÍNDICE DE FIGURAS

Figura 1.1 Diagrama Funcionamiento del Parquímetro	5
Figura 2.1 Placa Raspberry Pi 3 B+	10
Figura 2.2 Cámara Raspberry Pi 5 MP	12
Figura 2.3 Sensor de Ultrasonido HC-SR04.....	14
Figura 2.4 Monedero Electrónico.....	14
Figura 2.5 Monedero electrónico	15
Figura 3.1 Componentes del Sistema parquímetro automatizado simplificado	20

ÍNDICE DE TABLAS

Tabla 1.1 Cuadro comparativo Análisis de Costos	4
Tabla 1.2 Cuadro comparativo Ventajas y Desventajas	5
Tabla 4.1. Modificación implementada y definido pendiente.	29
Tabla 4.2 Pruebas realizadas.	30
Tabla 4.3 Ingresos anuales recaudado por uso de parquímetros.....	31
Tabla 5.1 Descripción de posibles mejoras para el proyecto.....	35

CAPÍTULO 1

1. INTRODUCCIÓN

1.1 Antecedentes

En países como México existe una plataforma que permite y facilita la ubicación, registro, pago y control de los parquímetros y estacionamientos. Los usuarios pueden realizar sus pagos a través de los negocios ubicados cerca de la zona o enviando un mensaje de texto; información que está vinculada con los celulares de los agentes de tránsito y les permite saber en tiempo real qué parqueos se encuentran pagados y de esta manera evitar multas por infracciones. Otro ejemplo de sistemas de parquímetros inteligentes lo posee Grecia, donde aproximadamente 260 dispositivos con sensores se encuentran operando, permitiendo detectar al vehículo y enviar una señal de radio con la información de los lugares de parqueo. Esto se refleja en una aplicación de tiempo real, la cual permite el pago del tiempo de parqueo que se desea utilizar, e incluso realizar el pago de manera remota al espacio de parqueo.

Antiguamente en la ciudad de Guayaquil no existía un sistema de parqueo, quienes se encargaban de esta labor eran personas de la zona (denominados cuidadores de carro) a cambio de dinero en efectivo; este ingreso era netamente para ellos sin destinar ganancias a la Municipalidad de la ciudad. Esta forma de parqueo, además de congestionar el tránsito de la ciudad, especialmente en localidades con mayor afluencia, ocasionaba malestar e incomodidad en conductores, incluso en transeúntes.

A partir del 2008, la Autoridad de Tránsito Municipal (ATM), en colaboración con la empresa Parqueo Positivo, instauró un sistema de parqueo en tres zonas regeneradas de la ciudad de Guayaquil, tales como: Urdesa, Alborada y Kennedy. El sistema está basado en el uso de parquímetros, en que los usuarios introducen una moneda (5, 10, 25 centavos o \$1) en el dispositivo de parqueo luego de presionar un botón (izquierda o derecha según se estacione). El horario de uso es de lunes a viernes de 08:00 hasta las 20:00 con un tiempo máximo de 4 horas diarias por espacio de estacionamiento.(El Universo, 2019)

Si los conductores exceden el tiempo límite de parqueo o no depositan el valor en el parquímetro, se les genera una multa equivalente al 15% del salario básico unificado (El Universo, 2018). Esta sanción es controlada por la ATM a través de agentes de tránsito, quienes circulan en bicicleta por las zonas antes mencionadas para identificar los vehículos a los que se les debe emitir la multa.

Fernando Navas, jefe de planificación de la ATM, expuso que uno de los principales beneficios que plantea el proyecto es evitar el doble parqueo o la doble columna. En cuanto a la percepción de este sistema de parqueo, Andrade, residente de Urdesa, lo considera una alternativa que beneficia a los moradores que deben esperar algunas horas para poder ingresar sus vehículos a los garajes (como se cita en El Universo, 2008).

En la actualidad, esta idea de cobro por estacionamiento en la vía pública está tomando fuerza con la finalidad de regular este inconveniente. En ciudades como Machala trabajan con la empresa Parqueo Positivo (Parqueo Positivo, 2003); mientras que Zaruma (EMOVTZA, 2013) y Piñas (GAD MUNICIPAL DE PIÑAS, 2015) utilizan el sistema de parqueo proporcionado por la empresa pública EMOVTZA. Sin embargo, los recursos destinados a esta actividad, tales como: parquímetros, personal de trabajo, transporte, entre otros, conllevan a una gran inversión para el cumplimiento del plan.

1.2 Descripción del Problema

Actualmente el proceso con el cual la empresa Parqueo Positivo y la Autoridad de Tránsito Municipal (ATM) identifican los vehículos para poder generar las multas no es el más idóneo. Al ser los agentes de tránsito quienes realizan los recorridos para identificar los vehículos sancionados y emitir las multas manualmente, la actividad no se cumple oportunamente; debido a que el personal de trabajo no es suficiente para inspeccionar estos tres sectores regenerados con más de 500 parquímetros. (El Telégrafo, 2019)

Por otra parte, si el proyecto se expande en otras zonas de la ciudad, el personal, así como sus transportes (bicicletas) deben aumentar; lo que repercute en una elevación de costos para la ATM.

En este contexto, la Autoridad de Tránsito Municipal requiere de un acompañamiento técnico para reformar el control de su actual sistema de parqueo; esto con el diseño y uso de herramientas tecnológicas que conviertan su sistema en un modelo eficiente y eficaz.

1.3 Delimitación del problema

Luego de un análisis realizado en los sectores regenerados (Urdesa, Alborada y Kennedy) en donde se evidenció la gran cantidad de dispositivos instalados (aproximadamente 500), el proyecto se delimita en el Sector Alborada en las calles: Av. Rodolfo Baquerizo Nazur Crotos 090507m, exactamente en los parquímetros ubicados al lado este del parque Gran Albocentro.

1.4 Posibles Soluciones

Después de analizar varias ideas que resuelvan la problemática planteada y que brinden un mejor impacto en relación con el esfuerzo aplicado, se describen las siguientes posibles soluciones:

- 1) Elaboración de un dispositivo general que controle el tiempo y permita el pago por el uso del parqueo por segmento de calle, con determinado rango de parquímetros, que permita al usuario introducir un indicador del lugar y tiempo que va a estacionar su vehículo, y efectuar el pago inmediatamente. Además, este dispositivo generará una notificación a los agentes de tránsito para que se realice la multa correspondiente en caso de que se venza el tiempo de parqueo.
- 2) Elaboración de un dispositivo por puesto de parqueo que controle el tiempo en el que se estaciona el usuario, y le permita generar el pago acorde al tiempo de uso. Si el tiempo límite es excedido, se genera la multa de manera automática por medio de una cámara fotográfica acoplada, que captura la imagen y aplica un proceso OCR (Reconocimiento Óptico de Caracteres) para obtener los datos de la placa del vehículo, ubicado en el lugar de parqueo. El pago de la multa será realizado de acuerdo a los trámites establecidos por la ATM.

- 3) Desarrollo de una aplicación móvil que permita al agente de tránsito visualizar la ubicación de los parquímetros que muestran infracciones; y el diseño y construcción de un sistema embebido que contenga un sensor de movimiento que notifique al aplicativo móvil el uso del parqueo que no ha sido pagado.

A su vez, se realizó dos cuadros comparativos: análisis de costos Tabla 1.1 y ventajas-desventajas Tabla 1.2 de las tres soluciones anteriormente expuestas, con el fin de optar por la mejor solución.

Tabla 1.1 Cuadro comparativo Análisis de Costos (Elaboración propia)

Solución 1	Solución 2	Solución 3
Esta solución es muy costosa, ya que al ser un dispositivo general debe ser robusto tanto en hardware como en software, sensores en cada uno de los estacionamientos y sensor en la máquina para poder reconocer el vehículo entrante o saliente. El costo de un dispositivo con estas funcionalidades puede estar entre \$5000 a \$10000, a estos valores hay que sumarle el costo de infraestructura de los sistemas a los que están vinculados para su funcionamiento.	Es una versión mini de la solución 1, lo que es menos costosa debido a que no serían tan robustos a nivel de hardware o software. Teniendo un costo aproximado de elaboración e implementación de cada dispositivo de \$800.	Esta solución requiere de una alta inversión en software, ya sea licencias, apis de Google, además de costos de personal. Teniendo en cuenta el desarrollo de un aplicativo móvil que tenga un soporte para aproximadamente de 1200 usuarios haciendo uso de este, se tiene un costo de inversión de aproximadamente \$10.000 fuera de costos de mantenimiento.

Tabla 1.2 Cuadro comparativo Ventajas y Desventajas (Elaboración propia)

Soluciones	Ventaja	Desventaja
Solución 1	Un solo dispositivo para una gran cantidad de estacionamientos.	Alto costo para su implementación.
Solución 2	Mejor gestión y control de los espacios de estacionamiento.	Un dispositivo por espacio de estacionamiento.
Solución 3	Mejor gestión y control de los espacios de estacionamiento.	Similar al actual sistema de generación de multas.

1.5 Solución propuesta

Con base al análisis de costos y las ventajas-desventajas de las ideas antes propuestas se ha determinado que el proyecto, el cual se enfoca en la automatización del proceso de generación de multas por incumplimiento de las reglas de los parquímetros, desarrollará la opción 2 “Elaboración de un dispositivo por puesto de parqueo que controle el tiempo en el que se estaciona el usuario, le permita el pago acorde al tiempo de uso y le genere la multa de manera automática por medio de una cámara fotográfica acoplada”, esto debido a que tiene un mayor impacto acorde a las necesidades de la problemática. En la figura 1.1 se muestra el diagrama de funcionamiento del parquímetro y el proceso para poder generar la multa.

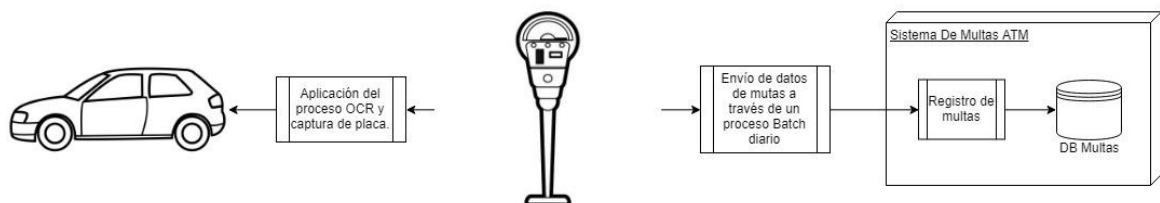


Figura 1.1 Diagrama Funcionamiento del Parquímetro (Elaboración propia)

1.6 Justificación

La ATM al implementar este dispositivo en su sistema de parqueo, evitará el requerimiento de agentes de tránsito para estar monitoreando constantemente los parquímetros, además de que el registro de la multa interactúe con su sistema para poder realizar un posterior pago.

En cuanto a beneficios tecnológicos, el uso de una placa de prototipado como la Raspberry Pi permitirá embeber todos los componentes físicos que se requiere para brindar la funcionalidad del parquímetro, tales como: cámara fotográfica, sensores de movimiento y monedero electrónico. Además, el uso de la librería de código abierto OpenCV permitirá realizar el proceso de OCR dentro de un sistema embebido, desarrollado a través del lenguaje de programación Python que brinda gran soporte sobre la manipulación digital de imágenes.

1.7 Objetivos

1.7.1 Objetivo General

Diseñar un parquímetro automatizado que permita al usuario el pago por tiempo de parqueo y la generación automática de la multa en caso de exceder el límite establecido.

1.7.2 Objetivos Específicos

1. Implementar un proceso OCR mediante la captura de una foto y el uso de librerías OpenCV para la identificación de los caracteres de la placa del vehículo.
2. Crear un sistema embebido acoplando todos los componentes (sensor de movimiento, cámara fotográfica y monedero electrónico) con la placa de prototipado Raspberry Pi 3 B+ para la automatización de las multas.
3. Realizar la conexión con el sistema de multas a través de microservicios que permitan el registro de la infracción, correspondiente a una base de datos

CAPÍTULO 2

2. MARCO TEÓRICO

2.1 Criterio Tecnológico

El sistema operativo de la herramienta de prototipado Raspberry Pi 3 B+ se denomina Raspbian que es una distribución de Linux. Esta distribución se trata de una versión de Debian modificada y optimizada para sacar el máximo partido de este micrordenador. (Velasco, 2019).

Raspbian se utiliza igual que Debian o que la mayoría de las distribuciones Linux populares; es muy sencilla, estable y rápida (Velasco, 2019). El presente proyecto, utiliza la Raspberry Pi 3 como servidor para ejecutar microservicios necesarios, capturar imagen y realizar conexión a un servidor de base de datos para generar las multas de manera automatizada; por lo que el sistema operativo se utiliza en modo consola, debido a que no se utilizan herramientas que requieran ser gestionadas desde la interfaz gráfica. De esta manera, se puede optimizar el consumo de recursos y evitar problemas al arranque de la Raspberry Pi 3.

Un punto importante en este proyecto es la seguridad, dado que se maneja información delicada para el cliente. Por ello, se consideran algunos aspectos como: realizar conexión VPN con la red del cliente para tener mayor seguridad y cambiar periódicamente la clave de acceso al sistema operativo, realizando una conexión diaria para hacer el registro de las multas al sistema de la ATM.

Para el desarrollo de los microservicios se utiliza Python como lenguaje de programación por su sencillez, legibilidad y precisión de sintaxis. Por lo tanto, en pocas líneas te permite programar algoritmos bastante complejos; es práctico en cuanto al uso del código. Además, Python es un lenguaje de alto nivel, lo que implica que no perderse entre teorías de programación y dedicarse a lo que realmente se espera: mecanizar ciertos procesos de analítica web. (Click Formación 360, 2018)

2.2 Restricciones y Alcances del Sistema

Existen limitantes dentro del desarrollo del sistema más allá del ámbito tecnológico, y esto hace definir alcances para tener indicadores que permitan evaluar el dispositivo final a entregar, tales como:

- La alimentación energética que usa actualmente los parquímetros para poder conectar el dispositivo que va a realizar el proceso OCR.
- Selección del lenguaje de programación con el cual se van a desarrollar los microservicios.
- No es posible la conexión con la base de datos de la ATM, por ello se ha optado por una base de datos de desarrollo.
- Análisis de la información del cliente para el tiempo límite de estacionamiento por vehículo.
- Análisis de la información del cliente para el tiempo de envío de notificación a los usuarios finales cuando el tiempo reservado esté por finalizar.

2.3 Hardware

2.3.1 Raspberry Pi 3 B+

Es un micro-ordenador con un buen rendimiento, cuenta con CPU acelerada, soporte WiFi de doble banda de frecuencia y soporte para Gigabit Ethernet. Dentro de la placa encontramos diversos circuitos, chips y conectores que permiten la integración de sus componentes para brindar una mayor funcionalidad. (Raspberry Pi Foundation).

Características de la tarjeta Raspberry Pi 3 B+:

- Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @1.4GHz.
- 1GB LPDDR2 SDRAM.
- 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE.
- Ethernet Gigabit sobre USB 2.0 (rendimiento máximo de 300 Mbps).
- Cabecera GPIO de 40 pines extendida.
- HDMI de tamaño completo.
- 4 puertos USB 2.0.

- Puerto de cámara CSI para conectar una cámara Raspberry Pi.
- Puerto de pantalla DSI para conectar una pantalla táctil de Raspberry Pi.
- Salida estéreo de 4 polos y puerto de video compuesto.
- Puerto micro SD para cargar su sistema operativo y almacenar datos.
- Entrada de corriente continua de 5V / 2.5A.
- Soporte de Power-over-Ethernet (PoE) (requiere PoE HAT por separado).

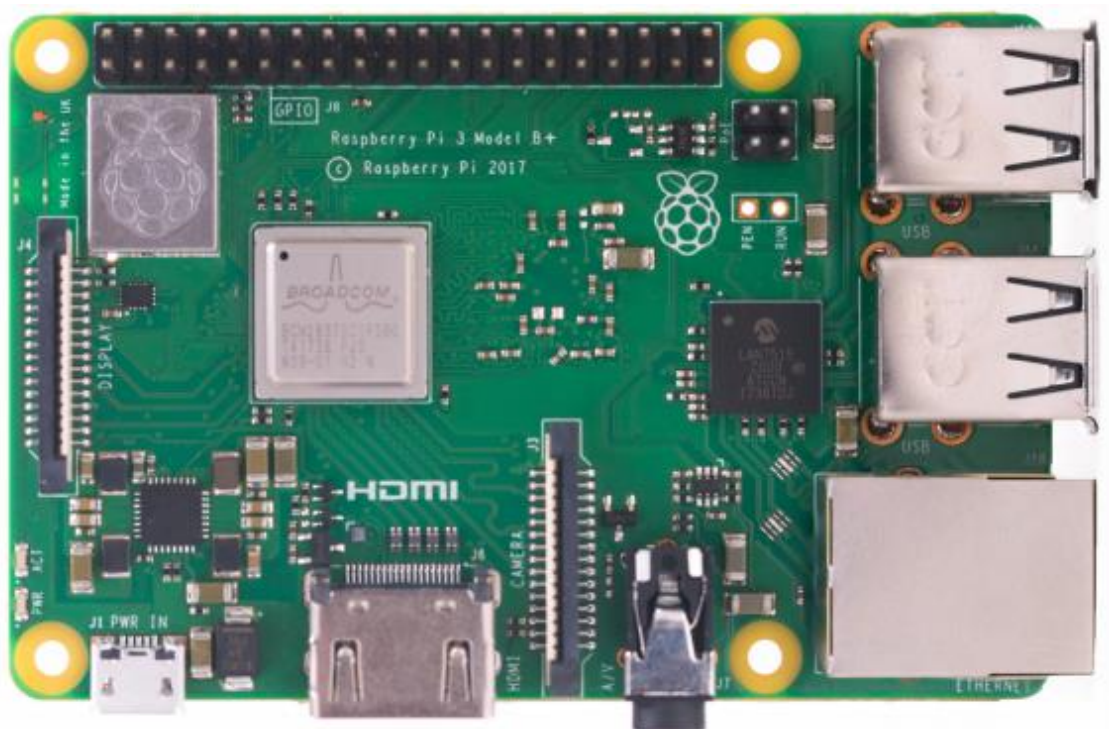


Figura 2.1 Placa Raspberry Pi 3 B+

2.3.2 Micro SD 32 GB (clase 10)

Micro SD es un formato para tarjetas de memorias *flash* para el almacenamiento de archivos digitales en dispositivos electrónicos; utilizada generalmente en teléfonos móviles, consola de videojuegos, unidades de memoria USB, etc. (Kingston Technology, 2019)

Características técnicas:

- Pesa alrededor de 0.258 gramos.
- Voltaje de funcionamiento 3.3 V para SDSC y 1.8/3.3 V para micro SDHC (Secure Digital High Capacity) y micro SDXC (Secure Digital Extended Capacity).
- No posee interruptor de protección contra escritura como en las tarjetas SD.
- 8 pines.
- Soporte para DRM.
- Velocidad de lectura/escritura 10 MB/s.

2.3.3 Cámara para Raspberry PI 5 MP

El módulo cámara de 5 megapíxeles está diseñado para el uso en la Raspberry PI, que dispone de lente fijo. Este módulo es capaz de capturar imágenes estáticas con resolución de 2592 x 1944, y para formato de video cuenta con resoluciones de 1080p30, 720p60 y 640x480p60/90. La conexión de la cámara a la placa de Raspberry PI se realiza mediante una interfaz dedicada CSI (Camera Serial Interface), diseñada especialmente para conexión de cámaras, que se encuentra ubicado en la parte superior de la tarjeta.



Figura 2.2 Cámara Raspberry Pi 5 MP

Características técnicas:

- 1,4 μm X 1,4 μm píxeles con tecnología OmniBSI de alto rendimiento (alta sensibilidad, baja diafonía, ruido bajo).
- Tamaño óptico de 1/4 ".
- Funciones de control de imagen automáticas, tales como:
 - Control automático de exposición (AEC).
 - Balance de blancos automático (AWB).
 - Filtro de banda automático (ABF).
 - Calibración del nivel de negro automático (ABLC).
- Controles programables para la velocidad de fotogramas, AEC / AGC 16 zonas / posición / control de peso, espejo y lado, recorte, ventanas, y el panorama.
- Puerto de vídeo digital (DVP) Interfaz de salida en paralelo.
- 32 bytes de memoria programable una sola vez incorporada (OTP).

2.3.4 Sensor Ultrasonido HC-SR04

El módulo HC-SR04 es un sensor de distancias por ultrasonido capaz de detectar objetos y calcular la distancia a la que se encuentra, en un rango de 2 a 450 cm. Es de fácil uso y programación con las placas de Arduino y microcontroladores. (ElectroniLab, 2018)

El funcionamiento es simple, envía una señal ultrasónica inaudible y entrega el tiempo en que demora ir y venir hasta el obstáculo más cercano que detecta. Generalmente están conformados por dos cilindros, puestos uno al lado del otro. Uno de ellos es quien emite la señal ultrasónica, mientras que el otro es quien la recibe. Es un sistema muy simple, pero efectivo.

El sensor HC-SR04 en particular, tiene una sensibilidad muy buena del orden de los 3mm., considerando que la mayoría de las aplicaciones donde este sensor utilizado para medir, detectar obstáculos o distancias mayores a varios centímetros resulta efectiva. (Veloso, 2016).

Características técnicas:

- Dimensiones del circuito: 43 x 20 x 17 mm
- Tensión de alimentación: 5 Vcc
- Frecuencia de trabajo: 40 KHz
- Rango máximo: 4.5 m
- Rango mínimo: 1.7 cm
- Duración mínima del pulso de disparo (nivel TTL): 10 μ S.
- Duración del pulso eco de salida (nivel TTL): 100-25000 μ S.
- Tiempo mínimo de espera entre una medida y el inicio de otra 20 mS.

Pines de Conexión:

- VCC
- Trig (Disparo del ultrasonido)
- Echo (Recepción del ultrasonido)
- GND

Fórmula para calcular la distancia:

$$\text{Distancia} = \{(\text{Tiempo entre Trig y el Echo}) * (\text{V.Sonido } 340 \text{ m/s})\}/2$$



Figura 2.3 Sensor de Ultrasonido HC-SR04

2.3.5 Monedero Electrónico

Dispositivo electrónico que permite introducir monedas a través de una ranura e identificarlas.

Características técnicas:

- Identifica 1-6 Monedas diferentes
- Tasa de precisión: 95%
- Monedas de aplicación:
 - Diámetro: 15mm-29mm
 - Grosor: 1.8mm-3.0mm
- Aplicable a todo tipo de monedas y monedas extranjeras.
- Presión atmosférica: 80Kpa-106Kpa
- Voltaje de funcionamiento: $DC12V \pm 10\%$
- Corriente de trabajo: 50mA
- Corriente máxima de momento: 350mA (menos de 0.5S)
- Humedad: $\approx 95\%$
- Temperatura de almacenamiento: $-20 - + 85$
- Temperatura de funcionamiento: $-15 - + 60$
- Velocidad de reconocimiento: $\approx 0.6\text{sec}$



Figura 2.5 Monedero electrónico

2.4 Software

2.4.1 Python v3.5

Python es un lenguaje de programación de propósito general muy popular, debido a su simplicidad y legibilidad de código permite al programador expresar sus ideas en menos líneas de código sin reducir la legibilidad.

En comparación con otros lenguajes como C / C ++, Python es más lento, sin embargo, puede extender fácilmente con C / C ++. Esta función ayuda a escribir códigos de computación intensiva en C / C ++ y crear un envoltorio de Python para usar estos envoltorios como módulos de este lenguaje. Esto nos brinda dos ventajas: primero, nuestro código es tan rápido como el código original de C / C ++ (ya que es el código real de C ++ que funciona en segundo plano) y, segundo, es muy fácil de codificar en Python. (Mordvintsev & K., 2013).

Fue desarrollado cinco años antes de la existencia del preprocesador-hipertexto. El lenguaje fue creado y desarrollado por Guido Van Rossum en 1989. Desde el año 2000, su popularidad entre los desarrolladores creció rápidamente gracias a su compatibilidad con Unicode. Además, Python no solo se creó para el desarrollo web, sino que también es compatible con el desarrollo completo, Python es

expuesto y asociado a casi todas las tecnologías (Education IT, 2017), lo que permite su funcionamiento con Raspberry Pi 3 B+.

2.4.2 OpenCV

OpenCV se inició en Intel en 1999 por Gary Bradsky y el primer lanzamiento salió en 2000. Vadim Pisarevsky se unió a Gary Bradsky para administrar el equipo ruso OpenCV del software de Intel. En 2005, OpenCV se utilizó en Stanley, el vehículo que ganó el Gran Desafío DARPA 2005. Más tarde, su desarrollo activo continuó bajo el apoyo de Willow Garage, con Gary Bradsky y Vadim Pisarevsky liderando el proyecto. En este momento, OpenCV admite una gran cantidad de algoritmos relacionados con la Visión por Computador y el Aprendizaje Automático y se está expandiendo día a día.

Actualmente, OpenCV admite una amplia variedad de lenguajes de programación como C ++, Python, Java, etc. y está disponible en diferentes plataformas, incluyendo Windows, Linux, OS X, Android, iOS, etc. Además, las interfaces basadas en CUDA y OpenCL también están en desarrollo activo para velocidad de las operaciones de GPU.

OpenCV-Python es la API de Python de OpenCV. Combina las mejores cualidades de OpenCV C ++ API y el lenguaje Python. (Mordvintsev & K., 2013).

Sus principales funciones son:

- Captura en tiempo real.
- Importación de archivos de vídeo.
- El tratamiento básico de imágenes (brillo, contraste, umbral).
- Detección de objetos (cara, cuerpo).
- Blob detección.

Sus aplicaciones son:

- OpenCV ha sido usado en el sistema de visión del vehículo no tripulado Stanley de la Universidad de Stanford, el ganador en el año 2005 del Gran desafío DARPA.
- OpenCV se usa en sistemas de vigilancia de vídeo.
- OpenCV es la clave en el programa Swistrack, una herramienta de seguimiento distribuida.

Cabe destacar que la última versión de OPENCV es compatible con la última versión de Python. Esto permite aprovechar todas las ventajas de la última versión de este lenguaje.

Quizás sea la biblioteca de visión artificial más importante y usada del mundo. Es utilizada por universidades, empresas y gente del movimiento Maker para dar rienda suelta a su imaginación al tratarse de un software libre. (Hernández, 2018)

Actualmente se proporcionan compilaciones para las siguientes versiones de Python: 2.7, 3.4, 3.5, 3.6 y 3.7.

2.4.3 Raspbian

Raspbian es el sistema operativo recomendado para Raspberry Pi (al estar optimizado para su hardware) y se basa en una distribución de GNU/Linux llamada Debian.

Para instalar Raspbian en nuestra Raspberry Pi se dispone de dos versiones; una más completa con entorno gráfico y otra más reducida sin entorno gráfico:

- Raspbian Píxel: Versión completa con entorno gráfico de Raspbian, es decir, la versión de escritorio con menús, ventanas, iconos, fondos de pantalla, etc., utilizado por la mayoría de los usuarios como ordenador de sobremesa.
- Raspbian Lite: Versión reducida sin entorno gráfico, es decir, la versión en modo consola sin gráficos. Esta opción generalmente es para usuarios avanzados con conocimientos de Linux que utilizan la Raspberry Pi como servidor. (Abellán, 2018)

2.4.4 Pytesseract v0.2.7

Es una herramienta OCR para el lenguaje de programación Python. Esta librería permite leer y reconocer texto incrustado en imágenes.

Python-Tesseract es una envoltura para el motor Tesseract-OCR de Google. También es útil como un script de invocación independiente para Tesseract, debido a que se pueden leer todas las imágenes compatibles con Python Imaging Library, incluidos ficheros de extensión jpeg (Joint Photographic Experts Group), png (Portable Network Graphics), gif (Graphics Interchange Format), bmp (Bits Maps Protocole), tiff (Tagged Image File Format) y otros, mientras que Tesseract-OCR por defecto solo admite imágenes de extensión tiff y bmp. (Python Software Foundation, 2019)

Prerrequisitos:

- Python 2.5.x o 3.x.
- Se necesita la librería Python Imaging Library (PIL o Pillow)

CAPITULO 3

3. METODOLOGÍA

3.1 Descripción General

Este capítulo presenta el método en que el cliente (ATM) podrá obtener la información necesaria de las personas que hayan sobrepasado el tiempo límite de estacionamiento comprado. Esta información hace referencia a la placa del vehículo, hora en que inició y terminó la infracción, y consecuentemente el tiempo que sobrepasó con relación al que adquirió. Todos estos datos son necesarios para generar la multa.

Para tener una idea clara del proyecto se describe el funcionamiento del sistema de parquímetro automatizado mediante diagrama de bloques y la ubicación de los sensores dentro del sistema incluyendo sus conexiones.

El proyecto consta de dos partes esenciales: la parte electrónica y la parte lógica. Dentro de la parte electrónica tenemos 3 partes importantes que son el núcleo del proyecto:

- Sensor de Ultrasonido: Este sensor detectará si un vehículo está cerca del sitio de estacionamiento, luego de esto se encenderá la cámara.
- Cámara: Para poder obtener la placa del vehículo y así poder realizar el proceso de reconocimiento de caracteres.
- Monedero electrónico: Para controlar el tiempo de estacionamiento de acuerdo con la cantidad de dinero ingresado.

Dentro de la parte lógica tenemos las siguientes estructuras:

- Reconocimiento de caracteres: De la captura de imagen obtenida con la cámara se procede a obtener la placa del vehículo en forma de texto.
- Enviar de información: De acuerdo con el cliente la información será enviada en un archivo Excel al finalizar el día.

- Reportaría: Una página donde el administrador puede verificar los datos de los vehículos obtenidos mediante el proceso de OCR, estos datos pueden ser de mucha ayuda ya que al analizarlos se puede obtener estimaciones del avance y productividad del proyecto.

3.2 Diagrama de Bloques

La figura 3.1 muestra un diagrama general de cómo es el proceso para obtener la información del infractor a través del bloque sensor y bloque *input* y de esta manera enviar la información a la ATM.

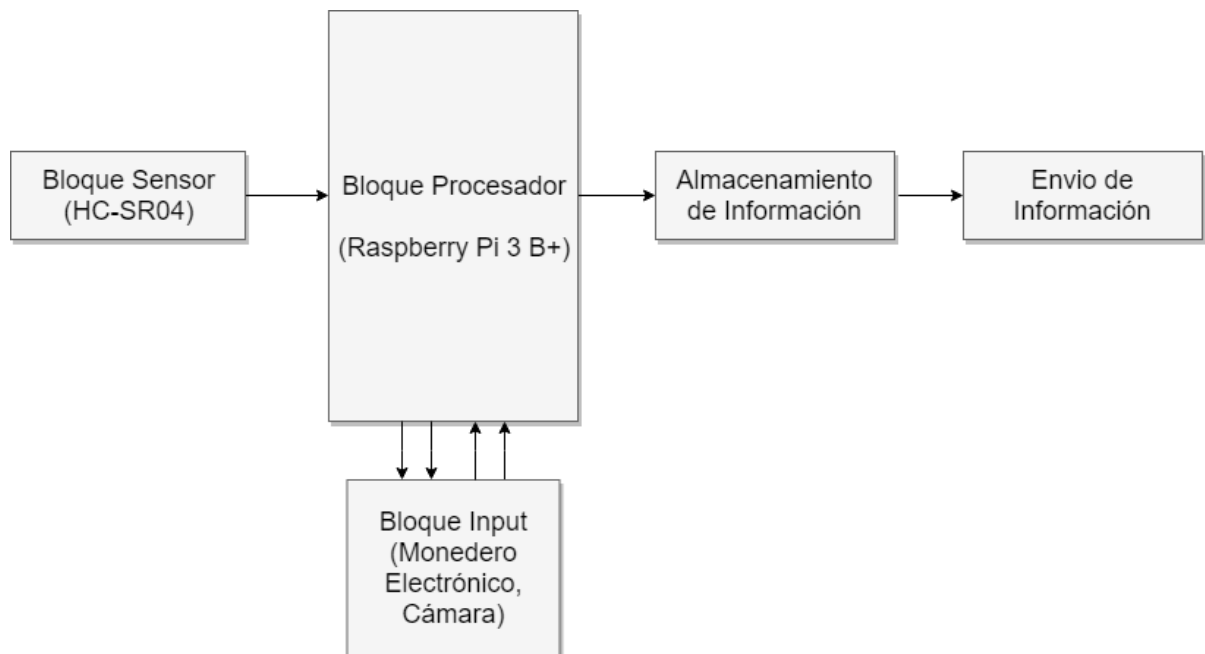


Figura 3.1 Componentes del Sistema parquímetro automatizado simplificado (Elaboración propia)

En esta sección se presentará la descripción de cada bloque anterior y la relación que existe con los otros bloques que conforman el sistema parquímetro automatizado.

3.2.1 Descripción del Bloque Sensor.

En este bloque se utiliza el sensor de ultrasonido HC-SR04, el cual sirve para identificar si un vehículo se encuentra cerca del parquímetro automatizado con base en las siguientes características:

- El sensor tomará el tiempo en que tarda en llegar la longitud de onda entre el parquímetro y el obstáculo denominado tiempo inicial.
- El sensor tomará el tiempo en que tarda en llegar la longitud de onda entre el obstáculo y el parquímetro denominado tiempo final.
- Se aplica la fórmula para obtener la distancia en la que se encuentra el objeto.

$$(tiempo_{final} - tiempo_{inicial}) * 10^6 / 58 .$$

Ecuación 3.2.1. Ecuación para obtener la distancia entre el objeto y el sensor

- El sensor siempre estará censando la distancia entre el parquímetro y un objeto.

Para la recepción de los datos censados se conecta el sensor HC-SR04 a la Raspberry Pi 3+ para manipular la información y establecer los umbrales en el procesador a nivel de código.

3.2.2 Descripción del Bloque Input.

En este punto se describe los componentes que conforman el bloque input y sus conexiones con los otros bloques.

3.2.2.1 Monedero Electrónico



Figura 3.2 Asignación del tamaño de moneda en el monedero electrónico.

La función principal del monedero electrónico es permitir el ingreso de dinero para alquilar el espacio de estacionamiento por un determinado tiempo.

Para fines de demostración se adquirió un monedero electrónico ajustable a un solo modelo de moneda, el cual se encuentra ubicado en la parte superior del monedero, en la figura 3.2 encontramos un ejemplo de cómo colocar la moneda..

Si se ingresa una moneda que no pertenece al modelo antes especificado, el monedero electrónico la rechazará, mientras que si la moneda que se ingresa pertenece al modelo especificado pasará por el sensor óptico y enviará una señal de alto a la Raspberry Pi B+ 3, la cual será interpretada por el bloque procesador y poder establecer el tiempo que ha adquirido el usuario.

3.2.2.2 Cámara Raspberry Pi

La función principal de la cámara es poder capturar la imagen de la placa del vehículo, lo cual se logra a través de una función desarrollada en el bloque procesador.

Para realizar lo anterior se usa el módulo de cámara de la Raspberry Pi 3 B +, el cual va conectado al puerto J3 de la Raspberry Pi. 3 B+.

Cabe mencionar que para utilizar el módulo de cámara de la Raspberry Pi 3 B+ se necesita configurar las opciones de Interfaz en el menú de configuración del microprocesador.

3.2.3 Descripción del Bloque Procesador.

El bloque procesador es el núcleo del proyecto, en él se centra toda la lógica (se la puede observar en la figura 3.3) desarrollada para obtener la información de los bloques anteriores, manipular y dar paso al siguiente bloque que es el proceso de información.

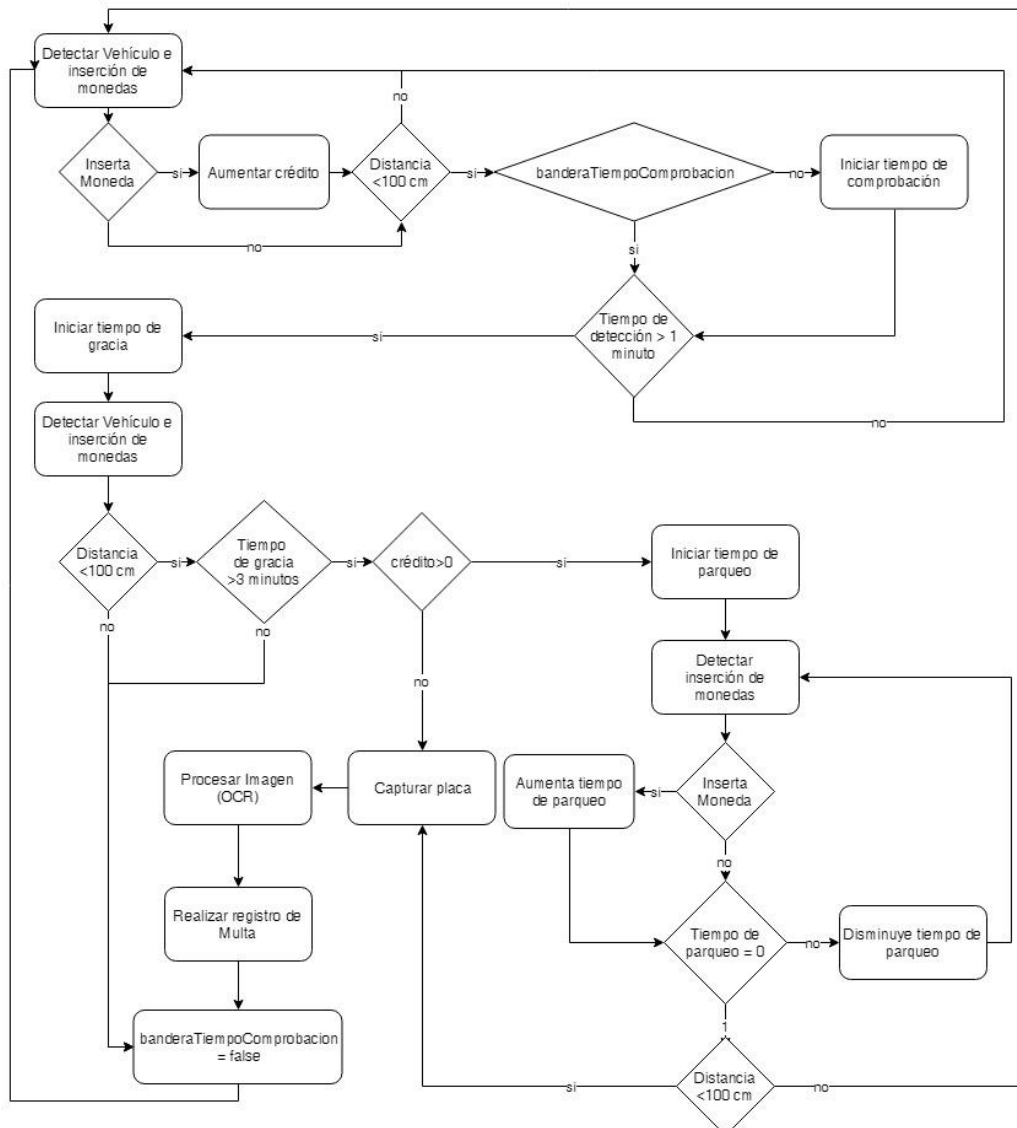


Figura 3.2.3 Diagrama de flujo funcional del parquímetro

La lógica desarrollada dentro del bloque procesador se detalla a continuación:

- **Detectar Obstáculo:** Su rol es detectar la distancia a través del sensor HC-SR04 entre el parquímetro automatizado y un objeto.
- **Captura Placa:** Su rol es encender la cámara, tomar una fotografía de la placa del vehículo y procesar la imagen para obtener la placa en texto plano.
- **Control Tiempo Gracia:** Su rol es contar el tiempo en que el usuario se demora en estacionar el vehículo e insertar una moneda en el parquímetro.

Para establecer el tiempo de gracia el sensor HC-SR04 debe detectar un objeto en una distancia menor a 100 cm, una vez detectado el vehículo se

dará un tiempo límite de 3 minutos para que el usuario pueda insertar el dinero en el parquímetro automatizado.

Si dentro de los 3 minutos el parquímetro detecta que ingresó dinero automáticamente, se procederá a establecer el tiempo de parqueo alquilado por el usuario. Si pasado los 3 minutos el usuario no ingresa dinero en el parquímetro, automáticamente hará referencia a la función *CapturaPlaca()* para proceder con el registro de datos suficientes, enviarlos a la ATM y generar la multa.

- **Control Tiempo Parqueo:** Su rol es determinar la cantidad de monedas que se han ingresado en el parquímetro automatizado, y de acuerdo a las políticas establecidas por el cliente, proceder a determinar el tiempo alquilado por el usuario para estacionar su vehículo en el parqueo.

Si el tiempo de parqueo finaliza, el vehículo sigue estacionado y el usuario no ha ingresado más dinero, automáticamente se hace referencia a la función *CapturaPlaca()* para obtener los datos de la placa del vehículo, registrar los datos en el archivo de Excel y así poder enviarlos a la ATM y generar la multa.

- **crearDocumentoMultas:** Su rol es generar un archivo de Excel en donde se guarde la información necesaria para que la ATM pueda generar la multa. Los datos que se registran en el archivo son los siguientes:
 - **ID_DISPOSITIVO:** El id con el que estará referenciado el parquímetro automatizado.
 - **PLACA:** Es la placa del vehículo que realizó la infracción.
 - **FECHA:** Se registra la fecha en que se realizó la infracción.
 - **HORA:** Se registra la hora en que se realizó la infracción.
- **registraMulta:** Su rol es el de registrar los datos del vehículo y fecha en el archivo de Excel que se genera al ejecutar la función *crearDocumentoMultas()*.
- **conexiónDB:** Esta función se encarga de registrar la información obtenida de los vehículos que sobrepasen el tiempo de límite adquirido en una base de

datos, para posteriormente acceder a dicha información y poder mostrarla en una página web.

En resumen, la función del bloque procesador es el de poder obtener la información que se obtiene del bloque sensor y del bloque input para poder establecer los límites de tiempo, límites de distancia y poder ejecutar el proceso OCR para reconocimiento de caracteres.

3.2.4 Descripción del bloque Procesamiento de Información.

En el bloque de procesamiento de Información se trata de la manipulación de los datos necesarios para que la ATM genere la multa.

3.2.4.1 *Almacenar Información.*

La función de este sub-bloque es el de poder registrar los datos de las diferentes infracciones a lo largo del día en un archivo de Excel.

3.2.4.2 *Enviar Información.*

Su función es la de enviar el archivo de texto plano con los detalles de las multas generadas a diario mediante correo electrónico a la ATM para que puedan registrar las multas con los datos obtenidos en el parquímetro.

Cabe recalcar que la lógica que engloba este sub-bloque se realiza dentro del bloque procesador.

3.3 Esquema de Conexiones

Se presentará el circuito de los sensores con la Raspberry Pi 3 B+.

Se especifica las conexiones de los sensores con la Raspberry Pi 3 B+ en las siguientes tablas:

- Para el sensor de movimiento se describe pines de conexión en la Tabla 3.1.

Tabla 3.1 Conexiones con el Sensor HC-SR04

Pines Sensor de Movimiento	Pines Raspberry Pi
VCC	2(VCC 5v)
ECHO	7
TRIGGER	11
GND	4(GND)

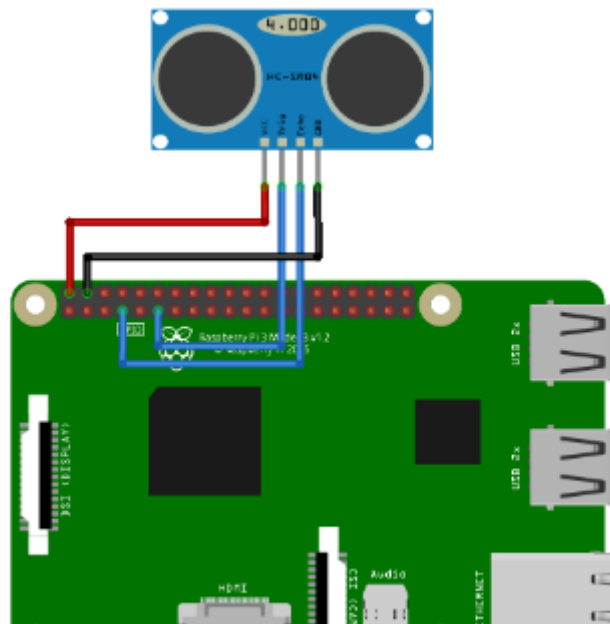


Figura 3.3.1 Conexión HC-SR04 y Raspberry Pi 3+

- Para la cámara se describe la conexión en la Tabla 3.2:

Tabla 3.2 Conexiones con los sensores de la cámara

Pines cámara	Pines Raspberry Pi
Bus de datos	CMIO Camera

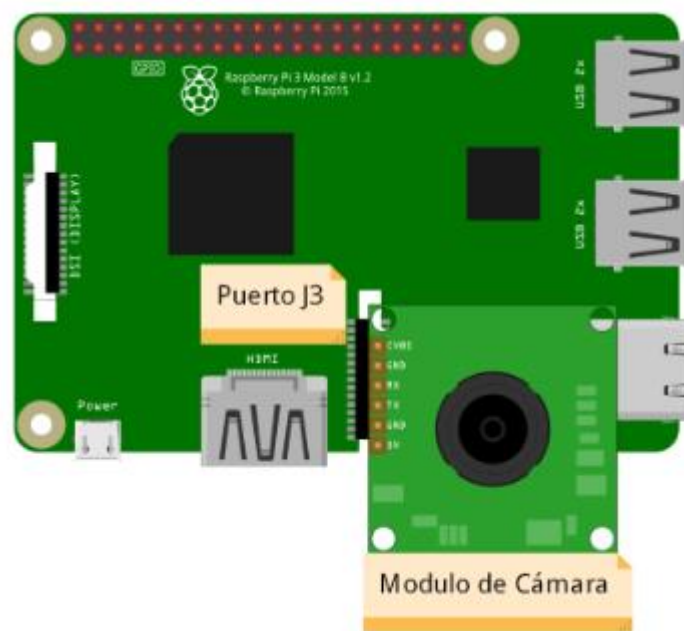


Figura 3.3.2 Conexión del módulo de cámara y Raspberry Pi 3+

- Para el monedero electrónico se describe los pines de conexión en la Tabla 3.3:

Tabla 3.3 Conexiones con el monedero electrónico

Pines Monedero Electrónico	Conexión
VCC	12V (Cargador)
COIN	Pin 18 (Raspberry Pi 3 B+)
GND	0 V (Cargador)

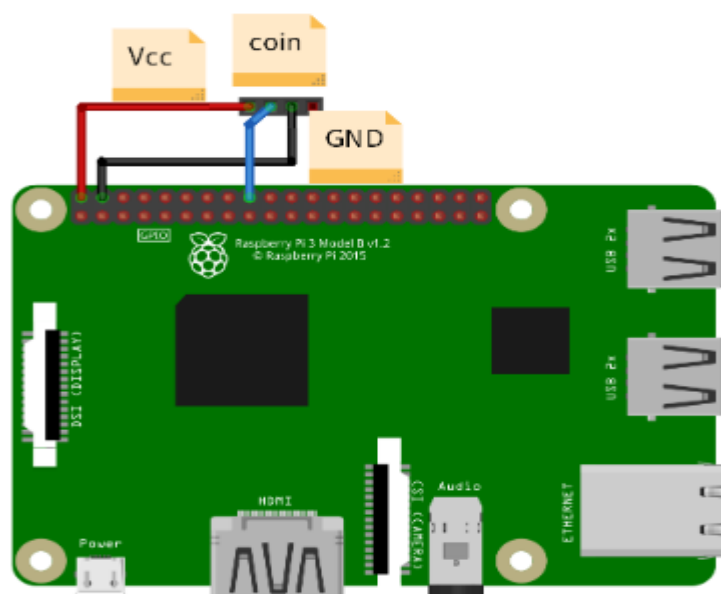


Figura 3.3.3 Conexión del monedero electrónico y Raspberry Pi 3+

3.4 Reportería

La reportería de datos es una página web desarrollada pensando en el administrador y la importancia de analizar los datos obtenidos a través del parquímetro para futuras mejoras o estimaciones. En la página web se presentará información de los registros referentes a las multas.

El administrador tendrá la opción de buscar los registros de multas a través de los siguientes filtros:

- Fecha Desde
- Fecha Hasta
- Placa

Se puede usar cualquier de los filtros por separado o realizando una combinación entre los tres filtros que se pueden visualizar en la Figura 3.4.

REPORTE DE MULTAS

Fecha Desde:

Fecha Hasta:

Placa:

Consultar



ID DISPOSITIVO	PLACA	FECHA	HORA	UBICACION
1	PCA-1234	2002-09-19	22:01:18	ALBORADA
2	GYA-1234	2002-09-19	22:03:55	URDESA
1	GYE-4321	2002-09-19	22:18:40	ALBORADA
2	PCB-1234	2019-09-05	12:17:17	URDESA
3	PCB-1234	2019-09-06	11:40:00	PTO. SANTA ANA

Figura 3.4 Pantalla de Consulta de Multas

Como resultado de la búsqueda por cualquiera de la combinación de filtros antes mencionada se obtendrá la siguiente información:

- Id dispositivo
- Placa
- Fecha
- Hora
- Ubicación

CAPÍTULO 4

4. PRUEBAS Y ANÁLISIS DE RESULTADOS

4.1 Modificaciones

A continuación, en la Tabla 4.1 se registran los cambios realizados en el dispositivo.

Tabla 4.1. Modificación implementada y definido pendiente.

Fecha	Versión anterior	Descripción de modificación	Componentes utilizados	Componentes retirados
17/07/2019	Se utilizó el sensor de movimiento HC-SR501 para detectar la presencia del vehículo en el lugar de parqueo.	Se reemplazó el sensor de movimiento por el sensor de ultrasonido que nos permitió comprobar la presencia de un vehículo sin la necesidad que se encuentre en movimiento	Sensor de ultrasonido HC-SR04.	Sensor HC-SR501
21/08/2019	No existían indicadores que permitan al usuario saber si el dispositivo está en uso.	Se añadieron 3 leds para diferenciar si está en uso el dispositivo	3 leds	-

4.2 Hipótesis y Variables

Las hipótesis que se plantean al iniciar el proyecto según los datos obtenidos de las investigaciones, diagramas de participantes (se los puede visualizar en el apartado de anexos A) generan variables dependientes e independientes son las siguientes:

- **Variable independiente:**

- Automatizar el proceso de generación de multas en los parquímetros ubicados en diversas zonas de la ciudad de Guayaquil.

- **Variables dependientes:**

- Optimizar el recurso de personal por parte de la Agencia de Tránsito Municipal de Guayaquil para la verificación y generación de multas en los parquímetros ubicados en diversas zonas de la ciudad.
- Aumentar los ingresos de la Agencia de Tránsito Municipal de Guayaquil, ya sea a través del cobro del uso de los parquímetros como en la generación de multas por infracciones.

4.3 Pruebas del dispositivo

En la Tabla 4.3 se muestran las pruebas realizadas para el funcionamiento del dispositivo describiendo el resultado y razón de la falla si es el caso.

Tabla 4.2 Pruebas realizadas.

<i>N° de Prueba</i>	<i>Descripción</i>	<i>Resultado</i>	<i>Razón</i>
1	Poder obtener la señal de entrada de dinero.	Fallido	Mala conexión entre el monedero electrónico y la Raspberry Pi 3 B+.
2	Poder obtener la señal de entrada de dinero.	Aprobada	
3	Poder identificar la señal del monedero electrónico en el tiempo de gracia.	Aprobado	
4	Poder identificar la señal del monedero electrónico en el tiempo de Parqueo y aumentar el tiempo de parqueo.	Aprobado	
5	El dispositivo detecte un vehículo que se le acabó el espacio de	Aprobado	Se genera o actualiza el archivo de Excel con los datos adquiridos a través de la cámara.

	estacionamiento comprado		
6	El dispositivo detecta un vehículo en la zona de parqueo y el usuario no ha ingresado dinero.	Aprobado	Se realiza el registro de la multa con la placa del infractor en el documento de extensión .xlsx.
7	Envío de información a la ATM.	Fallido	Mala conexión con el servidor de correo.
8	Envío de información a la ATM.	Fallido	Dirección de correo errónea.
9	Envío de información a la ATM.	Aprobado	Se adjunta el documento de extensión .xlsx en el correo que se envía a la ATM diariamente con los detalles de las multas generadas.
10	Visualizar datos por parte del administrador	Aprobado	Se visualiza los datos en una página web a nivel de sistema.

4.4 Resultados

- Incremento de ingresos y optimización de recursos.**

Se sabe que en la actualidad la ATM tiene un porcentaje de 20,5 % del monto total recaudado por la empresa que administra los parquímetros implementado en diversas zonas de la ciudad de Guayaquil, con lo que el porcentaje 79,5 % es el que le queda a la empresa de Parqueo Positivo. Valores de los porcentajes reflejados en dólares en la Tabla 4.4.

Tabla 4.3 Ingresos anuales recaudado por uso de parquímetros.

Entidad	Monto
ATM	\$195.051,40
Parqueo Positivo	\$756.419

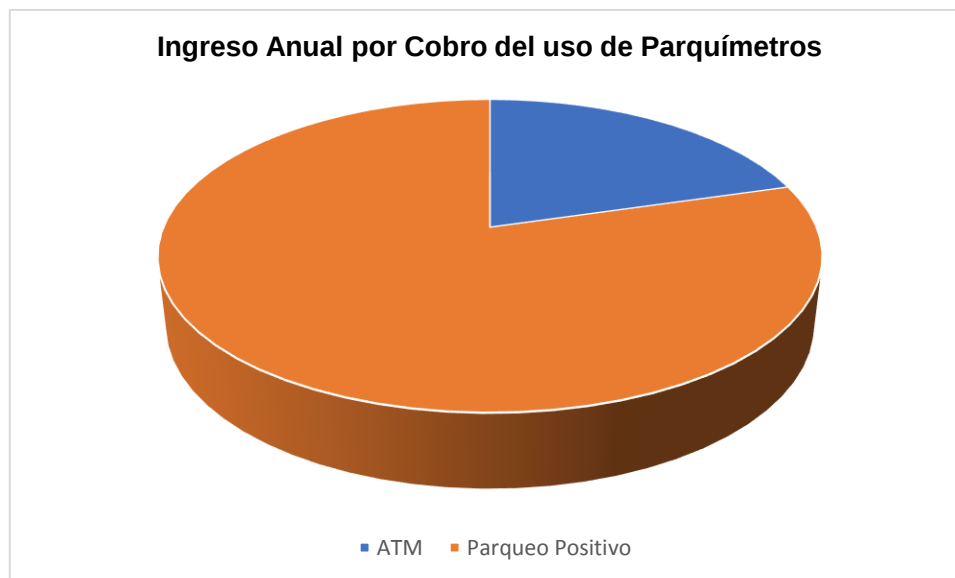


Figura 4.1 Ingresos anuales recaudado por uso de parquímetros.

La inversión total de la empresa cargo de la implementación y administración de los dispositivos actualmente implementados es de \$1.500.000,00 en 627 parquímetros que se encuentran instalados en diversas zonas de Guayaquil. Obteniendo una cotización previa de nuestro dispositivo tenemos un total de \$800,00 por elaboración e implementación de este.



Figura 4.2 Costo de inversión por elaboración e implementación del dispositivo.

- **Cantidad de vehículos sancionados.**

Según estadísticas se tiene un número promedio de 62 multas diarias por el mal uso del parqueo donde se tienen implementado los parquímetros en la ciudad. También se tiene que 4 de cada 10 personas han realizado algún tipo de dádiva

para evitar una multa por parte de un agente de tránsito, lo que se tiene que, si se implementa nuestro dispositivo, existiría un alza de la tasa de multas en un 40% obteniendo un total de aproximadamente 87 multas diarias.

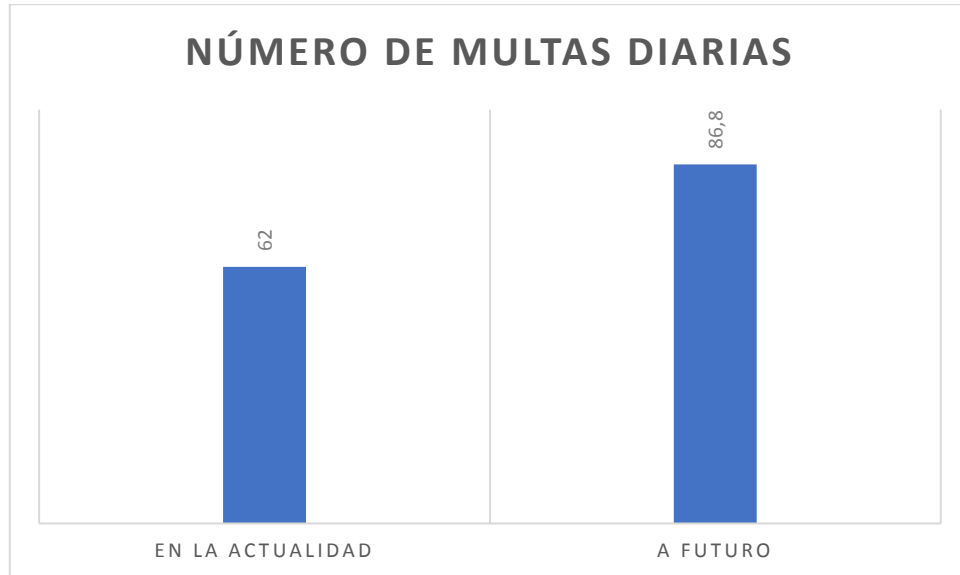


Figura 4.3 Proyección de cantidad de multas.

CAPÍTULO 5

5. CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

- Se desarrolló un dispositivo capaz de automatizar el proceso de multas que realiza la ATM, teniendo como variable de prueba la cantidad de vehículos sancionados por infringir las leyes de espacio y tiempo de estacionamiento; mejorando en un 40% los ingresos a través de multas y un 79,5% el dinero recaudado desde los parquímetros. Además, se obtiene un costo de inversión de \$868.000,00 por los 1085 parqueos que se encuentran en la actualidad funcionando con los parquímetros, lo que genera mayores regalías para el ATM.
- Se optimiza el sistema de generación de multas, con la automatización de estas se evitó que haya errores manuales que ocurren con los agentes de tránsito municipales, generando un mayor ingreso económico por motivos de infracciones y menor egreso como rubro por las labores que cumplen los agentes de tránsito municipales.
- Se crea un modelo de base de datos a nivel administrativo para poder obtener estadísticas a corto y largo plazo, y así poder validar que el prototipo desarrollado cumple con los objetivos establecidos y las expectativas del cliente.

5.2 Recomendaciones

- Comprobar si los dispositivos en la actualidad tienen puertos de comunicación necesarios para establecer enlaces al sistema de datos de la ATM. De esta manera, parte del desarrollo del dispositivo podría acoplarse a los otros que se encuentran instalados en diversas zonas de la ciudad de Guayaquil.

- Usar una cámara con una resolución adecuada para evitar que el procesamiento de imagen (OCR) tenga un mayor costo computacional en el microprocesador utilizado y se evite forzar que el dispositivo tienda a fallos.
- El dispositivo elaborado puede incorporar mejoras que pueden realizarse a futuro se encuentran descritos en la Tabla 5.2 que se detalla a continuación:

Tabla 5.1 Descripción de posibles mejoras para el proyecto.

Versión anterior	Descripción de modificación	Componentes utilizados	Componentes retirados
Se utilizó un monedero electrónico analógico.	Por motivos de pruebas de interacción con usuario se optó por implementar este tipo de monedero.		Monedero electrónico para una sola moneda
Pago a través de efectivo para el uso del parquímetro.	Implementar sistema de pago que permita realizar el pago a través de código USD, tarjeta de débito y aplicativo móvil.	Infraestructura tecnológica que soporte la cantidad de solicitudes de pagos por los medios mencionados y brinde seguridad.	
Generar reportes con estadísticas e indicadores para la ATM.	Crear una base de datos para que se pueda manipular la información contenida en la misma y se pueda generar estadísticas e indicadores para la ATM respecto al uso de los parquímetros y las multas.	Base de datos.	

REFERENCIAS BIBLIOGRÁFICAS.

- Abellán, M. Á. (2018). *PROGRAMO ERGO SUM*. Obtenido de <https://www.programoergosum.com/cursos-online/raspberry-pi/232-curso-de-introduccion-a-raspberry-pi/instalar-raspbian>
- Andrade, A. (04 de Diciembre de 2018). *Expreso.ec*. Obtenido de <https://www.expreso.ec/guayaquil/parquimetros-urdesa-kennedy-transitovehicular-atm-EC2510611>
- BEJOV. (10 de Septiembre de 2019). *BEJOV SANTILLAN*. Obtenido de <https://www.bejob.com/que-puedo-programar-con-python-te-comentamos-3-principales-aplicaciones/>
- Click Formación 360. (2018). *Click Formación 360*. Obtenido de <https://clickformacion360.com/para-que-sirve-python/>
- EcuRED. (2018). *EcuRED*. Obtenido de <https://www.ecured.cu/OpenCV>
- Education IT. (22 de Diciembre de 2017). *Education IT*. Obtenido de <https://blog.educacionit.com/2017/12/22/php-vs-python-cual-es-mejor-para-programacion-del-lado-del-servidor/>
- El Universo. (02 de Diciembre de 2018). *El Universo*. Obtenido de <https://www.eluniverso.com/guayaquil/2018/12/02/nota/7077852/manana-pago-espacios-parquimetros>
- El Universo. (25 de Enero de 2018). *El Universo*. Obtenido de <https://www.eluniverso.com/guayaquil/2018/01/25/nota/6583525/zonas-parquimetros-solo-se-podra-estacionar-hasta-3-horas>
- El Universo. (02 de Enero de 2019). *El Universo*. Obtenido de <https://www.eluniverso.com/guayaquil/2019/01/02/nota/7119825/hoy-rige-cobro-multas-parquimetros>
- EMOVTZA. (2013). *EMOVTZA*. Obtenido de <http://www.emovtza.zaruma.gob.ec/noticias>
- GAD MUNICIPAL DE PIÑAS. (21 de Mayo de 2015). *GAD MUNICIPAL DE PIÑAS*. Obtenido de <https://www.pinas.gob.ec/gadm-pinas/2015-07-03-03-53-36/unidad-de-transito.html>
- Hernández, L. d. (12 de Noviembre de 2018). *Programar Facil.com*. Obtenido de <https://programarfacil.com/podcast/81-vision-artificial-opencv-phyton/>
- Kingston Technology. (2019). *https://www.kingston.com*. Obtenido de <https://www.kingston.com/latam/community/articledetail/articleid/29664>

Mordvintsev, A., & K., A. (2013). *https://opencv-python-tutroals.readthedocs.io/*.
 Obtenido de *https://opencv-python-tutroals.readthedocs.io/*: Alexander
 Mordvintsev & Abid K

Parqueo Positivo. (2003). *Parqueo Positivo*. Obtenido de
http://www.parqueopositivo.com/nosotros/

Punto Flotante S.A. (2017). *https://puntoflotante.net*. Obtenido de
https://puntoflotante.net/MANUAL-DEL-USUARIO-SENSOR-DE-MOVIMIENTO-PIR-HC-SR501.pdf

Python Software Foundation. (18 de Junio de 2019). *https://pypi.org/*. Obtenido de
https://pypi.org/project/pytesseract/

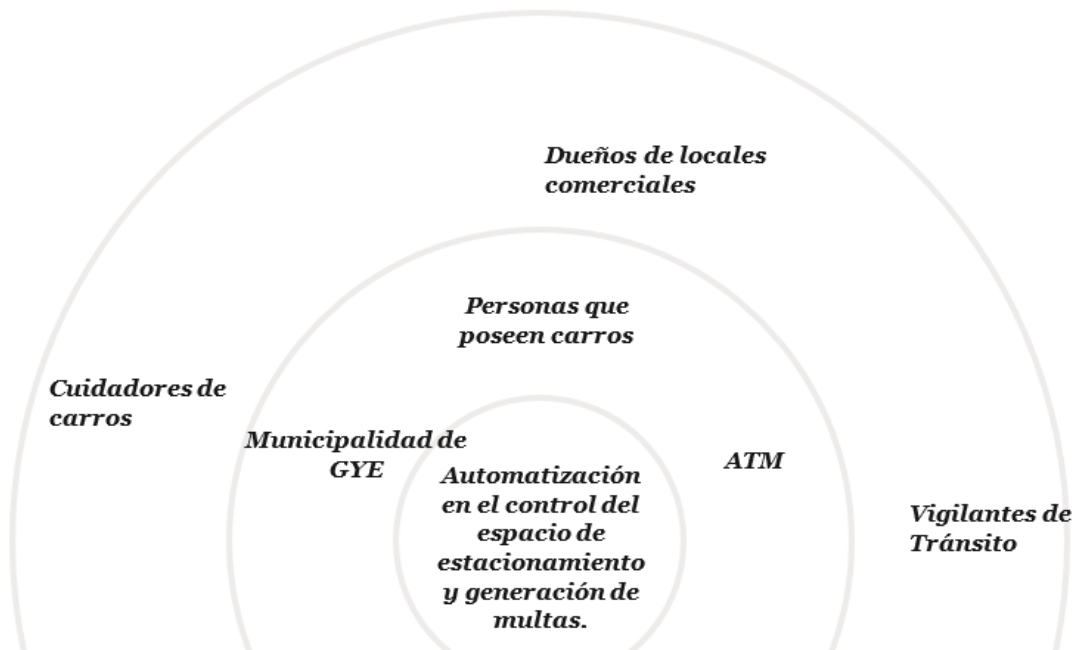
Raspberry Pi Foundation. (s.f.). *https://static.raspberrypi.org*. Obtenido de
https://static.raspberrypi.org: https://static.raspberrypi.org/files/product-briefs/Raspberry-Pi-Model-Bplus-Product-Brief.pdf

Velasco, R. (24 de Febrero de 2019). *SoftZone SZ*. Obtenido de
https://www.softzone.es/2019/02/24/mejores-sistemas-raspberry-pi-3-2019/

ANEXOS

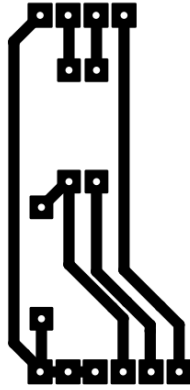
Anexo A

Mapa de actores



Anexo B

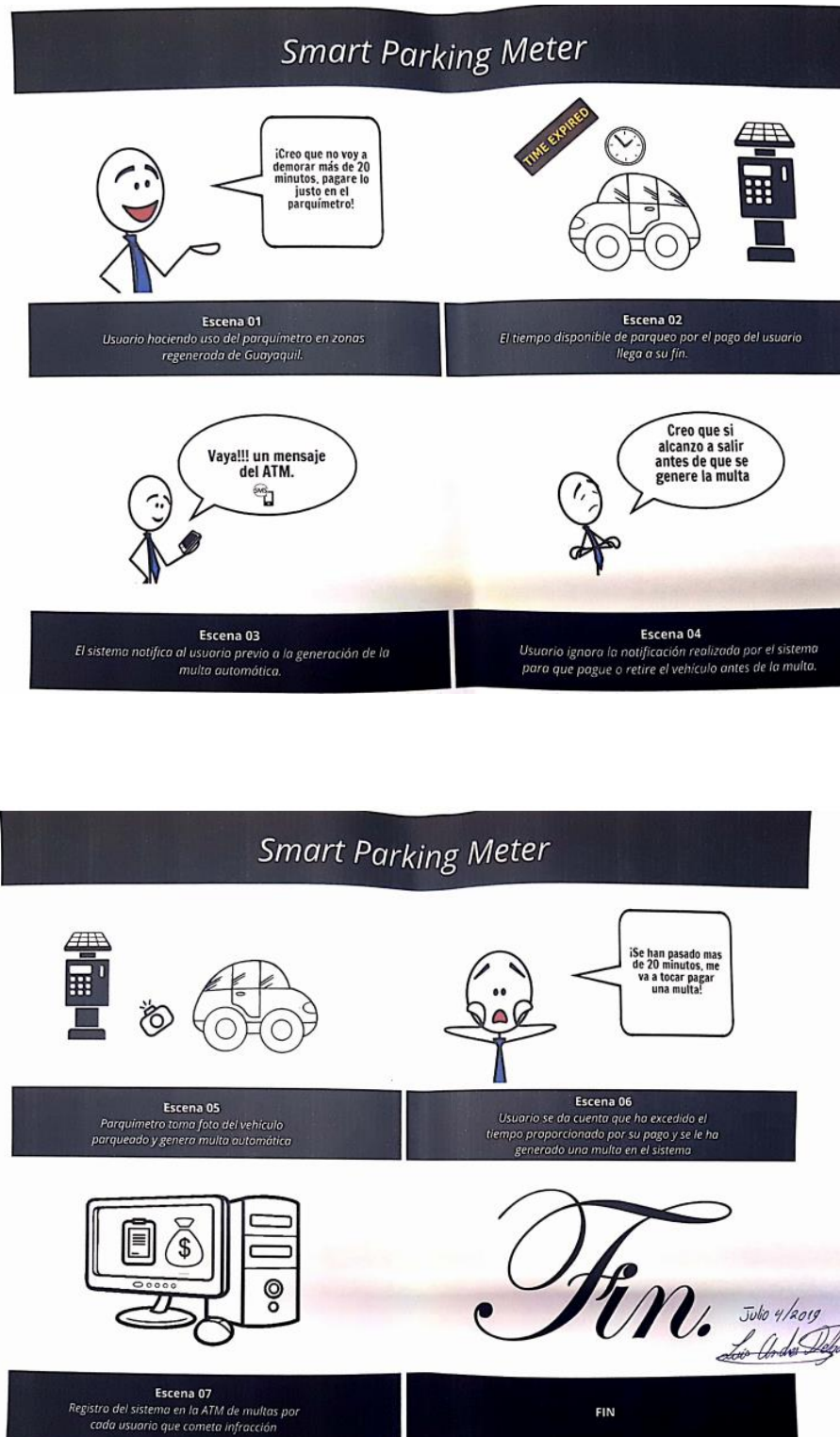
Diseño de la PCB para la conexión de sensores con la Raspberry Pi 3 B+:



Anexo C

Validaciones del cliente:

- Validación de prototipo de baja resolución del cliente:



- Validación de prototipo de alta resolución del cliente:

Automatización de Multas en los Parquímetros de las Zona Alborada
(Parque Gran Albocentro) en Guayaquil

VALORACIÓN DE PROTOTIPO DE ALTA RESOLUCIÓN

Diseño de Prototipo

- ☐ Malo
 ☐ Regular
 ☐ Excelente
☐ Bueno
 ☒ Muy Bueno

Diseño de Página Web

- ☐ Malo
 ☐ Regular
 ☐ Excelente
☐ Bueno
 ☒ Muy Bueno

Funcionamiento del prototipo

- ☐ Malo
 ☐ Regular
 ☒ Excelente
☐ Bueno
 ☐ Muy Bueno

Información Presentada en la Página

- ☐ Malo
 ☐ Regular
 ☐ Excelente
☒ Buena
 ☐ Muy Buena

Observaciones

Sería buena una app que te muestre en tiempo real
 que parquímetros se encuentran ocupados y libres para
 el uso del parqueo

Luis Delgado

Luis Delgado

Fiscalizador de la ATM.

Anexo D

Manual de Instalación.

1. Descargar los archivos `main.py`, `enviarDatos.py`, `test.sh` y `enviarDatos.sh` del servidor de la atm.
2. Dirigirse al directorio principal, para ello ejecutamos `cd ~`
3. Una vez en el directorio principal ejecutamos el comando `crontab -e` con el editor de texto `nano`.
4. Al final del archivo añadir las siguientes líneas
 - 4.1. `python /Documents/ParquimetroInteligente/main.py`
 - 4.2. `python /Documents/ParquimetroInteligente/enviarDatos.py`
5. Guardar el documento con los cambios realizados, para ello ejecutamos el comando `CTRL + O`, luego presionamos la tecla `Enter` y luego ejecutamos el comando `CTRL+X`.
6. Una vez guardado las modificaciones realizadas en el archivo `crontab`, reiniciamos la raspberry y automáticamente el proceso empezará a ejecutarse una vez se haya iniciado el dispositivo.

Anexo E

Para acceder a la reportería dirigirse al siguiente enlace:

<http://localhost:8080/reporteria.registros.php>

Se nos presentará la siguiente página en donde el usuario podrá ingresar el rango de fechas en el que prefiera buscar o por la placa del vehículo.

REPORTE DE MULTAS

Fecha Desde:

Fecha Hasta:

Placa:

Consultar



ID DISPOSITIVO	PLACA	FECHA	HORA	UBICACION
1	PCA-1234	2002-09-19	22:01:18	ALBORADA
2	GYA-1234	2002-09-19	22:03:55	URDESA
1	GYE-4321	2002-09-19	22:18:40	ALBORADA
2	PCB-1234	2019-09-05	12:17:17	URDESA
3	PCB-1234	2019-09-06	11:40:00	PTO. SANTA ANA

Donde podemos encontrar los siguientes escenarios.

1. Si ingresa solo la **fecha desde**, la información que se mostrará será que pertenezca desde esa fecha ingresada hasta el día actual.
2. Si ingresa solo la **fecha hasta**, la información que se le mostrará será desde el inicio hasta la fecha ingresada.
3. Si ingresa solo la **placa**, la información que se le presentará será correspondiente solo al número de placa.
4. Si ingresa **fecha desde** y **fecha hasta**, la información que se presentará será referente a la que se encuentre en el rango de fechas antes mencionado.
5. Si ingresa **fecha desde** y **placa**, la información que se presentará será la correspondiente a la placa y el rango de fechas desde la fecha antes mencionada hasta la actualidad.
6. Si ingresa **fecha hasta** y **placa**, la información que se le presentará será la referente a la placa y el rango de fechas desde el inicio hasta la fecha antes mencionada.

7. Si ingresa **fecha desde**, **fecha hasta** y **placa**, la información presentada será muy específica referente a la placa y al rango de fechas ingresado.
8. SI no ingresa nada el sistema automáticamente le presentará todos los registros.