

**Tecnológico Nacional de México**

Instituto Tecnológico de La Laguna



**Parkimaniacos**

**PROPUESTA DE PROYECTO**

LÍNEA DE GENERACIÓN Y APLICACIÓN DEL CONOCIMIENTO:

INGENIERÍA DE SOFTWARE

**INGENIERÍA EN SISTEMAS COMPUTACIONALES**

Presenta:

José Ángel García Arce NC: 17130786

Luis Emmanuel Méndez Barrios NC: 17130804

Enrique Antonio Belmarez Meraz NC: 17130765

José Baltazar Martínez De La Rosa NC: 1713004

Torreón, Coahuila a 12/14/2020

**Índice general**

[**1. Introducción**](#_40ohrti85djy) **3**

[1.1. Antecedentes teóricos que fundamenten la problemática a tratar](#_ya1yd95rcyaa) 4

[1.2. Descripción detallada del problema a resolver](#_3eaeiomdz121) 6

[1.3. Objetivos (general y particulares)](#_15y4xjtklo1h) 6

[1.4. Alcances y limitaciones](#_7eqx49blen91) 7

[1.5. Justificación del proyecto](#_agyvx6e3t12e) 8

[Nombre y Logotipo del Proyecto:](#_6xh9g7hca1pm) 9

[**2. Metodología**](#_3n10jyjhhoi) **9**

[Metodología SCRUM:](#_8qgs6xqqx922) 10

[Al estar enmarcada dentro de las metodologías agile, Scrum se basa en aspectos como:](#_dqcrv25zmkg) 10

[1. Product owner:](#_b86spnjce8wy) 11

[2. Scrum Master:](#_98kbq6atfdjk) 12

[3. Equipo de desarrollo:](#_kzuz4vjhernl) 12

[**3. Desarrollo de la aplicación de software**](#_759i1na1ogvf) **13**

[3.1. Ingeniería de requisitos](#_pjo0g7usdu2d) 14

[3.1.1 Registro e inicio en la aplicación móvil](#_c6i8o1d6gt01) 14

[3.1.2 Funcionamiento con Temporizador y lector QR](#_lpeizpqs5a3) 16

[3.1.3 Capacidad para poder interactuar con diferentes sistemas.](#_hpui44ogczqg) 17

[3.2.1 Requerimientos de interfaces externas](#_m3u8uwvhtixk) 17

[3.2.2 Requerimientos de interfaz de usuario](#_i7kl6fy9li71) 17

[Interfaces de hardware](#_hpxhvixqiqjw) 19

[Interfaces de software](#_mkpj4q6teol) 19

[Interfaces de comunicación](#_1ukh31xu176l) 19

[3.1.1. Análisis](#_ti4y2wh0pexx) 23

[3.2.1. Especificación de casos de uso](#_bnionsr9jule) 24

[3.2.2. Diagrama de flujo](#_atgy3b2nls) 25

[3.2.3. Diagrama de estados](#_9n1c94x42qt7) 26

[3.2.4. Diagrama de clases](#_nlbsh9s4lexd) 27

[3.2.5 Diagrama entidad relación](#_9oxtfxt7bmjw) 28

[3.3.3. Realizar prototipo](#_ytpxvivzax02) 29

[3.4. Implementación](#_c70tsc66p6wn) 31

[3.4.1. Estructurar modelo de implementación (estándares en la implementación, estructura de la aplicación por módulos, etc.)](#_7kczt0p05nom) 32

[3.4.2. Planificar la integración (módulos u otras aplicaciones)](#_7ie0gtwlc99w) 34

[3.5. Pruebas.](#_gx9d4itdnro) 36

[3.6. Implantación](#_cgnswntxpgqf) 41

[3.6.1. Planificación de implantación](#_5n9ieplq5f5r) 41

[3.6.2. Desarrollo de material de apoyo y manual de usuario](#_tb92vq974n60) 42

[Material de apoyo:](#_yhlz1qy1ylgj) 42

[**4. Cronograma de actividades**](#_75sxdelqa6of) **49**

[**5. Referencias**](#_o73oar58hawe) **54**

[**6. Lista de figuras y tablas**](#_5ymwwlboyg5k) **55**

[**7. Anexos**](#_qoleb76w65pr) **56**

# 1. Introducción

Un parquímetro es un dispositivo ubicado en la vía pública que permite el ordenamiento y medición del estacionamiento en áreas definidas para ello. Su función consiste en recolectar dinero a cambio del derecho de estacionar un vehículo en un lugar público, por una cantidad de tiempo. Los parquímetros pueden ser usados por las municipalidades o ayuntamientos como una herramienta para hacer valer sus políticas de estacionamiento en la calle, que por lo general están relacionadas con el tráfico y las políticas de gestión de movilidad.

En la actualidad el mercado ofrece diferentes alternativas para ofrecer una solución integral al problema del estacionamiento medido. Algunas empresas continúan fabricando parquímetros mecánicos y siguen proporcionando mantenimiento por lo que en algunas ciudades es común ver todavía los antiguos parquímetros en funcionamiento. Estos sistemas fueron ya desplazados notablemente por los sistemas más avanzados de control de múltiples espacios, los cuales mediante recibos de pago controlan el estacionamiento y poseen sistemas de comunicación que le proporcionan alertas tanto a usuarios como a encargados del control de tránsito de la situación de los mismos, así como también facilitan el pago mediante tarjetas de crédito débito y tarjetas prepagas sin depender exclusivamente de monedas.

Algunos parquímetros proporcionan sistemas de detección de vehículos que permiten controlar cuando se encuentran vehículos estacionados en los diferentes lugares, posibilitando así el anuncio de un lugar vacante, la detección de violaciones de tiempos de estacionamiento y la recolección de datos estadísticos.

Con este proyecto se busca optimizar el funcionamiento de un parquímetro mediante la implementación de la tecnología, utilizando los dispositivos móviles se perfeccionará el pago y uso de los parquímetros.

## 1.1. Antecedentes teóricos que fundamenten la problemática a tratar

En el año 2016 el primer sistema de parquímetros virtuales fue instalado en Zacatlán de las Manzanas Puebla, donde los usuarios tenían la opción de pagar el servicio a través de una aplicación, enviando un mensaje de texto o pagando en los negocios de la zona, información que llega directamente a los oficiales que conocen en tiempo real los lugares que ya están pagados.

Como pueden ver, este era un sistema mucho más primitivo, en cuanto a las formas de pago, a lo que vendría a continuación.

El **15 de junio** **del año 2017** inició el modelo a prueba de **parquímetros virtuales “Aquí hay lugar”**, que sustituirá a los parquímetros con monedas en el municipio.

Fue desarrollado ahora para la ciudad **de Guadalajara**

Además, tiene como objetivo promover la rotación de los vehículos, para que exista mayor disponibilidad de estacionamiento.

En la primera etapa funcionará con un total de 1,625 cajones, distribuidos en 5 zonas de la ciudad: **Arcos Vallarta**, **Lafayette**, **Libertad, Americana** y **Centro**.

**¿Cómo funcionan los parquímetros virtuales?**

* Es un modelo a prueba y sin concesión.
* La operación estará a cargo del Municipio. El Gobierno de Guadalajara retoma el control del estacionamiento en la vía pública.
* Su operación representará una mínima inversión para el municipio, ya que sólo se invertirá en balizamiento.
* A través del uso de la tecnología se hará más sencillo el pago. Podrás pagar de 3 maneras diferentes:
* Pago en red de negocio Aquí hay Lugar en efectivo
* A través de la app Parkimovil, paga con tarjeta de crédito, débito o abonar dinero en la red de negocios.
* Enviando un mensaje de texto al 25505 con cargo a tu saldo del celular.
* Después este proyecto se fue extendiendo a ciudades como : Tampico, Durango etc.
* Pero cabe recalcar que eran compañías diferentes las encargadas apoyadas con el gobierno del estado de poner en funcionamiento el proyecto de los parquímetros virtuales.

## 1.2. Descripción detallada del problema a resolver

El proyecto, como se puede apreciar en la figura 1.0, tiene como nombre Parkimaniacos, lo cual hace referencia al concepto del proyecto está desarrollando con el nombre de la empresa Figura 1.0 Maniacorp.

Actualmente estamos en la versión 1.1 del proyecto, contando con diversas mejoras al anterior diseño del mismo.

Este documento se compone de la totalidad del proyecto, incluyendo el hardware que será utilizado para la correcta administración de los parquímetros, esperamos que este documento cumpla todas las

necesidades del proyecto en general.

## 1.3. Objetivos (general y particulares)

**Objetivo general:**

* Optimizar el uso de los parquímetros y la buena gestión de estos mismos.

**Objetivos particulares:**

* Realizar una aplicación móvil la cual permita al usuario el pago del parquímetro de manera sencilla y eficaz
* Realizar una página web la cual nos permite publicitar el proyecto.
* Proponer un método de uso y pago del parquímetro que permita facilitar la interacción que existe entre el cliente y los parquímetros.
* Elaborar un protocolo robusto que nos permita actuar en casos de accidentes.
* Verificar la correcta gestión de los recursos monetarios tanto por los usuarios y las autoridades.
* Brindar un servicio que sea fácil de utilizar y que cuente con seguridad durante el uso del parquímetro.
* Brindar un servicio de soporte técnico para aquellos clientes que tengan dudas sobre el funcionamiento de la aplicación o quieran dar su opinión sobre la aplicación.

## 

## 

## 1.4. Alcances y limitaciones

Con este proyecto se pretende que cualquier persona que hace uso de los parquímetros de manera regular logre tener acceso a una opción práctica y sencilla de utilizar los parquímetros, se espera tener un alcance amplio.

El tipo de clientes a los que está dirigido son: Sector de gobierno o cualquier empresa que necesite la administración de su estacionamiento. El usuario final será a quien consideramos para lograr una aplicación amigable con el usuario, el proyecto tendrá un amplio alcance , de lo descrito anteriormente, también se busca que aparte del uso que le pudiera dar el sector poblacional a la aplicación Parkimaniacos, las autoridades correspondientes también tendrán un papel importante en el correcto uso de los parquímetros ya que ellos de la mano con la aplicación en cuestión van a poder mejorar el uso de los parquímetros, y a su vez monitorear las horas en las cuales haya más flujo vehicular en ellos.

## 1.5. Justificación del proyecto

Un parquímetro es un dispositivo ubicado en la vía pública que permite el ordenamiento y medición del estacionamiento en áreas definidas para ello. Su función consiste en recolectar dinero a cambio del derecho de estacionar un vehículo en un lugar público, por una cantidad de tiempo. Los parquímetros pueden ser usados por las municipalidades o ayuntamientos como una herramienta para hacer valer sus políticas de estacionamiento en la calle, que por lo general están relacionadas con el tráfico y las políticas de gestión de movilidad.

En la actualidad el mercado ofrece diferentes alternativas para ofrecer una solución integral al problema del estacionamiento medido. Algunas empresas continúan fabricando parquímetros mecánicos y siguen proporcionando mantenimiento por lo que en algunas ciudades es común ver todavía los antiguos parquímetros en funcionamiento. Estos sistemas fueron ya desplazados notablemente por los sistemas más avanzados de control de múltiples espacios, los cuales mediante recibos de pago controlan el estacionamiento y poseen sistemas de comunicación que le proporcionan alertas tanto a usuarios como a encargados del control de tránsito de la situación de los mismos, así como también facilitan el pago mediante tarjetas de crédito débito y tarjetas prepagas sin depender exclusivamente de monedas.

Algunos parquímetros proporcionan sistemas de detección de vehículos que permiten controlar cuando se encuentran vehículos estacionados en los diferentes lugares, posibilitando así el anuncio de un lugar vacante, la detección de violaciones de tiempos de estacionamiento y la recolección de datos estadísticos.

Con este proyecto se busca optimizar el funcionamiento de un parquímetro mediante la implementación de la tecnología, utilizando los dispositivos móviles se perfeccionará el pago y uso de los parquímetros.

## Nombre y Logotipo del Proyecto:

El color rosa representa la modernidad comercial como un color emotivo y sensible que trata de llegar a su audiencia conectando a través de sus emociones. Ya que va en tasa de crecimiento este color además de tener un flow muy bueno.

El negro representa un estado de formalidad lujo seriedad en este caso nos pudo ayudar al gran contraste que tiene como a la formalidad seriedad que presenta este equipo

El nombre del Proyecto hace alusión al nombre de la empresa que lo está desarrollando **Maniacorp** por lo tanto el nombre del mismo es **Parkimaniacos.**



**Figura 1.0:** Logotipo del proyecto

# 2. Metodología

Para el desarrollo del proyecto **Parkimaniacos** utilizaremos la **metodología Scrum** la cual se trata un marco de trabajo o framework que se utiliza dentro de equipos que manejan proyectos complejos. Es decir, se trata de una **metodología de trabajo ágil** que tiene como finalidad la entrega de valor en períodos cortos de tiempo y para ello se basa en tres pilares: la transparencia, inspección y adaptación. Esto permite al cliente, junto con su equipo comercial, insertar el producto en el mercado pronto, rápido y empezar a obtener ventas.

## Metodología SCRUM:

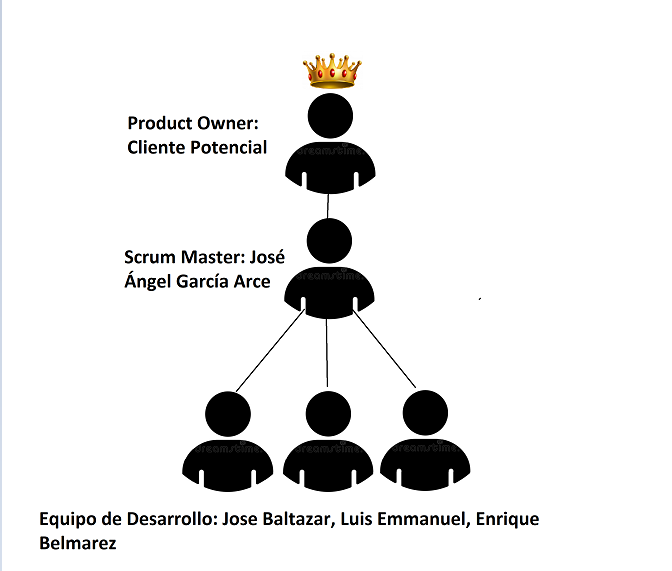
## Al estar enmarcada dentro de las [metodologías agile](https://www.wearemarketing.com/es/blog/que-es-la-metodologia-agile-y-que-beneficios-tiene-para-tu-empresa.html), Scrum se basa en aspectos como:

* La **flexibilidad** en la adopción de cambios y nuevos requisitos durante un proyecto complejo.
* El factor **humano.**
* La **colaboración** e interacción con el cliente.
* El desarrollo iterativo como forma de asegurar buenos **resultados.**

Con la metodología Scrum, el equipo tiene como foco entregar valor y ofrecer resultados de calidad que permitan cumplir los objetivos de negocio del cliente.

Para ello, los equipos de Scrum son auto-organizados y multifuncionales. Es decir, cada uno es responsable de unas tareas determinadas y de terminarlas en los tiempos acordados. Esto garantiza la entrega de valor del equipo completo, sin necesidad de ayuda o la supervisión minuciosa de otros miembros de la organización.

En Scrum existen 3 roles muy importantes : Product Owner, Scrum Master y Equipo de desarrollo. Así como se muestra en la figura 2.0 se observar los roles que se han asignado a los participantes del desarrollo del proyecto.



**Figura 2.0:** Jerarquía de Desarrollo Metodologia Scrum.

Como se vio anteriormente la jerarquía ya está bien definida como se muestra en la figura anterior y a continuación se describirán cada uno de estos roles:

### 1. Product owner:

Es el responsable de maximizar el valor del trabajo del equipo de desarrollo. La maximización del valor del trabajo viene de la mano de una buena gestión del **Product Backlog**, el cual explicaremos más adelante.

El **Product owner** es el único perfil que habla constantemente con el cliente, lo que le obliga a tener muchos conocimientos sobre negocio.

Para finalizar, un **equipo Scrum** debe tener solo un Product Owner y este puede ser parte del equipo de desarrollo.

### 2. Scrum Master:

Es el responsable de que las **técnicas Scrum** sean comprendidas y aplicadas en la organización. Es el manager de Scrum, un líder que se encarga de eliminar impedimentos o inconvenientes que tenga el equipo dentro de un *sprint* (que ya revisaremos en detalle más adelante), aplicando las mejores técnicas para fortalecer el equipo de marketing digital.

Dentro de la organización, el Scrum Master tiene la labor de ayudar en la adopción de esta metodología en todos los equipos.

### 3. Equipo de desarrollo:

Son los encargados de realizar las **tareas priorizadas por el Productor Owner**. Es un equipo multifuncional y auto-organizado. Son los únicos que estiman las tareas del product backlog, sin dejarse influenciar por nadie.

Los **equipos de desarrollo** no tienen sub-equipos o especialistas. La finalidad de esto es transmitir la responsabilidad compartida si no se llegan a realizar todas las tareas de un sprint.

# 3. Desarrollo de la aplicación de software

Consideramos que el proyecto Parkimaniacos, tiene una idea que al menos en la región (la comarca lagunera) no está siendo explotada, consideramos que de aplicarse de manera correcta el proyecto podría tener un gran impacto en la sociedad, solucionando un problema que a simple vista podría ser no muy impactante, pero que a su vez facilita la interacción de las personas con los parquímetros.

Además, es un proyecto que traerá grandes beneficios no solo a las personas que utilizan los parquímetros día con día sino al departamento de tránsito y vialidad de la entidad en cuestión.

Actualmente el proyecto se encuentra en la primera fase desarrollo al igual que estamos haciendo una ponderación de que tan rentable podría ser el proyecto en cuestiones económicas(Arduino, cámaras, equipo en general para la adecuación de los parquímetros).

A continuación, se mencionan alguna de las tareas que se tienen que realizar para poder desarrollar el proyecto de manera satisfactoria :

* Realizar un análisis más detallado sobre el uso del parquímetro inteligente por si es necesario implementar más tecnologías (cámaras, sensores etc.) **2**
* Realizar el diseño y terminar de desarrollar la aplicación móvil, así como la página web del proyecto. **4**
* Hacer pruebas y solucionar todos los problemas que se vayan dando con el desarrollo del proyecto. **4**
* Empezar a negociar con las autoridades locales sobre la implementación del proyecto. **5**
* Implementar el proyecto de forma satisfactoria. **3**

Hablando ahora sobre donde está localizada la empresa **Maniacorp** actualmente no contamos con una localización física establecida, para poder trabajar como equipo en dicha área. Sin embargo, adoptamos la metodología de trabajar desde casa o como se le conoce comúnmente como Home office. (Independencia 27010 colonia navarro interior b)

La manera en la que se desarrolla el proyecto **Parkimaniacos** va ser utilizando la metodología SCRUM los miembros de la empresa realizan el proyecto de manera satisfactoria entregando resultados (Sprint) cada semana. En nuestro caso los avances se entregan cada viernes. como se explicó anteriormente.

## 3.1. Ingeniería de requisitos

En este punto se revisarán los requerimientos del proyecto en general los cuales tienen que ser cubiertos dada la necesidad de los clientes.

### 3.1.1 Registro e inicio en la aplicación móvil

Descripción: En este caso es muy claro el título el usuario se tiene que registrar ya sea con su correo electrónico(Obviamente uno válido) o con alguna red social eso ira cambiando conforme con el desarrollo de la aplicación.

Prioridad: La prioridad de esta tarea es muy alta, ya que se tiene que establecer una cuenta de usuario válida para poder acceder a la aplicación de los parquímetros ya que esta va estar de la mano con el método de pago que elija el usuario.

Acciones iniciadas y comportamiento esperado: En este caso sería que el usuario se registre en la app con una cuenta válida, ya sea de correo o con alguna red social como se ha mencionado previamente.

Una vez hecho lo anterior se tendrá que loguear en la app para poder comenzar con el uso de la aplicación.

**Requerimientos funcionales:**

**REQ 1 – Registro de Cuentas**

El usuario para poder ingresar a la aplicación por primera vez tiene que registrarse previamente para poder interactuar con la app utilizando correo electrónico o alguna red social

**REQ 2 - Control de acceso mediante un login**

En este requerimiento se habla acerca de que el usuario se tiene loguear en la aplicación para poder interactuar con ella obviamente si hay un error es porque el registro o el inicio de sesión tienen algún dato erróneo así que verifique bien la información.

**REQ 3 - Gestión de Cuentas de Usuario**

En este apartado se deja claro que la gestión de las cuentas de los usuarios es muy importante, ya que la seguridad que se va a brindar tiene que ser la de mejor nivel, para que el usuario se sienta seguro al momento de utilizar la aplicación.

**REQ 4 - Protección de la información**

La información tiene que estar protegida bajo los estándares determinados correspondientes.

### 3.1.2 Funcionamiento con Temporizador y lector QR

Descripción: el usuario a la hora de querer empezar a utilizar la aplicación después de registrarse tendrá que leer un código QR que se encontrara en el parquímetro, para después empezar con el contador de tiempo y cuando deje el parquímetro el sensor dará la señal a la aplicación y se detendrá el cronómetro y se procederá al cobro del tiempo que se estuvo en el parquímetro.

Prioridad: Este es un elemento que es de suma importancia ya que dependiendo del tiempo que el usuario esté en el parquímetro será el monto a pagar por lo tanto se generará un ingreso.

Acciones iniciadas y comportamiento esperado: Lo esperado en esta parte es que el usuario empiece con la lectura del QR y después el cronómetro empieza a contar, para que cuando el usuario regrese y desaloje el cajón el cronómetro pare y se haga el cobro.

**Requerimientos funcionales:**

**REQ 1 – Sensores**

En este caso se utilizarán unos sensores de proximidad los cuales vamos a definir mejor en la sección de requerimientos no funcionales.

**REQ 2 – Pagos Online**

El usuario deberá poder pagar de manera online con total seguridad

## 

## 

## 

### 3.1.3 Capacidad para poder interactuar con diferentes sistemas.

Descripción: El Cliente podrá utilizar el software en el sistema que él desee y para cuando él lo desee.

Prioridad: No es demasiado alta, pero si es a tomar en cuenta y además que tenga seguridad

**Requerimientos funcionales:**

**REQ 1 – Seguridad en el parquímetro**

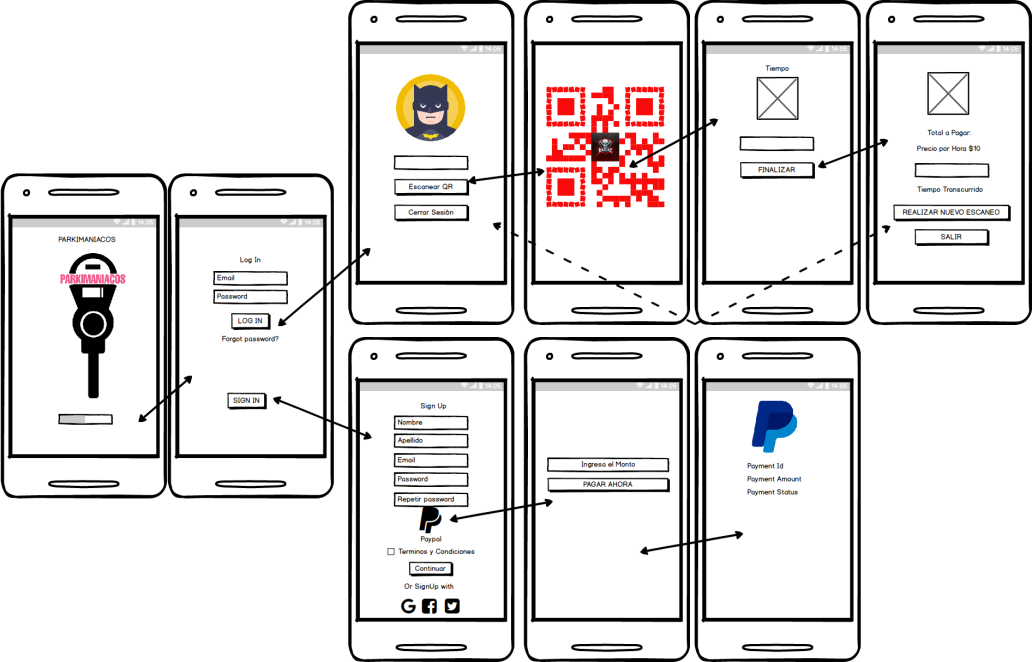
El Parquímetro tiene que ser seguro y para eso se implementarán cámaras aparte seguridad al momento del pago.

## 3.2.1 Requerimientos de interfaces externas

## 3.2.2 Requerimientos de interfaz de usuario

**GUÍA para el usuario final**

En esta interface se estandarizó un diseño minimalista para la fase de pruebas donde se pusieron iconos sencillos, en la ventana de inicio de sesión se incrusto botones predeterminados por el entorno gráfico , al igual que los campos de textos Las capturas de la Aplicación fueron tomadas cronológicamente:



**Figura 3.0** Diseño en Balsamiq

## Interfaces de hardware

Computadores: Página web para mostrar estadísticas y publicidad

dispositivos móviles: Aplicación Principal del proyecto con mayor interacción con el cliente

otros dispositivos: Arduino para poder controlar los sensores y actuadores de nuestro sistema y poder interactuar con la base datos

Protocolos de comunicación que soporta

TCP/IP, HTTP ,HTTPS SSH

## Interfaces de software

Se creó un triángulo de interacción entre hardware - DB – móvil

Donde el hardware le mandara los espacios disponibles a la base de datos para cuando la aplicación móvil consulte los espacios aparezcan en tiempo real , por medio de triggers en la base de datos y por medio de sensores ultrasónicos en el hardware , con la herramienta de ethernet shield se conecta el arduino a la base de datos y por medio de web service en el apartado de móvil

Y es en el mismo caso cuando el espacio está lleno , manda la información y en la aplicación móvil empieza a contar el tiempo para después que el hardware le comunique que está disponible nuevamente , termina de conteo y cobra la cantidad.

## Interfaces de comunicación

ETHERNET: Es la de un estándar de comunicación de red capaz de manejar grandes cantidades de datos a velocidades de 10 Mbps o 100 Mbps. La especificación utiliza un protocolo abierto en la capa de aplicación. En la industria es especialmente popular para aplicaciones de control.

HTTP: es el protocolo de comunicación que permite las transferencias de información en la World Wide Web. HTTP fue desarrollado por el World Wide Web Consortium y la Internet Engineering Task Force, El desarrollo de aplicaciones web necesita frecuentemente mantener estado. Para esto se usan las cookies, que es información que un servidor puede almacenar en el sistema cliente.

HTTP: es un protocolo de comunicación de Internet que protege la integridad y la confidencialidad de los datos de los usuarios entre sus ordenadores y el sitio web. Dado que los usuarios esperan que su experiencia online sea segura y privada, te recomendamos que adoptes HTTPS para proteger sus conexiones con tu sitio web, independientemente de lo que este contenga.

WAP: es un estándar abierto internacional para aplicaciones que utilizan las comunicaciones inalámbricas, por ejemplo: acceso a servicios de Internet desde un teléfono móvil.

SSH: es un protocolo que facilita las comunicaciones seguras entre dos sistemas usando una arquitectura cliente/servidor y que permite a los usuarios conectarse a un host remotamente. A diferencia de otros protocolos de comunicación remota tales como FTP o Telnet, SSH encripta la sesión de conexión, haciendo imposible que alguien pueda obtener contraseñas no encriptadas.

**Requerimientos no funcionales**

Aquí se definen los requerimientos implícitos en las peticiones del usuario los cuales son los siguientes:

Web services: todo lo que tiene que ver con el manejo del servidor y servicios en la nube para poder acceder al servidor y a la información del mismo.

**Eficiencia:**

El sistema debe ser capaz de procesar N transacciones por segundo. Esto se medirá por medio de la herramienta SoapUI aplicada al Software Testing de servicios web.

Toda funcionalidad del sistema y transacción de negocio debe responder al usuario en menos de 5 segundos.

El sistema debe ser capaz de operar adecuadamente con hasta 100.000 usuarios con sesiones concurrentes.

Los datos modificados en la base de datos deben ser actualizados para todos los usuarios que acceden en menos de 2 segundos.

**Seguridad lógica y de datos:**

Los permisos de acceso al sistema podrán ser cambiados solamente por el administrador de acceso a datos.

El nuevo sistema debe desarrollarse aplicando patrones y recomendaciones de programación que incrementen la seguridad de datos.

Todos los sistemas deben respaldarse cada 24 horas. Los respaldos deben ser almacenados en una localidad segura ubicada en un edificio distinto al que reside el sistema.

Todas las comunicaciones externas entre servidores de datos, aplicación y cliente del sistema deben estar encriptadas utilizando el algoritmo RSA.

Si se identifican ataques de seguridad o brecha del sistema, el mismo no continuará operando hasta ser desbloqueado por un administrador de seguridad.

**Usabilidad:**

El tiempo de aprendizaje del sistema por un usuario deberá ser menor a 4 horas.

La tasa de errores cometidos por el usuario deberá ser menor del 1% de las transacciones totales ejecutadas en el sistema.

El sistema debe contar con manuales de usuario estructurados adecuadamente.

El sistema debe proporcionar mensajes de error que sean informativos y orientados al usuario final.

El sistema debe contar con un módulo de ayuda en línea.

La aplicación web debe poseer un diseño “Responsive” a fin de garantizar la adecuada visualización en múltiples computadores personales, dispositivos tablets y teléfonos inteligentes.

El sistema debe poseer interfaces gráficas bien formadas y su mejora continua.

## 

## 

## 

## 

## 

## 3.1.1. Análisis

En este caso además uno de los temas a analizar es que dentro de todos los requerimientos existen muchos requerimientos explícitos o que no son expresados por el usuario así que tenemos que identificar bien dichos requerimientos de manera específica ya que se tienen que desarrollar de manera independiente ya que estos están implícitos como se mencionó previamente.

Cuando se diseña un sistema no basta con crear diagramas de uso acerca de las funciones, sino identificar cuáles son los elementos que nos guiaran como subsistemas, dependencias, interfaces, nodos, entre otros; estos elementos nos muestran de dónde partir y hasta dónde debemos llegar. Una vez que identificamos estos elementos, podemos representarlos gráficamente mediante diagramas en los cuales se expresan las entradas de información, las salidas, las estructuras lógicas y las formas en que la información se procesa.

Los diagramas UML son una representación gráfica de la arquitectura del sistema, cada uno muestra la misma información, pero con diferentes enfoques (recordando el ejemplo de la construcción del edificio, varios colaboradores con diferentes funciones, pero con el mismo objetivo).

En este caso, para la etapa de análisis de nuestro proyecto realizamos los 3 siguientes tipos de diagramas para la compresión de una manera gráfica del proyecto en cuestión.

1. Diagrama de casos de uso: es una descripción breve, a alto nivel, del caso de uso y del flujo de acciones que comprende

2. Diagrama de flujo: es una forma esquemática de representar ideas y conceptos en relación. Comúnmente, se utiliza este tipo de **diagramas** para detallar el proceso de un algoritmo y, así, se vale de distintos símbolos para representar la trayectoria de operaciones precisas a través de flechas

3. Diagrama de estados: muestran el conjunto de **estados** por los cuales pasa un objeto durante su vida en una aplicación en respuesta a eventos

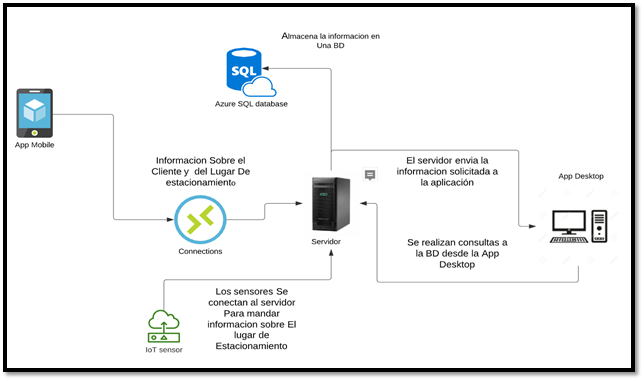
**Como podremos observar en las siguientes figuras.**

## 3.2.1. Especificación de casos de uso

****

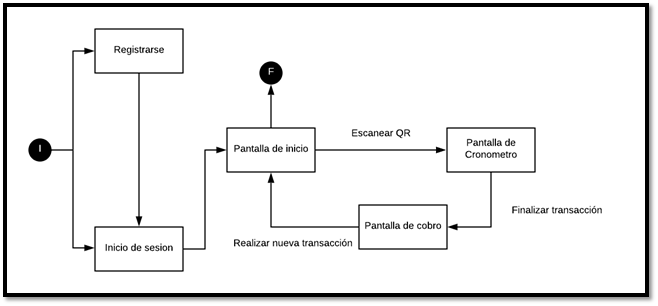
**Figura 3.1:** Diagrama de usos: los usuarios o las entidades son el cliente y el administrador, y se tiene como escenas a la Base de datos, el Hardware del parquímetro(Físicamente) y a las distintas apps las cuales será la móvil(Usuario) y la de escritorio (Autoridad).

### 3.2.2. Diagrama de flujo



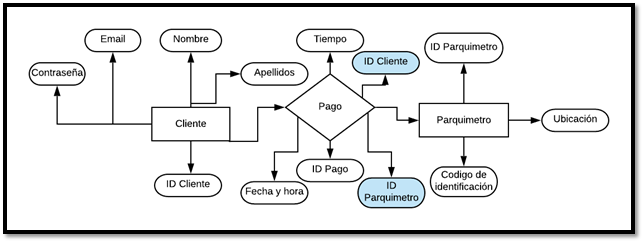
**Figura 3.2** Diagrama de Flujo**:** Como se puede apreciar el diagrama de flujo explica o más bien expresa como funciona el proceso interior que tendría la aplicación. En este caso desde que el usuario la ejecuta la aplicación en su dispositivo móvil y se relaciona con el servidor y la aplicación de escritorio

### 3.2.3. Diagrama de estados



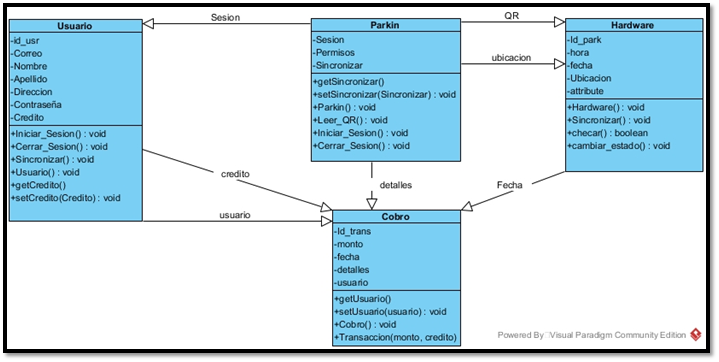
**Figura 3.3** Diagrama de estados**:** aquí se encuentran las etapas por las cuales pasa la aplicación al ser ejecutada, ya sea registro hasta el escanear un código QR representando los cuadros una pantalla de la aplicación y las líneas de texto los botones.

### 3.2.4. Diagrama de clases



**Figura 3.4 Diagrama de clases:** cómo se puede ver en la figura anterior las clases principales son la de cliente y parquímetro con sus diferentes atributos y se puede apreciar que la relación del cliente y el parquímetro es el pago que existe.

### 3.2.5 Diagrama entidad relación



**Figura 3.5 Diagrama entidad relación:** cómo se puede observar se define la estructura de la aplicación atreves de las clases y se deja en claro que cada clase siempre estar.

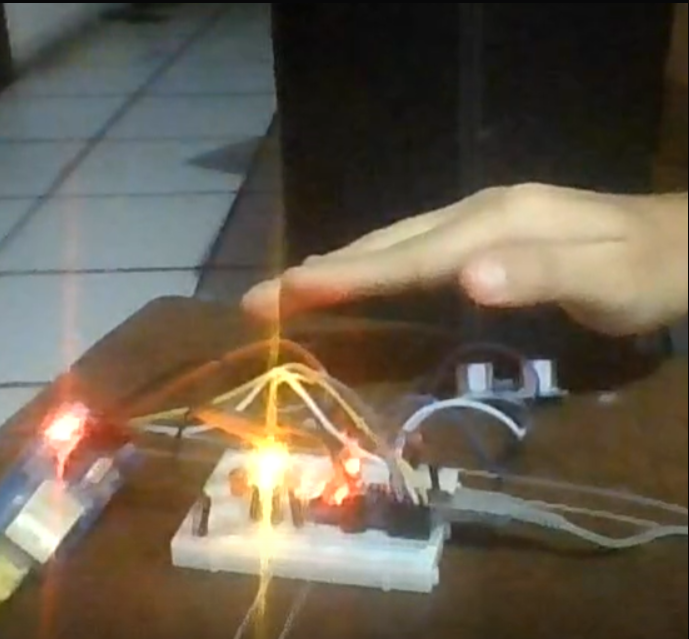
## 

## 3.3.3. Realizar prototipo

El prototipo que realizamos del proyecto al menos de la manera física en arduino sería que led que enciende en color azul representa cuando un cajon esta desocupado el amarillo es cuando esta ocupado pero aun no esta leído el QR. y el Rojo es cuando esta ocupado pero ya esta leído el QR. En la presentación del día 17 ahondaremos

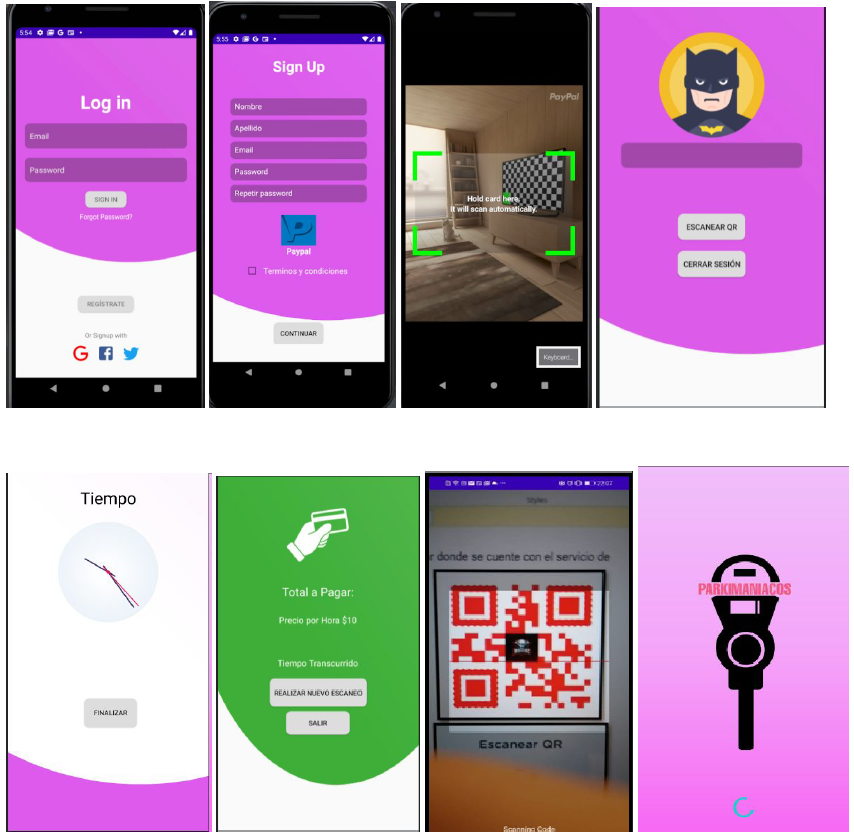
más en este tema.

También se presentan unas capturas de pantalla de la aplicación móvil del diseño que se tiene hasta ahora.





**Figura 3.6**: Circuito de Arduino del proyecto

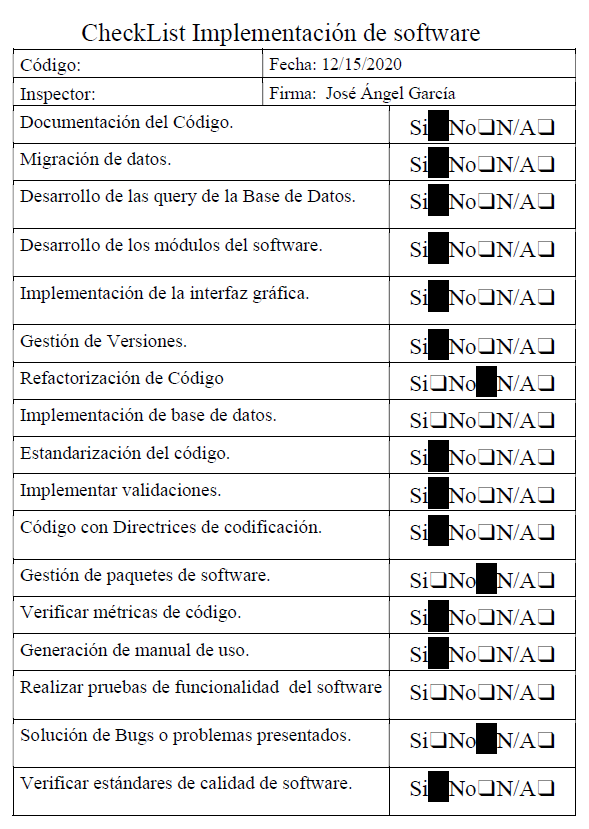


**Figura 3.7:** Capturas de pantalla de la aplicación móvil

## 3.4. Implementación

El plan de implementación se tiene que apegar a diversos estándares para desarrollar de manera correcta la aplicación apegado obviamente a diferentes estándares como por ejemplo los estándares de programación estándares de diseño etc.

Para esto se tuvieron que llenar diferentes plantillas las cuales contenían checklist que nos daban pauta o nos ayudaban al momento de ir realizando la implementación del proyecto para cerciorarse de que se estaban cumpliendo los requerimientos y cumplir como se muestra en la siguiente Figura, la cual es el check list de la implementación de el proyecto



**Figura 3.8:** Checklist implementación del software

### 3.4.1. Estructurar modelo de implementación (estándares en la implementación, estructura de la aplicación por módulos, etc.)

Los módulos que se encuentran en nuestra aplicación son 3 principales los cuales serían la **aplicación móvil, la base de datos** , y la **página web**.

Siendo la **aplicación móvil** la actividad principal o la que más prioridad tiene sobre los demás, obviamente todas son importantes pero consideramos que esta es la principal ya que es con la que el usuario final va terminar interactuando y por eso mismo damos nuestro principal enfoque en el mismo. Y dicha aplicación la realizamos en **Android Studio** con el lenguaje de programación Java. Ya que todo el equipo de desarrollo está familiarizado con dicho lenguaje de programación.



**Figura 3.9:** Logo de Android Studio y Java

Para la **Base de datos** utilizamos la base de datos de **MySql** comunicada con la app mediante una API Rest echa en **Node JS**



Utilizaremos Arduino para programar los componentes que vamos a utilizar al momento de la interacción con los parquímetros.



### 

### 3.4.2. Planificar la integración (módulos u otras aplicaciones)

Primeramente al momento de elegir las tecnologías se tiene que tener en mente tanto los requerimientos del usuario como la planificación de la integración de las aplicaciones así como la compatibilidad de las mismas como la experiencia de los desarrolladores con dichas herramientas.

Después de explicar lo anterior la manera mas optima que encontramos para comunicar primeramente el servidor con la aplicación móvil fue mediante la creación de una **API Rest** en Nodejs, a continuación se explicara brevemente a qué nos referimos con el anterior concepto.

El término REST (Representational State Transfer) se originó en el año 2000, descrito en la tesis de Roy Fielding, padre de la especificación HTTP. Un servicio REST no es una arquitectura software, sino un conjunto de restricciones con las que podemos crear un estilo de arquitectura software, la cual podremos usar para crear aplicaciones web respetando HTTP.

Hoy en día la mayoría de las empresas utilizan API REST para crear servicios. Esto se debe a que es un estándar lógico y eficiente para la creación de servicios web. Por poner algún ejemplo tenemos los sistemas de identificación de Facebook, la autenticación en los servicios de Google (hojas de cálculo, Google Analytics, …).

Según Fielding las restricciones que definen a un sistema RESTful serían:

* **Cliente-servidor**: esta restricción mantiene al cliente y al servidor débilmente acoplados. Esto quiere decir que el cliente no necesita conocer los detalles de implementación del servidor y el servidor se “despreocupa” de cómo son usados los datos que envía al cliente.
* **Sin estado**: aquí decimos que cada petición que recibe el servidor debería ser independiente, es decir, no es necesario mantener sesiones.
* **Cacheable**: debe admitir un sistema de almacenamiento en caché. La infraestructura de red debe soportar una caché de varios niveles. Este almacenamiento evitará repetir varias conexiones entre el servidor y el cliente para recuperar un mismo recurso.
* **Interfaz uniforme**: define una interfaz genérica para administrar cada interacción que se produzca entre el cliente y el servidor de manera uniforme, lo cual simplifica y separa la arquitectura. Esta restricción indica que cada recurso del servicio REST debe tener una única dirección, “URI”.
* **Sistema de capas**: el servidor puede disponer de varias capas para su implementación. Esto ayuda a mejorar la escalabilidad, el rendimiento y la seguridad.

Después la siguiente interacción o integración importante sería la del circuito de **arduino** con la base de datos mediante el uso de la librería ethernet tiene la función connect que es la encargada de hacer la conexión al servidor para hacer las peticiones GET que nos pide 2 parámetros , 1 la ip del servidor y 2 el puerto por el cual se mandará las peticiones.

## 

## **3.5. Pruebas**.

El protocolo del proceso de testing es el encargado de gestionar todo lo relacionado con el proceso de prueba del proyecto. Las pruebas son básicamente un conjunto de actividades dentro del desarrollo de software. Dependiendo del tipo de pruebas, estas actividades podrán ser implementadas en cualquier momento de dicho proceso de desarrollo. Existen distintos modelos de desarrollo de software, así como modelos de pruebas. A cada uno corresponde un nivel distinto de involucramiento en las actividades de desarrollo.

En el desarrollo de software existen dos diferentes tipos de pruebas las cuales son las siguientes:

**Pruebas estáticas**

Son el tipo de pruebas que se realizan sin ejecutar el código de la aplicación. Puede referirse a la revisión de documentos, ya que no se hace una ejecución de código. Esto se debe a que se pueden realizar "pruebas de escritorio" con el objetivo de seguir los flujos de la aplicación.

**Pruebas dinámicas**

Todas aquellas pruebas que para su ejecución requieren la ejecución de la aplicación. Las pruebas dinámicas permiten el uso de técnicas de caja negra y caja blanca con mayor amplitud. Debido a la naturaleza dinámica de la ejecución de pruebas es posible medir con mayor precisión el comportamiento de la aplicación desarrollada.

Estos tipos de pruebas expuestos anteriormente son los que se estarán haciendo a él proyecto con el fin de garantizar el correcto funcionamiento del mismo a continuación se estará expresando cómo se realizarán cada una de estas.

**Prueba estática:**

Como se mencionó en la página anterior para este tipo de pruebas lo primordial no es ejecutar el código sino la revisión de los diferentes documentos que componen el proyecto, así como también los patrones de diseño seguidos durante la creación del código y obviamente valga la redundancia la creación del código.

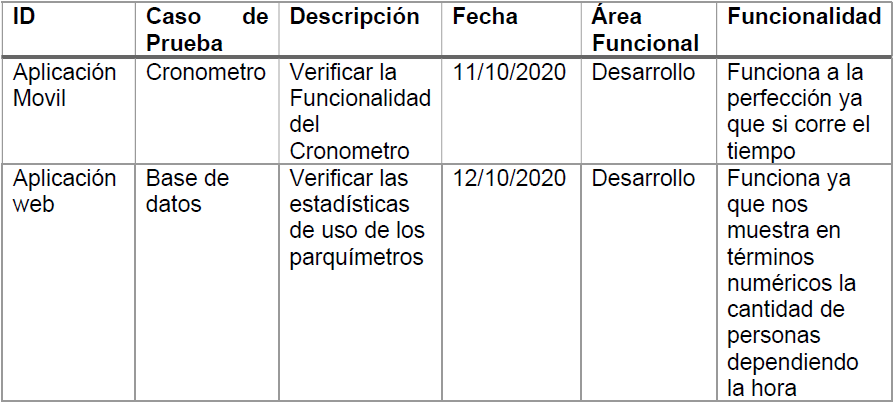
Por lo cual la implementación de dichas pruebas es fundamental a la hora de asegurarnos de la calidad de el producto en cuestión, por lo cual en este caso las pruebas serán sobre este tipo de documentos, así como del diseño de la aplicación siguiendo los estándares y las normas necesarias para seguir con los lineamientos previamente establecidos por el cliente que a su vez satisfacen nuestros estándares de calidad.

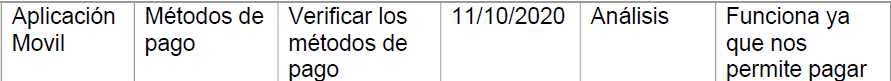
**Caja Negra:**

Las pruebas de caja negra, es una técnica de pruebas de software en la cual la funcionalidad se verifica sin tomar en cuenta la estructura interna de código, detalles de implementación o escenarios de ejecución internos en el software.

En las pruebas de caja negra, nos enfocamos solamente en las entradas y salidas del sistema, sin preocuparnos en tener conocimiento de la estructura interna del programa de software. Para obtener el detalle de cuáles deben ser esas entradas y salidas, nos basamos únicamente en los requerimientos de software y especificaciones funcionales.

**Ejemplo de plantilla para caso de Prueba de Caja Negra:**

****

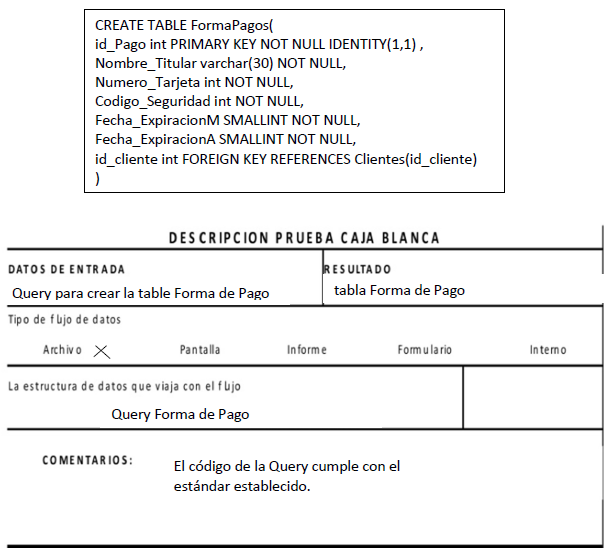
****

**Tabla 1.0:** Prueba de Caja Negra

**Caja Blanca:**

En programación, se denomina cajas blancas a un tipo de pruebas de software que se realiza sobre las funciones internas de un módulo. Así como las pruebas de caja negra ejercitan los requisitos funcionales desde el exterior del módulo, las de caja blanca están dirigidas a las funciones internas. Entre las técnicas usadas se encuentran; la cobertura de caminos (pruebas que hagan que se recorran todos los posibles caminos de ejecución), pruebas sobre las expresiones lógico-aritméticas, pruebas de camino de datos (definición-uso de variables), comprobación de bucles (se verifican los bucles para 0,1 e interacciones, y luego para las interacciones máximas, máximas menos uno y más uno).

**Ejemplo de plantilla para realizar pruebas de caja blanca:**

****

**Tabla 2.0:** Prueba de caja blanca

**Pruebas de la API Rest:**

Para realizar los test sobre la api utilizaremos una herramienta llamada Postman esta herramienta Postman nace como una herramienta que principalmente nos permite crear peticiones sobre APIs de una forma muy sencilla y poder, de esta manera, probar las APIs Alrededor de la idea de testear las APIs, Postman nos ofrece un conjunto de utilidades adicionales para poder gestionar las APIs de una forma más sencilla.



## 3.6. Implantación

la Implantación es una fase muy importante ya que es en esta fase es donde se van a poner a prueba todos los conocimientos aplicados en todo el desarrollo del proyecto por eso mismo se tiene que desarrollar una planificación muy bien implementada para poder tener éxito y una recepción adecuada de parte de los usuarios que utilicen la aplicación.

## **3.6.1. Planificación de implantación**

Para comenzar se debe determinar cómo y cuándo el producto estará disponible al usuario final. La planificación de la implantación en el ambiente del usuario requiere un alto nivel de colaboración y preparación por parte del cliente. El Plan de Implantación debe lograr un software pronto para liberar en el ambiente del usuario, material de capacitación y material de apoyo para asegurar que el usuario final pueda usar exitosamente el producto liberado.

La conclusión exitosa del proyecto puede ser muy influenciada por factores que están fuera del alcance del desarrollo del software como son la infraestructura edilicia o la infraestructura de hardware no adecuada y personal no preparado para trabajar con el nuevo sistema.

Dicho lo anterior , necesitamos que nos den al menos 3 semanas de anticipación para tener listo el funcionamiento del parquímetro de la manera más óptima, obviamente el tiempo va en función a la cantidad de parquímetros solicitados. Además de requerir un depósito extra para la mano de obra.

Entonces en el siguiente punto se presentará el manual de usuario así como un pequeño video para representar mejor el uso de la aplicación de una manera gráfica al usuario final y al cliente potencial.

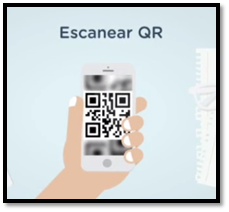
## 3.6.2. Desarrollo de material de apoyo y manual de usuario

### Material de apoyo:

Video de uso de la aplicación

**1.** Como primer paso debemos estacionar el vehículo en un lugar donde se cuente con el servicio de “Parkimaniacos”.



**2.** Después se procederá a escanear el código QR que se encuentre situado en el Parquímetro, el cual será sólo compatible con la aplicación.

**3.** Una vez hecho lo anterior empezará a correr el tiempo



**4.** Una vez el usuario haya desocupado el lugar, la transacción habrá finalizado

**Manual de Usuario:**

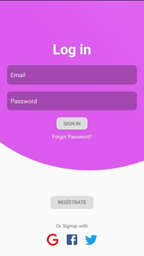
1. Primeramente se tiene que instalar la aplicación dentro de la play store en este caso Parkimaniacos como podemos ver en la siguiente imagen.



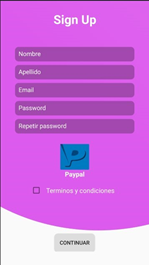
2. Al Iniciar la aplicación como podremos ver es el splash después de un momento (Máximo 3) veremos la splash screen dándole tiempo a la aplicación que cargue la app.

3.-Ingreso al Sistema

Primero se debe crear un usuario por distintas formas por Facebook Google o Twitter o en ya en segunda instancia se pone un correo electrónico



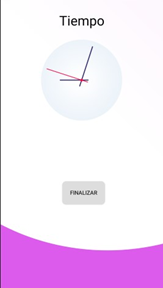
4.- Seguido de eso pondremos nombre, Apellido, email, contraseña y repetir de nuevo la contraseña después de eso se deben aceptar los términos seguido de eso pondremos el botón continuar



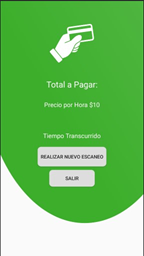
5.- Después de eso podremos escanear el código QR o por su defecto cerrar sesión



6.-Después de eso comenzaremos a ver la siguiente ilustración del tiempo donde comienza a correr y después de pasar el tiempo y te pueda sobrar tiempo se podrá finalizar o en este caso correrá el tiempo hasta que usuario desee finalizar



6.-Después de eso se pondrá el total a pagar en este caso serán 10 pesos donde la transacción será por pay pal y si no desea pagar solo salir, pero se acreditará a una multa



7- Seguido de esto la otra opción será el pago en pay pal inicia su sesión de paypal donde será acreditará la hora que se cobró

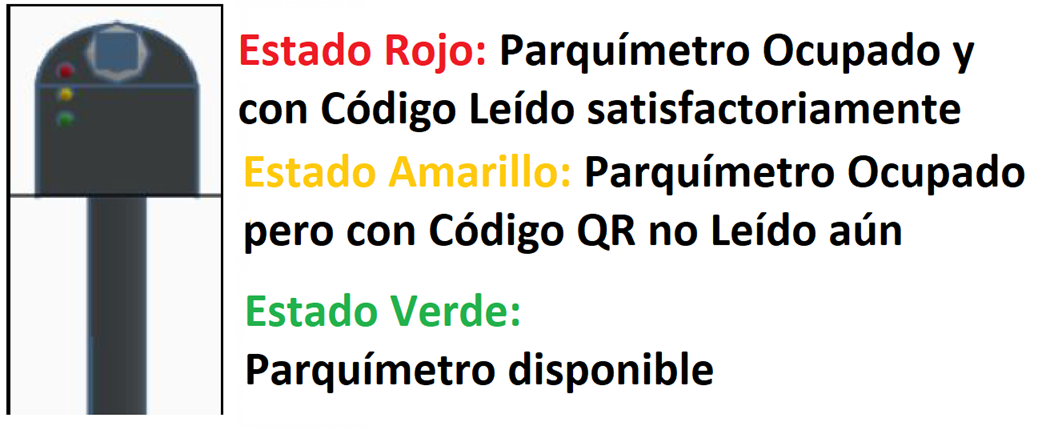


8.-Aquí podemos observar donde solo se oprime o se selecciona el pagar con paypal y podemos concluir el pago del parquímetro



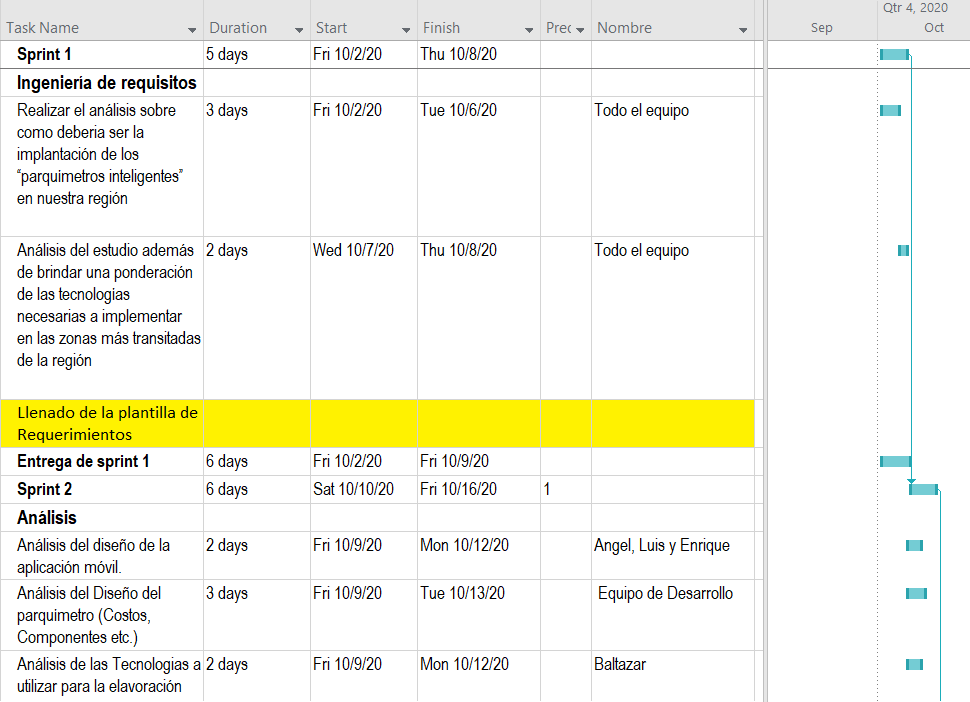
Ejemplo de QR situado en cada parquímetro

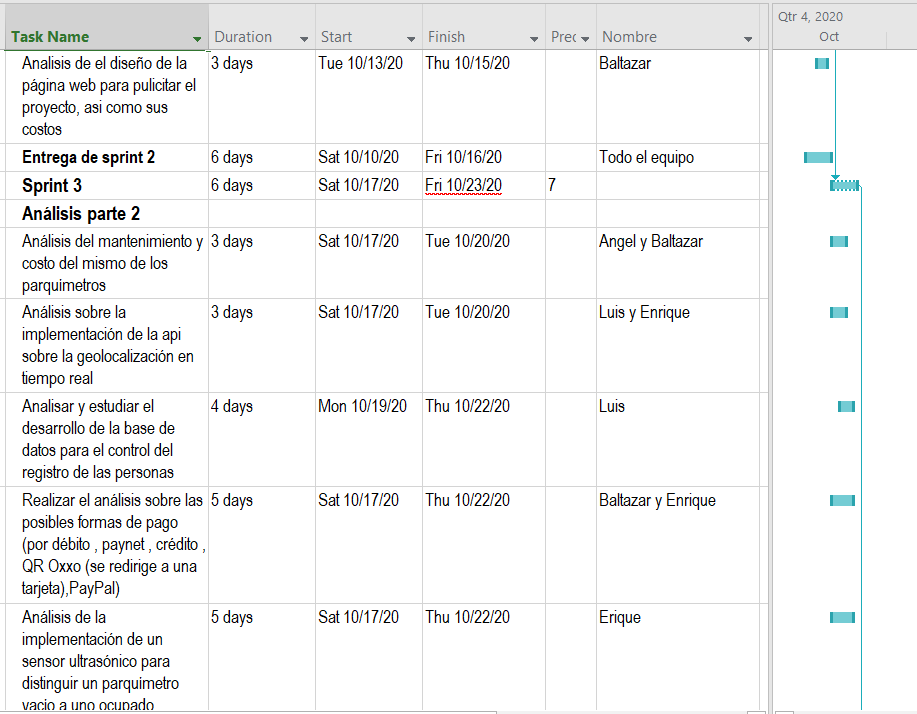
En todo caso si el usuario no sabe si ya se leyó su QR o no cada parquímetro contará con 3 Leds los cuales indican diferentes estados del parquímetro a continuación se hará una breve explicación de los mismos.

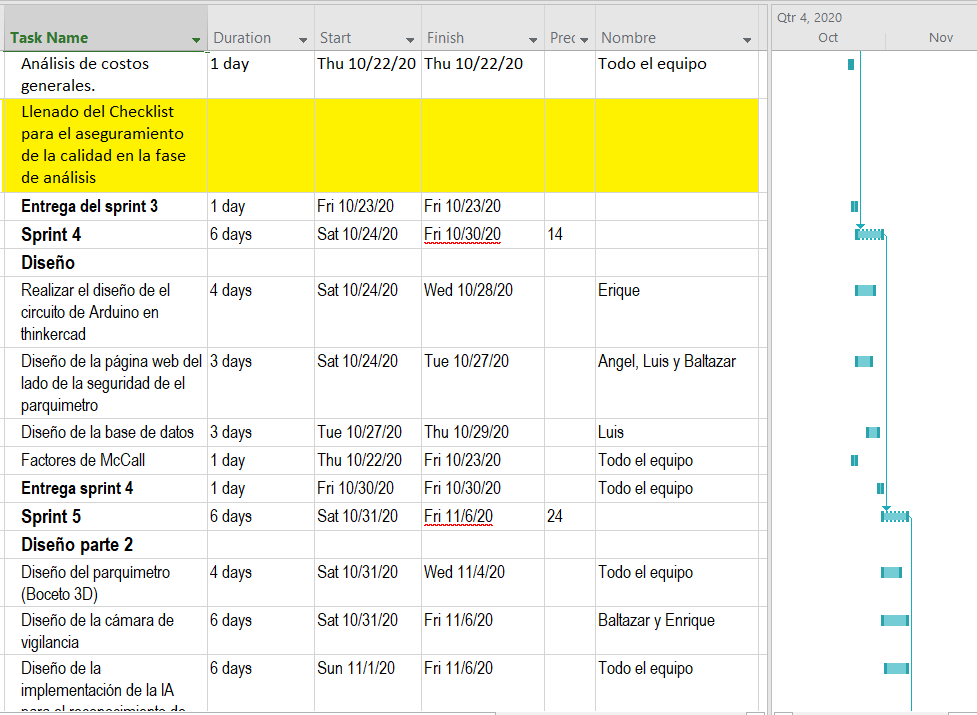


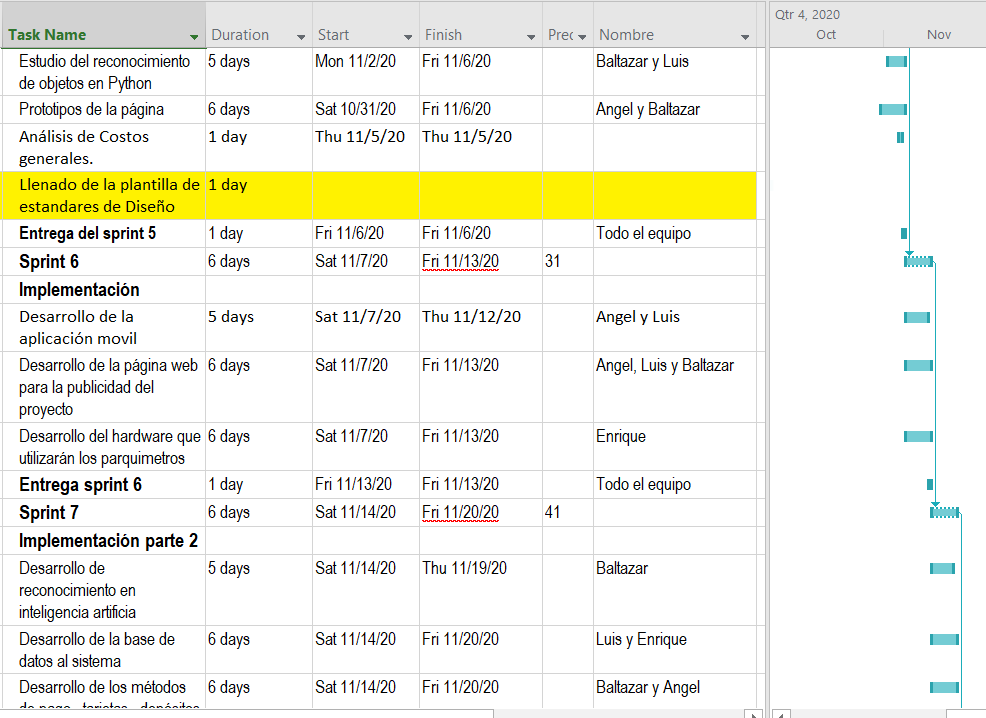
**Figura 3.10**: Explicación de los estados del parquímetro

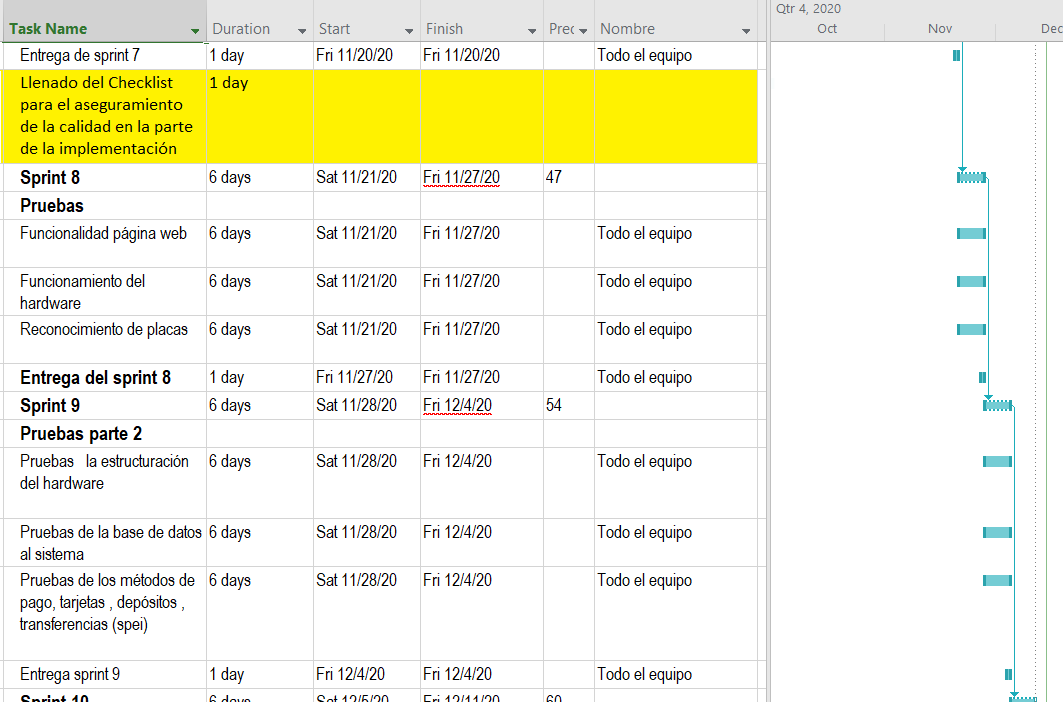
# 4. Cronograma de actividades

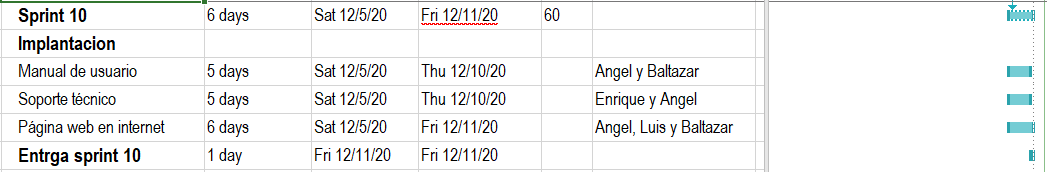












# 5. Referencias

[1]<https://www.idento.es/blog/desarrollo-web/que-es-una-api-rest/#:~:text=Un%20servicio%20REST%20no%20es,API%20REST%20para%20crear%20servicios>.

[2]<https://www.unitag.io/es/qrcode/what-is-a-qrcode>

[3]<https://www.arsys.es/blog/programacion/diseno-web/web-services-desarrollo/>

[4]<https://parkimovil.com/#estacionamientos>

[5]<https://www.elsoldetampico.com.mx/local/video-los-parquimetros-virtuales-4758066.html>

[6]<https://www.eluniversal.com.mx/articulo/techbit/2016/07/30/parquimetros-inteligentes>

[7]<https://parkimovil.com/#navigation>

[8]<https://www.visual-paradigm.com/> [9][https://www.lucidchart.com/pages/es/que-es-un-diagrama-de-flujo-de-datos](https://www.lucidchart.com/pages/es/que-es-un-diagrama-de-flujo-de-datos%20/) [10]<https://carloszr.com/diagrama-de-estado-uml-ejemplo/>

[11]<https://www.wearemarketing.com/es/blog/metodologia-scrum-que-es-y-como-funciona.html#:~:text=La%20metodolog%C3%ADa%20Scrum%20es%20un,equipos%20que%20manejan%20proyectos%20complejos.&text=Esto%20permite%20al%20cliente%2C%20junto,obtener%20ventas%20(Sales%20enablement)>

[12]<https://www.arduino.cc/>

# 6. Lista de figuras y tablas

1. [**Figura 1.0:** Logotipo del proyecto](#u3269gxil3v)
2. [**Figura 2.0:** Jerarquía de Desarrollo Metodologia Scrum.](#z2m84gqto0ku)
3. [**Figura 3.0** Diseño en Balsamiq](#iq6np12483q)
4. [**Figura 3.1:** Diagrama de usos](#nmod6unsbhyt)
5. [**Figura 3.2** Diagrama de Flujo](#4jnf46d00j7i)
6. [**Figura 3.3** Diagrama de estados](#7kkn58sb97x6)
7. [**Figura 3.4** Diagrama de clases](#e1cw45xhi36s)
8. [**Figura 3.5** Diagrama entidad relación](#xckmiv6z1q89)
9. [**Figura 3.6**: Circuito de Arduino del proyecto](#3w8jn9nedcfq)
10. [**Figura 3.7:** Capturas de pantalla de la aplicación móvil](#bnib5n2bus72)
11. [**Figura 3.8:** Checklist implementación del software](#jmucvdkal8go)
12. [**Figura 3.9:** Logo de Android Studio y Java](#pflc5oqtstu6)
13. [**Figura 3.10**: Explicación de los estados del parquímetro](#qix71ftp0159)
14. [**Tabla 1.0:** Prueba de caja negra](#2h29y4rwou84)
15. [**Tabla 2.0:** Prueba de caja blanca](#sltuve1rl0kg)

# 