FACULTAD DE INGENIERIA Y CIENCIAS APLICADAS

DESARROLLO DE UN PROTOTIPO DE UN SISTEMA CONCENTRADOR DE PAGOS DE SERVICIOS PÚBLICOS Y PRIVADOS

AUTOR

LUIS SEBASTIAN MANZANO ALMACHI

AÑO

2020

FACULTAD DE INGENIERIA Y CIENCIAS APLICADAS

DESARROLLO DE UN PROTOTIPO DE UN SISTEMA CONCENTRADOR DE PAGOS DE SERVICIOS PÚBLICOS Y PRIVADOS

Trabajo de Titulación presentado en conformidad con los requisitos establecidos para optar por el título de Ingeniero en Sistemas de Computación e Informática

Profesor Guía

MSc. Carlos Andrés Muñoz Cueva

Autor

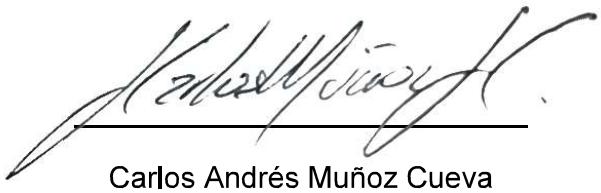
Luis Sebastian Manzano Almachi

Año

2020

DECLARACIÓN DE AUTORÍA DEL PROFESOR GUÍA

"Declaro haber dirigido el trabajo, desarrollo de un prototipo de un sistema concentrador de pagos de servicios públicos y privados a través de reuniones periódicas con el estudiante Luis Sebastian Manzano Almachi, en el semestre 202020, orientando sus conocimientos y competencias para un eficiente desarrollo del tema escogido y dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación".



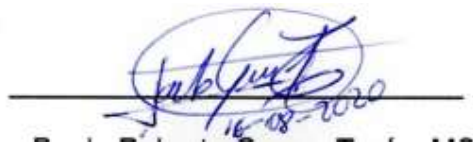
Carlos Andrés Muñoz Cueva

Master en Gerencia de Sistemas

C.I. 1712981511

DECLARACIÓN DE AUTORÍA DEL PROFESOR CORRECTOR

"Declaro haber revisado este trabajo, desarrollo de un prototipo de un sistema concentrador de pagos de servicios públicos y privados, del estudiante Luis Sebastian Manzano Almachi, en el semestre 202020, dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación".



Ing. Paulo Roberto Guerra Terán, MSc.

Master en Software y Sistemas

C.I: 1002856050

DECLARACIÓN DE AUTORÍA DEL ESTUDIANTE

"Declaro que este trabajo es original, de mi autoría, que se han citado las fuentes correspondientes y que en su ejecución se respetaron las disposiciones legales que protegen los derechos de autor vigentes."

Luis Sebastian Manzano Almachi

C.I. 1723730626

AGRADECIMIENTOS

Agradezco primero a dios por guiarme en este largo camino y también a mi familia por el apoyo incondicional a lo largo de la carrera.

DEDICATORIA

Dedico esta tesis a mis padres y hermanos que sin ellos nada de esto habría sido posible.

RESUMEN

Las aplicaciones móviles son muy usadas hoy en día, y el poder tener una que permita el pagar servicios, tanto privados como públicos, obtiene un valor agregado en donde el usuario puede realizarlo sin asistir a las receptores de pago, obviamente con transacciones donde él tiene el control las cuales pueden ser mediante la pasarela de pagos PayPal o sus servicios integrados, pudiendo utilizar tarjetas de débito o crédito.

Existen recepción de pagos en línea, pero estos servicios ofrecen los sistemas bancarios, siendo que estos se los puede utilizar solo si es que se tiene registrado una cuenta bancaria en los mismos así mismo en diferentes cooperativas o mutualistas.

Para desarrollo del prototipo de la aplicación se utilizó las últimas tecnologías hasta el momento, ademas de un framework que es cross-plataform que lo que permite es que la aplicación se podrá utilizar tanto en Android como Apple, enfocándonos en los sistemas operativos más utilizados en el mercado.

Se simuló un web service con su base de datos de empresa para poder simular el pago que se realizará a las diferentes empresas que cuenten con servicios de pago. Está se diseñó en SQL 2017 en conjunto con net core, la cual es consumida a partir de la aplicación.

El desarrollo del prototipo de la aplicación evidenció la simulación de pagos de servicios a partir del consumo de web services que el administrador de la misma puede registrar según el servicio deseado.

ABSTRACT

Mobile applications are widely used today, and being able to have one that allows payment of services, both private and public, add a lot of value where the user can do it without assisting places where receive payments, obviously with the transactions where the he has the control which can be through PayPal payment gateway or its integrated services, being able to use debit or credit cards.

There are online payment services, but these services are offered by banking systems, being these can be used only if you have a bank account registered with them as well in different credit union places.

For the development of the application prototype, the latest technologies can be used so far, a framework that is cross-platform, which allows the application to be used on both Android and Apple, focusing on the most widely used operating systems.

In the market It simulated a web service with its company database in order to simulate the payment that will be made to the different companies that have payment services. It is being implemented in SQL 2017 in conjunction with the central core, which is consumed from the application.

The development of the prototype of the application shows the simulation of payment of services from the consumption of web services that the administrator of the same can register according to the desired service.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1. Capítulo I. Introducción

1.1 Antecedentes

A lo largo de los últimos años, los pagos de servicios ya sea de públicos como luz, agua, teléfono, etc. O privados como seguros, educación, televisión por cable, etc. Han permitido un gran movimiento del comercio en general. Si bien todos estos pagos se los debe realizar en diferentes puntos como bancos, centros de recepción de pagos o incluso en las propias empresas que ofrecen sus diferentes productos.

Existen recepción de pagos en línea, pero estos servicios ofrecen los sistemas bancarios, siendo que estos se los puede utilizar solo si es que se tiene registrado una cuenta bancaria en los mismos así mismo en diferentes cooperativas o mutualistas.

Aplicaciones móviles como tal que permitan pagos de servicios no existen al momento, viendo esto como un impulso a innovar y proceder al desarrollo de una app que se la pueda utilizar en todo momento desde un celular.

Un Estudio eCommerce 2015 elaborado por IAB Spain afirma que el 22% de los usuarios abandona su compra por no encontrar formas de pago adaptadas a sus necesidades. (Mora, 2016)

Es por eso por lo que es vital trabajar para ganarse la confianza de los usuarios, ofreciendo una experiencia de compra más segura, más variada y con más garantías, si cabe, que una tienda física. (Mora, 2016)

Basándonos en la explicación anterior; la propuesta es implementar un prototipo de aplicación móvil que solucione los inconvenientes de realizar un pago con la centralización de todos estos servicios en una misma aplicación.

El sistema permitirá que el usuario pueda registrar tantos servicios como él deseé, todos estos con la respectiva información para poder realizar la acreditación de los mismos.

Para poder realizar todo esto se asumirá que existen las empresas y publicaron previo acuerdo sus web services los cuales son lo que permitirán la acreditación de los respectivos valores que el usuario necesita pagar.

Las aplicaciones de pago han ido creciendo y actualmente se pueden encontrar en cualquier dispositivo, entre algunos de los métodos de pago más utilizados podemos seleccionar los 4 sistemas de pago favoritos entre los usuarios como lo son Google Wallet, Paypal, Apple Pay y Venmo. (Servilia, 2018)

1.2 Alcance

En la actualidad, una gran cantidad de servicios son consumidos por las personas, pero muchos de estos tienen que ser abonados de manera personal en los centros de recepción de pagos o incluso en cajas de los propios almacenes.

Si lo vemos desde un punto general, ahora se utilizan muchos servicios públicos y privados, además de los diferentes pagos de tarjetas. Todo esto conlleva a tener una aplicación centralizada que permita la acreditación de los mismos según el usuario lo necesite.

La creación de este software permitirá que el usuario pueda pagar todos aquellos valores, tanto públicos como privados, optimizando su tiempo para realizar otras tareas de su día a día.

El creciente uso de los estos dispositivos inteligentes ha provocado que desarrollar una app se convierta en una estrategia que puede aportar grandes ventajas. (Webmagic, 2017)

1.3 Objetivos

1.3.1 Objetivo General

Desarrollar un prototipo de aplicación móvil que permita el pago de servicios públicos y privados.

1.3.2 Objetivos Específicos

- Analizar los requerimientos necesarios para el desarrollo del prototipo.

- Diseñar un modelo de base de datos robusto que permita organizar toda la información posible de las funcionalidades de la aplicación.
- Validar la propuesta de solución con la aplicación de pruebas de software.

2. Capítulo II. Marco Teórico

En el presente capítulo se tratará los temas teóricos centrales en los cuales se basa el desarrollo del prototipo de la aplicación móvil. Todo siempre basado en lo más actualizado en lo que desarrollar aplicaciones se trata.

2.1 Metodología

Las metodologías son un aspecto muy importante al momento de desarrollar un proyecto de software, ya que son las que dan un orden y una esquematización de cómo desarrollarlo enfocándose en un producto eficaz y amigable con los usuarios.

2.1.1 Scrum

Scrum es una metodología de enfoque ágil que sirve para desarrollar software, mostrando un desarrollo simple, genérico y por su puesto ágil. Se guía por siempre por el trabajo más importante o de mayor prioridad; si no se tiene suficientes recursos para cualquier tarea, esta se vuelve de menor prioridad diferente a una tarea completada.

Cada tarea se lo realiza como iteración con diferentes intervalos de tiempo que están dentro de un mes calendario. Cada iteración tiene un equipo de trabajo el cual debe ser multifuncional y se encarga de diferentes puntos como lo son el diseño, la construcción y las pruebas del software, todos estos necesarios para completar con los objetivos propuestos y poner el software en producción.

Scrum demostró ser especialmente efectivo en la transferencia iterativa e incremental de conocimiento. Scrum se usa ahora ampliamente para productos, servicios y gestión de la organización matriz. (Schwaber, 2017)

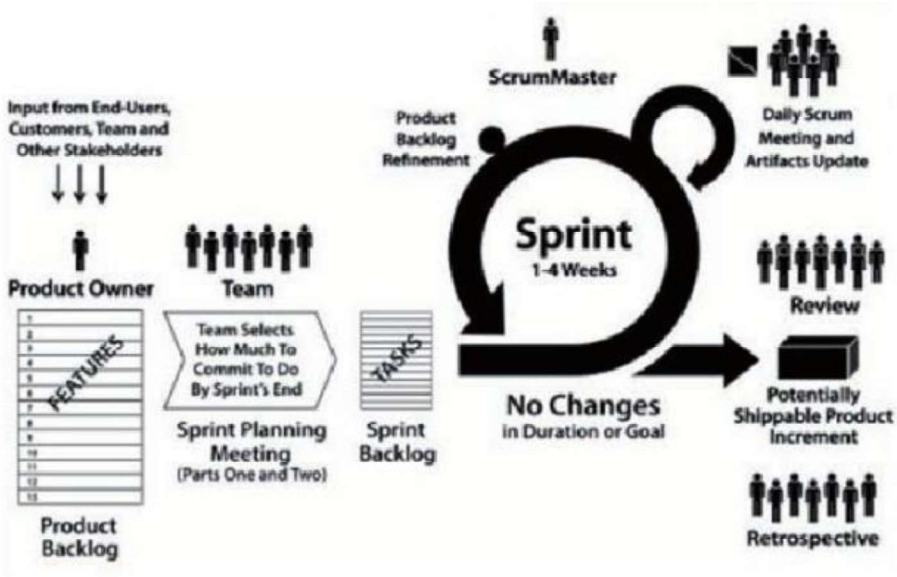


Figura 1. Flujo de scrum

Adaptado de (Pressman, 2010, p.70)

Scrum cuenta con tres pilares, lo cuales son procesos empíricos y son los siguientes:

Transparencia

Los aspectos significativos del proceso deben ser visibles para aquellos que son responsables del resultado. La transparencia requiere que dichos aspectos sean definidos por un estándar común, de tal modo que los observadores compartan un entendimiento común de lo que se están viendo. (Schwaber, 2017)

Inspección

Los usuarios de Scrum deben inspeccionar frecuentemente los artefactos de Scrum y el progreso hacia un objetivo para detectar variaciones indeseadas. Su inspección no debe ser tan frecuente como para que interfiera en el trabajo. Las inspecciones son más beneficiosas cuando se realizan de forma diligente por inspectores expertos en el mismo lugar de trabajo. (Schwaber, 2017)

Adaptación

Si un inspector determina que uno o más aspectos de un proceso se desvían de límites aceptables y que el producto resultante será inaceptable, el proceso o el

material que está siendo procesado deben ajustarse. Dicho ajuste debe realizarse cuanto antes para minimizar desviaciones mayores. (Schwaber, 2017)

2.1.1.1 Roles de Scrum

Los roles de scrum forman de parte importante de los grupos de desarrollo de Scrum. Entre ellos se encuentran el ProductOwner, ScrumMaster, y el equipo de desarrollo. Puede que aparezcan otros roles, pero los importantes y requeridos son los tres nombrados.

Dueño del producto (ProductOwner)

Es el propietario del producto, es el líder del proyecto, la autoridad responsable al momento de las decisiones sobre las funcionalidades, características y el orden de la construcción del producto. El ProductOwner es el intermediario de la comunicación entre todos los participantes del equipo de Scrum.

Cada equipo de Scrum tiene una sola persona que es el ProductOwner la cual será la responsable de cumplir con todos los objetivos del equipo asignado. En cambio, si se tiene una empresa se puede tener un “equipo ProductOwner” debido a que, por diferentes circunstancias, se puede necesitar de otras personas para proporcionar información y orientación.

ScrumMaster

Es la persona que ayuda a todos los involucrados en el proyecto a entender, comprender y adoptar los principios, valores y prácticas de Scrum. Brinda liderazgo y actúa como entrenador en todo el proceso; al mismo tiempo, ayuda a la organización en general en la tarea de gestión de cambios mientras se ejecuta la metodología. Ayuda al equipo a resolver los diferentes problemas del uso de Scrum.

También es el responsable de proteger enteramente a todo el equipo asumiendo el papel de líder para mejorar la productividad del mismo. No tiene la autoridad para tener el control completo del equipo debido a que no es el gerente de proyecto o gerente de desarrollo. Funciona como líder mas no como administrador.

Equipo de desarrollo

Las metodologías tradicionales del desarrollo de software se enfocan en varios puntos como en un arquitecto de software, un programador, tester, dba, diseñador de interfaz de usuario, etc. Scrum muestra el papel que tendrá cada uno de los integrantes del equipo. Este equipo se debe organizar de la mejor manera video el potencial de cada uno. Esta establecido que el equipo suele tener entre cinco y nueve personas, todos ellos con las habilidades necesarias para desarrollar un proyecto de buena calidad.

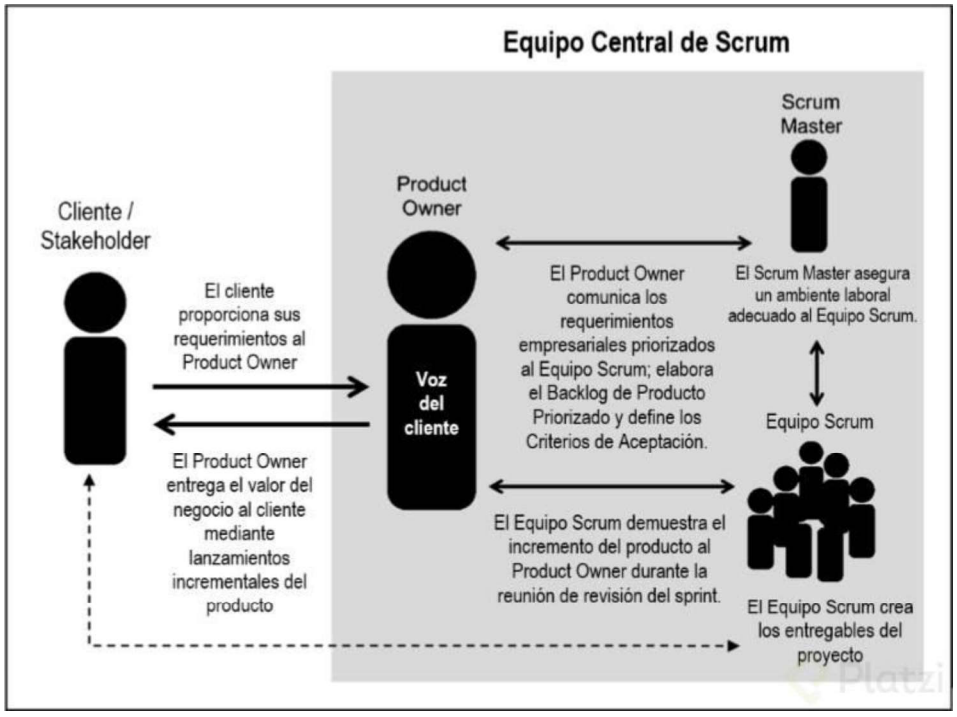


Figura 2. Roles de Scrum

Adaptado de (Platzi, s.f)

2.1.1.2 Actividades y Artefactos de Scrum

Product Backlog

Es aquello que conforma todo el producto que se está desarrollando. Siempre se hace el trabajo más valioso al inicio. La secuencia del trabajo y sus prioridades las realiza el ProductOwner, siendo este el responsable del equipo, siempre

guiado por el aporte del resto del equipo y las partes interesadas del proyecto. Esto es el Product Backlog, una lista ordenada de las tareas a realizar.

El contenido del Product Backlog está conformado por requerimientos, funcionalidades, características, mejoras y correcciones a realizarse en el proyecto que se va a desarrollar. A medida que se va logrando los objetivos, se incrementa el valor del producto, siempre enfocado en la cantidad de usuarios que va a tener.

Sprint

Es el nombre que se le da a un periodo de tiempo en el cual se realizan en iteraciones. Al completarlo tiene como objetivo obtener un entregable o producto de valor tangible para el usuario o el cliente.

Los sprints tienen una fecha de inicio y fin previamente establecida. Por lo general todos ellos deben tener una duración de tiempo similar. Cada uno de ellos comienza cuando su antecesor termina. No se es permitido por ningún motivo el cambio del objetivo o el alcance mientras se ejecuta el sprint, sin embargo, ocurre que por motivos o necesidades comerciales se incumpla esta regla.

Sprint 0

No se lo toma como parte de scrum en sí ya que en la guías, manuales o libros no dicen nada acerca del tema, pero supone una gran cantidad de información que el product owner puede recolectar o realizar diferentes diseños antes de planificar los sprints.

Si bien no puede aportar valor funcional o tangible, si no el de tener que elaborar parte de la arquitectura de los sprints de forma eficiente. También el incluir diferentes temas como el de análisis o diseños previos, trabajos de investigación y elección de herramientas.

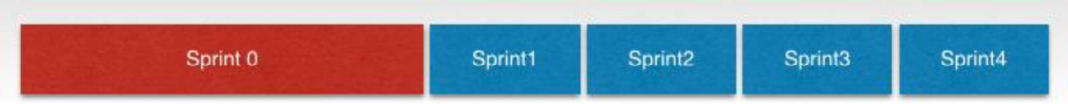


Figura 3. Sprint 0

Adaptado de (Menzinsky, 2015)

Sprint Planning

Un Product Backlog puede mostrar muchos días y meses de trabajo. Para planificar lo más importante a realizar en el siguiente spring se reúnen todos los roles que conforman Scrum como lo son el ProductOwner, el ScrumMaster y el o los equipos de desarrollo. Todos ellos acuerdan objetivos en cada spring a realizar. Se coordina ritmos de trabajo en periodos de tiempo prolongados donde el equipo de desarrollo se sienta cómodo.

El equipo de desarrollo puede adquirir confianza cuando se juntan a cada miembro con características similares para completar un Product Backlog. Aquí es donde se crea una sprint backlog. El equipo mostrara una estimación de tiempo para completar cada tarea.

Planning Poker

Es una técnica de estimación ágil que permite estimar tiempos en diferentes medidas para la dirección de proyectos. Siempre enfocándose en con el coste y tiempo que puede dar el cliente.

Su proceso consiste en planificar el proyecto de software a realizar a partir del equipo, el cual ayudara con los objetivos de cada tarea a realizar, así como sus puntos de vista, todo esto siempre enfocado en valorar el proyecto.

Al seguir esta técnica todos los miembros mantienen una reunión que permitirá estimar las diferentes tareas que se tengan, tanto con las pequeñas como con las grandes inicial muy precisa y una escala poco precisa correspondientemente.

Todo se representará a partir de los numeros de la serie de Fibonacci los cuales manifestarán el grado de complejidad que se utilizará para completar las tareas. Las ventajas que se encuentran al utilizar esta técnica son:

- Cada uno de los integrantes del equipo puede expresar su punto de vista para ir calificando cada tarea.
- Todos son conscientes de las dificultades que se pueden presentar al momento de desarrollar el proyecto.

- El compromiso aumenta ya que cada uno expresa sus opiniones y son participes de los mismos.
- Se muestra una mayor efectividad al momento de estimar las diferentes fechas para la entrega del proyecto.

Daily Scrum

Diariamente se lleva a cabo un daily scrum, el cual consiste en una actividad de inspección; una reunión que permite a cada integrante del equipo de desarrollo el trabajo realizado y por realizar en las siguientes 24 horas. Esto optimiza la proyección del trabajo y puede llegar a reducir horas de tareas en las cuales se puede estar atorado.

Se tiene preguntas definidas, las cuales ayuda al equipo a establecer los siguientes objetivos a lograr. Las cuales son:

- ¿Que se logró desde el ultimo scrum realizado?
- ¿Qué es lo que se va a trabajar para el próximo scrum?
- ¿Se tiene impedimentos u obstáculos que impiden avanzar una tarea?

La idea es que todos entiendan el panorama actual de lo que está ocurriendo con el proyecto, buscando el progreso u modificación que se pueda realizar. Es esencial para administrar el flujo rápido y flexible del trabajo para ayudar al equipo de desarrollo.

Sprint Review

Existen dos actividades importantes que se realizan al final del sprint de inspección y adaptación. Una de ellas es el sprint review.

Su actividad principal es la conversación que se tiene entre sus integrantes los cuales incluyen el equipo de desarrollo, los stakeholders, los patrocinadores, y los miembros autorizados de otros equipos. Se centra en el desarrollo general de las características recién completadas. Todos deben tener visibilidad clara de lo que está ocurriendo para garantizar las soluciones más adecuadas para el negocio.

Por lo tanto, el sprint review representa una oportunidad para inspeccionar y adaptar el producto.

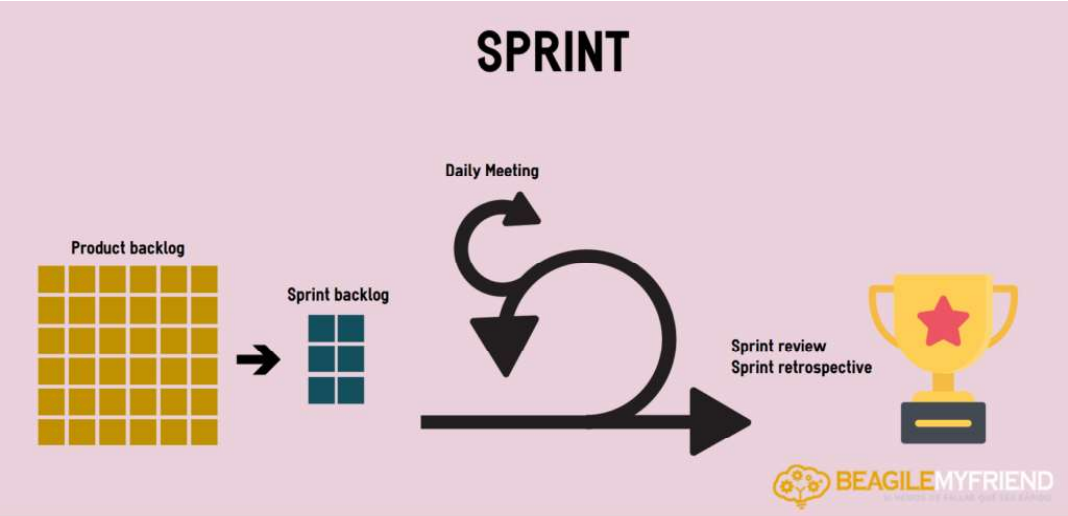


Figura 4. Sprint en Scrum

Adaptado de (Beagile, s.f)

Terminado o “Done”

Se refiere a los resultados del sprint obteniendo un producto entregable, siendo así que todo el equipo scrum acabó con todos los objetivos acordados y que el trabajo es de buena calidad y potencialmente transportable.

Se puede decir que el desarrollo completo de una funcionalidad es que está correctamente diseñada, construida, integrada, probada y documentada. Incluso se permite expresar que cada spring se podría enviar o implementar para clientes internos o externos, a través de la decisión de la empresa.

El envío depende de una decisión comercial a lo que normalmente se realiza las siguientes preguntas:

- ¿Se tiene suficientes funcionalidades o flujos completos para justificar una implementación de un cliente?
- ¿Se puede depender del cliente para que ayudar con cambios dado que se le dio un entregable hace dos semanas?

El producto terminado en si es cuando este se integra a los productos anteriores y correctamente probado y que todo funciona en conjunto.

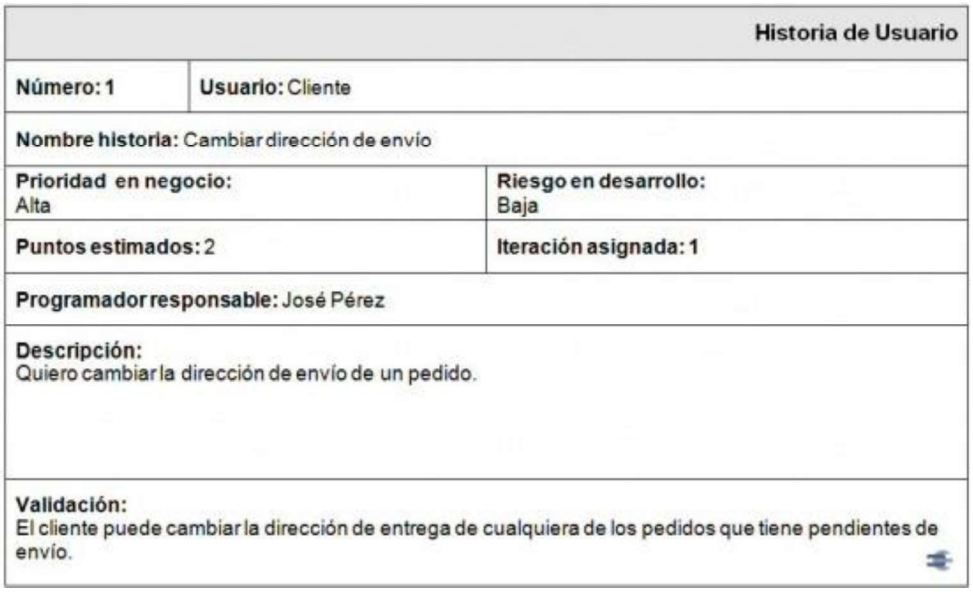
Historias de usuario

Conforma una explicación de manera adecuada que permite obtener el valor comercial del product backlog. Estas se las realizan con una estructura adecuada para que, tanto los técnicos como los empresarios, las comprendan. Son tomados como un enfoque que va en conjunto con los principios de las metodologías agiles siendo efectivas y eficientes.

Se los puede describir de 3 formas:

- Tarjetas o notas

  Son notas que se toman y se anotan en papeles pequeños como recordatorios. Se especifica el rol del usuario, lo que el usuario propone como objetivo y con qué fin.



Historia de Usuario	
Número: 1	**Usuario:** Cliente
Nombre historia: Cambiar dirección de envío	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Baja
Puntos estimados: 2	**Iteración asignada:** 1
Programador responsable: José Pérez	
Descripción: Quiero cambiar la dirección de envío de un pedido.	
Validación: El cliente puede cambiar la dirección de entrega de cualquiera de los pedidos que tiene pendientes de envío.	

Figura 5. Formato historias de usuario

Adaptado de (Scrum Manager, s.f)

Las notas no debe mostrar toda la información que puede tener un requerimiento ya que su espacio es muy limitado, por ello se adjunta oraciones clave que muestren la intención o esencia del requisito.

Ademas es una ayuda para las reuniones y discusiones que se llevaran a cabo entre los stakeholders, el product owner y el equipo de desarrollo.

- Conversaciones

  Todos los datos o detalles de los requerimientos se loas extrae mediante la conversación entre el product owner, el equipo de desarrollo y los stakeholders.

  Existe una conversación inicial donde se comienza a escribir la historia de usuario, después hay otra conversación en donde se refina, otra que se la realiza mientras ocurre la planificación de un sprint y por último cuando se está diseñando, desarrollando y probando el sprint.

  Todas las conversaciones que se realizan se lo toma como un punto base para intercambiar información acerca de los requerimientos, garantizando que estos sean entendidos y examinados por todos donde también se pueden complementar con documentos.

- Confirmación

  En las historias de usuario a partir de la confirmación pueden basarse los criterios de aceptación. El equipo de desarrollo lo usa para desarrollar, comprender y probar de mejor manera el producto. El product owner es el que revisa y acepta si la historia de usuario se ha implementado satisfactoriamente.

  Las pruebas que se realizan no serían las únicas, existen una gran cantidad de pruebas que se llevará a cabo, muchas a un nivel técnico detallado que ni el product owner conocerá. Aunque la idea es bastante intuitiva. Las discusiones sobre las historias con frecuencia se centran en definir ejemplos específicos o comportamientos deseados.

Ademas de las formas de obtener las historias de usuario se tiene también criterios para obtener buenas historias. Se toma el acrónimo INVEST donde serían los siguientes:

- Independientes (Independent)

  Las historias de usuario deben ser independientes o no depender de muchas historias entre sí. Primeramente, se tiene que iniciar con las historias dependientes siempre teniendo en cuenta que puede ser difícil

el priorizar cuales de ellas se van a trabajar en cada sprint. El objetivo principal no es eliminar las dependencias entre historias, si no el minimizar las dependencias.

- Negociables (Negotiable)

  Los detalles de las historias de usuario debe ser negociables. Si bien, no son parte de un documento adelantado, son señaladores que se utilizaran en las conversaciones donde se negociaran los detalles, la esencia en sí de la funcionalidad empresarial que quiere y por qué se quiere, dejando libre para la reunión entre el product owner, los stakeholders y el equipo de desarrollo.

  Al ser negociable las historias puede evitar problemas con los asociados, ya que se mantendrá un dialogo frecuente con el equipo de desarrollo y se puede asegurar que se aclararan muchas dudas que pueden surgir a partir del desarrollo del proyecto.

- Valiosos (Valuable)

  Las historias deben ser valiosos para un cliente y usuario. Los clientes seleccionan y pagan el producto. Si la historia de usuario no es valiosa para nadie, no es parte del product backlog. Para que haya una historia de usuario el product owner debe entender por qué está pagando y el valor total en última instancia.

  Se tiene que identificar las historias valiosas a partir de las tareas asociadas al negocio y la perspectiva del product owner. No todas las historias son independientes ni tampoco son totalmente negociables, pero todas deben ser valiosas.

- Estimados (Estimatable)

  Las historias de usuario deben ser estimables por el equipo de desarrollo que es el que diseña, desarrolla, construye y prueba. La estimación puede ayudar a proporcionar el tamaño, el esfuerzo y el costo obteniendo información que será procesada por el equipo de scrum. Las historias más grandes son las que más recursos necesitan y por ello cuestan más dinero.

El product owner es el que debe saber el costo de una historia para determinar la prioridad de la misma en el product backlog. El equipo de scrum a partir del tamaño de la historia puede necesitar un refinamiento o convertirlo en historias más manejables.

- Tamaño Apropiado (Small)
  Las historias de usuario deben tener un tamaño adecuado para trabajar en ellas. A partir de la duración del sprint es que se trabajaran las historias de usuario, si realizamos historias de usuario de gran tamaño en sprints pequeños, se tiene la gran probabilidad de no terminar la historia.
  Siempre hay que buscar la manera de manejar historias de usuario pequeñas para que puedan ser desarrolladas por el equipo, se podrán terminar a tiempo y se podrá continuar con el siguiente sprint.
- Comprobables (Testable)
  Las historias deben ser comprobables, es decir, tener buenos criterios de aceptación de acuerdo a condiciones de satisfacción. A partir de las diferentes pruebas que se realizan se puede obtener detalles importantes de las historias de usuario antes de que el equipo de desarrollo pueda estimar los tiempos.
  En ocasiones puede haber una historia que el propietario del producto considere valiosa, pero puede que no haya una forma práctica de probarla. Aunque los criterios de aceptación pueden ser claros, existen pruebas que no se pueden ejecutar en tiempos de producción que se lo puede tomar como demostración de que se ha cumplido este nivel de tiempo de actividad.

2.2 Herramientas para el desarrollo de la aplicación

A partir de las herramientas de desarrollo es que se desarrolla una aplicación, estas deben ser compatibles entre sí para obtener un producto final correcto que cumpla con los requerimientos necesitados.

2.2.1 Base de Datos

La base de datos que se va a utilizar es Microsoft Sql Server 2017 debido a que primeramente trabaja con SQL el cual es un lenguaje estructurado de consultas

y apoya a la utilización de las bases de datos relacionales como lo es Sql server. Este software proporciona niveles altos de rendimiento que permite una mayor rapidez en las transacciones a realizar, ademas contiene una seguridad muy adecuada para el tipo de aplicación a desarrollar.

La forma en que se manejan los datos en la actualización del 2017, la velocidad y la latencia en las funciones a realizar es la razón de elegir este producto. Ademas de su fácil conexión con las diferentes herramientas de desarrollo que se utilizaron para la creación de la aplicación a realizar.

Sql Server 2017 cuenta con mejoras en el uso de la memoria para optimizar el uso de tablas, compilación nativa de los procedimientos almacenados y la definición de funciones por el usuario para compilación nativa enfocándose en la opción de migraciones. También mejora los diseños In-Memory OLTP en donde se encuentra la optimización para el acceso a datos de la memoria principal, nuevas herramientas para la migración, agilitar el procedimiento de lógica empresarial, una alta disponibilidad y las mejoras en procedimientos transaccionales.

Se simulará una base de datos replicando lo más cercano posible a la de una empresa que tiene un módulo de pago de servicios. Lo cual con la ayuda sql server 2017 se lo va a poder realizar.

2.2.2 Lenguaje de programación

Se utilizará c# debido a la posibilidad de crear diferentes aplicaciones para propósito general como pueden ser web apps y para móviles.

Este lenguaje es basado en C y C++, siendo este un lenguaje de alto nivel orientado a objetos de amplia gama, capaz de ser utilizado para desarrollar aplicaciones en Microsoft .NET.

Su objetivo principal es de obtener todas las posibilidades para un desarrollador el poder crear aplicaciones más complejas con mayor facilidad y rapidez e incluso eliminando diferentes errores que pueden ocurrir como pueden ser:

- Evitar que el usuario lleve el manejo manual del uso de la memoria.
- Inicialización dinámico de objetos a cero.

- Unifica el sistema de tipos viéndose estos como un objeto.

2.2.3 Frameworks de desarrollo

2.2.3.1 Net Core

ASP.NET Core es un marco de trabajo web creado por Microsoft para desarrollar aplicaciones web, APIs y microservicios. Usa patrones comunes como MVC (Modelo-Vista-Controlador), inyección de dependencias y una canalización de solicitudes formada por middleware (Barbettini, 2018)

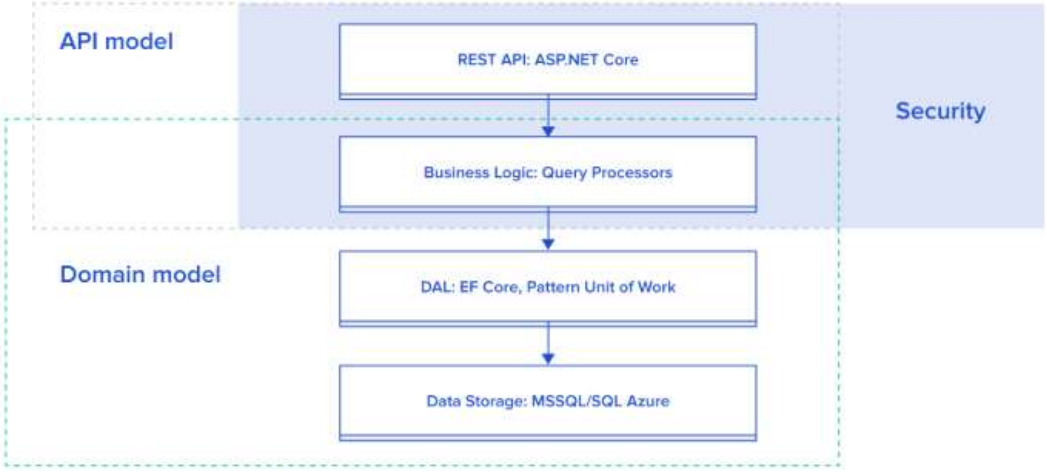


Figura 6. Capas de api común ASP.NetCore

Adaptado de (Imangulov, s.f)

Se ejecuta con la ayuda de Microsoft .NET, y tiene muchos beneficios a diferencia de los diferentes marcos de trabajo entre ellos:

- Rapidez: La compilación del código .NET se ejecuta más rápido nen comparación de Javascript o Ruby. Optimizado con multihilos y tareas asíncronas.
- Ecosistema: Los miles de paquetes de nuget que tiene permiten realizar cualquier cosa que se pueda idear.
- Seguridad: Se toma muy enserio la seguridad creando fundamentos seguros. Previene la falsificación de sitios cruzados para realiza ataques/

Se utilizará la versión 3.1 para desarrollar el servicio de la aplicación móvil. Entre las nuevas actualizaciones están el SameSite que es una actualización de

cookies que utilizan las nuevas versiones de los navegadores, librerías obsoletas fueron quitadas y se actualizaron paquetes propios de Net Core.

2.2.3.2 Xamarin Forms

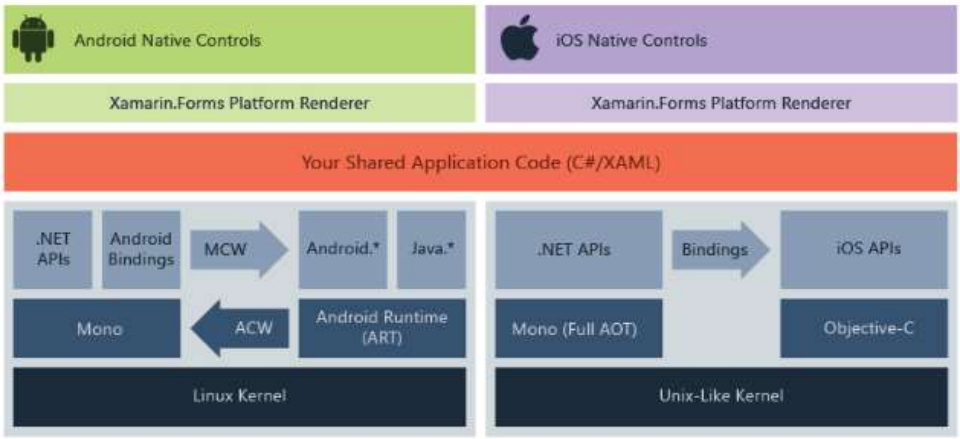


Figura 7. Como trabaja Xamarin Forms

Adaptado de (Microsoft, 2019)

Es un framework móvil de código abierto, que permite desarrollar aplicaciones para Android, IOS y Windows ya que comparten código y lógica empresarial. Se utiliza C# para su codificación y código de etiquetas xaml para las vistas.

El patrón que normalmente se utiliza es Model-View-ViewModel (MVVM). Las app que se desarrollan en Xamarin Forms contienen un biblioteca estándar de .NET que contiene las vistas la lógica de negocios ya sean modelos, servicios, etc.

Como todo entorno de desarrollo de C#, tiene una gran cantidad de paquetes NuGet que ayuda con la implementación de diferentes funcionalidades para diferentes aplicaciones.

2.2.3.3 Visual Studio

Es un IDE (Entorno de desarrollo integrado) que permite depurar, editar y compilar código ademas de publicar la aplicación. También cuenta con una gran cantidad de características que ayudan a desarrollar software de una manera más sencilla. Para depurar las aplicaciones se utiliza el propio emulador de visual studio en el caso de Android de Windows Mobile. En el caso de IOS se necesita

de una MAC, ya que se necesitan herramientas de desarrollo que solo este dispositivo proporciona.

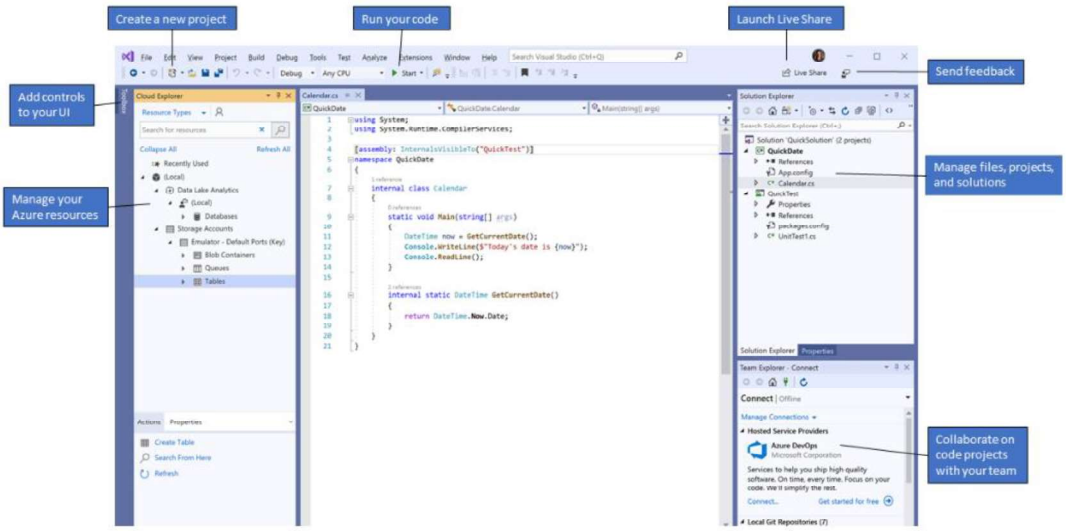


Figura 8. Ventana Visual Studio 2019

Adaptado de (Microsoft, 2019)

Se tiene las partes enmarcadas, en donde se ve del lado derecho el explorador de soluciones donde se tiene todos los archivos de código que forman parte del proyecto y el Team Explorer, donde se ubican las tecnologías de control de versiones. El editor es donde se codifica las funcionalidades del software, ademas de diseñar las vistas y se ubica en el centro de la pantalla. Por último, de lado izquierdo se tiene los diferentes controladores que se puede usar para el diseño de las vistas.

2.2.3.4 Web API Service

API (Interfaz de Programación de Aplicaciones). Tradicionalmente, de forma local, una API se expone a través de archivos DLL. Una API se expone a través de Servicios Web que permiten que las aplicaciones cliente obtengan y realicen operaciones con los datos que el servicio expone (Muñoz, 2017)

REST es un acrónimo de Representational State Transfer (Transferencia de Estado Representacional) y es un estilo de arquitectura de software para crear

APIs que utilicen HTTP como su método de comunicación subyacente (Muñoz, 2017)

Los Web API utilizan URL para las diferentes peticiones que se pueden realizar obteniendo resultados tipo JSON (JavaScript Object Notation), el cual es mostrado como un formato para intercambio de datos. El formato JSON esta enteramente basado en javascript, pero se hace muy familiar usarlo en C, C++, Java, Python, etc.


```
[{"Id":1,"Name":"Chai","Category":"Beverages","Price":18.0000},{"Id":2,"Name":"Cha
ng","Category":"Beverages","Price":19.0000},{"Id":3,"Name":"Aniseed
Syrup","Category":"Condiments","Price":10.0000}]
```


Figura 9. Notas de historias de usuario

Adaptado de (Muñoz, 2017)

2.2.4 Servidor de publicaciones

2.2.4.1 IIS (Internet Information Service)

Es un servidor web creado por Microsoft que permite publicar servicios web, tanto locales como remotos. La versión que se utiliza para el proyecto es la IIS 10.0, la cual maneja características como FTP, SMTP, entre otros. La publicación se alojará de manera local, permitiendo que la aplicación opere de la misma manera.

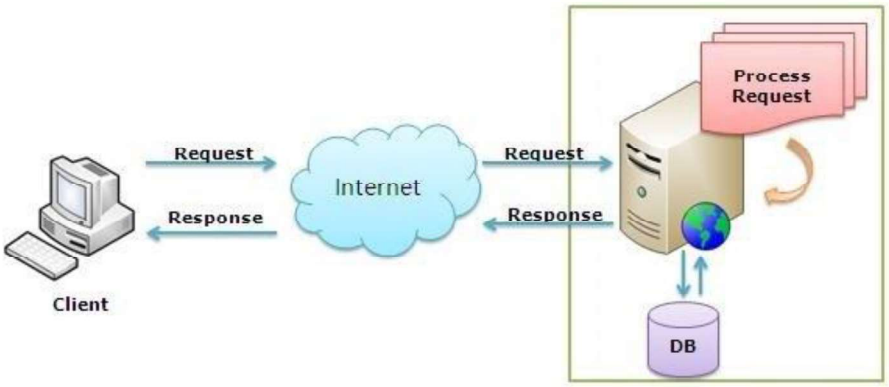


Figura 10. Arquitectura de IIS (Internet Information Service)

Adaptado de (Quora, 2016)

Entre sus principales características se encuentran:

- Seguridad: Acepta todo tipo de capas de seguridad, donde su principal función es proteger los datos que entran y salen del servicio.
- Contenido: Acepta diferentes tipos de archivos generados por varias aplicaciones como lo pueden ser CGI, PHP, PERL, .NET, etc.
- Comprensión: A partir de sus dos módulos de contenido estático y dinámico se comprimen de manera correcta.
- Almacenamiento caché: Obtiene una memoria interna en donde se guardan diferentes datos que permiten mejorar el rendimiento de diferentes aplicaciones que sean publicadas en su servidor, obteniendo mejores tiempos de respuesta.
- Registro y diagnóstico: Trabaja directamente alojando las acciones de las diferentes herramientas que contiene la aplicación web, funcionando como un capturador de eventos monitoreando las solicitudes tanto de entrada como de salida.

3. Capítulo III. Análisis y Diseño

En el presente capítulo se explicará el diseño de simulación de una base de datos de una empresa pública, así como la de una empresa privada y la de la aplicación que simulará los pagos. También se enfocará en la creación del product backlog con las historias de usuario con los sprints que se enfocaran en el desarrollo del proyecto.

3.1 Sprint 0

Para el sprint 0 se realizará lo que es el diseño de diagramas que se utilizará en la aplicación, así como el diseño que se utilizará en la misma y también se desarrollará el login de la aplicación siendo esta funcionalidad separada del product backlog.

3.1.1 Diseño de base de datos para la aplicación

Se va a obtener dos diseños de base de datos en donde el primero servirá para la simulación de consultas a servicios que pertenezcan a las diferentes empresas, ya sea pública o privada, y la segunda que será para los datos de la aplicación.

3.1.1.1 Base de datos para empresa pública y privada

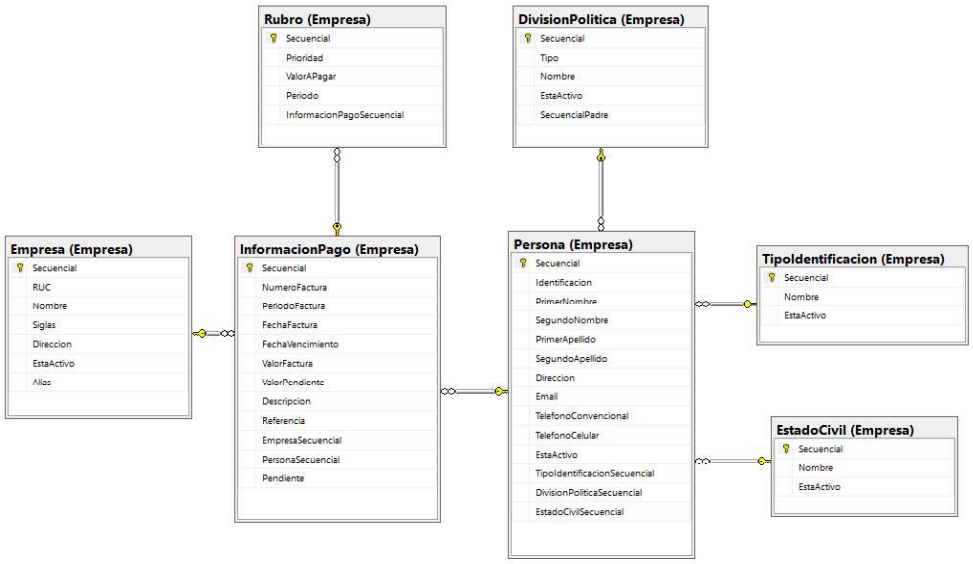


Figura 11. Base de datos diseño empresa privada

3.1.1.2 Base de datos aplicación pago de servicios

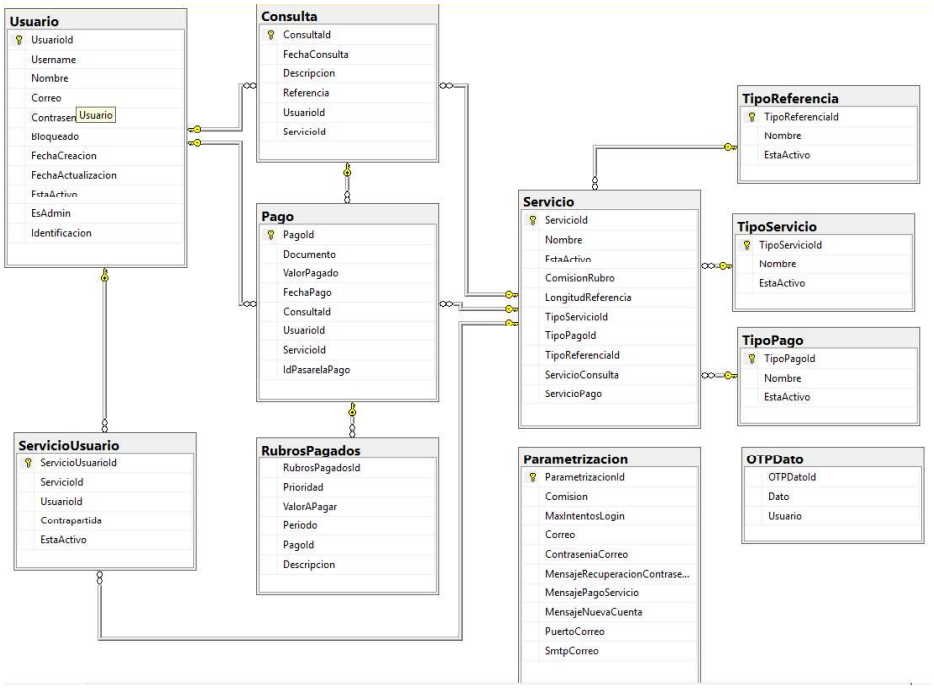


Figura 12. Base de datos diseño para app móvil

3.1.2 Diseño de la aplicación

Figura 13. Colores de la aplicación.

Figura 14. Logo y nombre de la aplicación.

3.1.3 Funcionalidad login

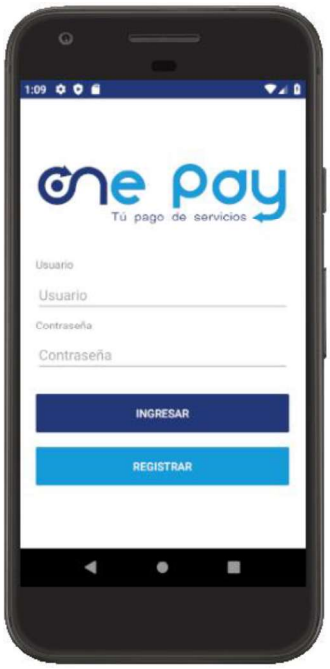


Figura 15. Pantalla login de app móvil.

3.1.3 Menú administrador

Figura 16. Pantalla menú administrador app móvil.

3.1.4 Menú Usuario

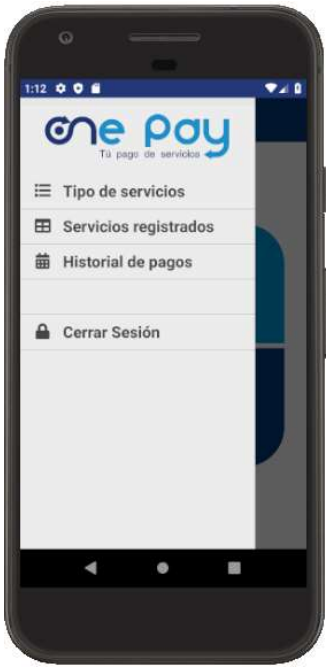


Figura 17. Pantalla menú usuario app móvil.

3.2 Product Backlog

En la tabla se muestra las historias de usuario que se recolectaron para el desarrollo de la aplicación, a partir de ellas se comenzará a crear las funcionalidades, ya sea para el móvil como para los servicios. Las historias se las realizó a partir de usuarios que mostraron interés de utilizar la aplicación y todos los beneficios que esta brinda.

La estimación se lo realiza a partir de parámetros de complejidad los cuales van a representar las diferentes actividades que se va a realizar para la planificación. A partir del Planning Poker es que se lo va a realizar donde tiene su propia escala de dificultar en donde se guía a partir de proporcionalidad donde el valor de la estimación equivale al doble de la equivalencia en días, siendo que si la estimación equivale a 8 el tiempo en días de desarrollo será 4 días, es decir 32 horas

Tabla 1

Product Backlog

Id	Historia de Usuario	Prioridad	Estimación	Horas
1	Como administrador quiero poder registrar nuevos datos de servicios para mostrar los datos registrados en la aplicación.	Alta	8	24
2	Como administrador quiero poder realizar el mantenimiento de catálogos para mostrar los datos registrados en la aplicación.	Alta	8	24
3	Como administrador quiero poder agregar parametrizaciones para afectación directa con la aplicación.	Alta	8	24
4	Como administrador quiero poder ingresar datos de servicios para generar un reporte de consultas y pagos.	Alta	8	24
5	Como usuario quiero poder registrar un usuario y contraseña para poder ingresar a la aplicación.	Alta	8	24
6	Como usuario quiero poder ingresar mis datos para realizar una consulta de información de pago de servicio.	Alta	13	53

7	Como usuario quiero poder registrar mis pagos frecuentes para realizar un pago más rápido.	Alta	8	24
8	Como usuario quiero poder conectarme con mis datos de PayPal para poder realizar el pago a través de esta plataforma.	Alta	13	53
9	Como usuario quiero que me llegue un código OTP para verificar el pago.	Alta	13	53
10	Como usuario quiero poder ingresar mis datos para realizar el pago de un servicio.	Media	13	53
11	Como usuario quiero que me llegue la información del pago realizado a mi mail para poder revisar y validar el mismo.	Media	8	24
12	Como usuario quiero poder ingresar mi huella para ingresar a la aplicación.	Media	5	21
13	Como usuario quiero poder ingresar mis datos para revisar el historial de pagos.	Baja	8	24

3.3 Arquitectura de One Pay

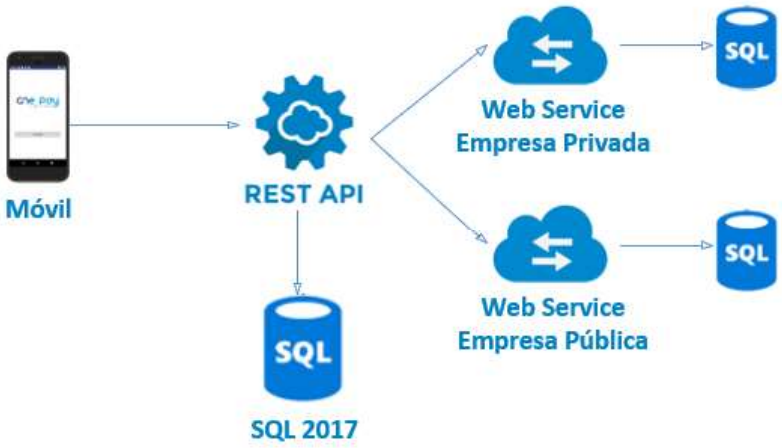


Figura 18. Arquitectura de la aplicación.

4. Capítulo IV. Desarrollo de la aplicación

En el presente capítulo se mostrará el desarrollo de la aplicación, cada funcionalidad ha sido separada por sprints los cuales son una guía de cómo se desarrollará la aplicación, todo esto se ejemplificará por medio de gráficos.

4.1 Sprints

4.1.1 Sprint 1

Tabla 2

Product Backlog Sprint 1

Id	Historia de Usuario	Prioridad	Estimación	Horas
1	Como administrador quiero poder registrar nuevos datos de servicios para mostrar los datos registrados en la aplicación.	Alta	8	24
2	Como administrador quiero poder realizar el mantenimiento de catálogos para mostrar los datos registrados en la aplicación.	Alta	8	24
3	Como administrador quiero poder agregar parametrizaciones para afectación directa con la aplicación.	Alta	8	24
4	Como administrador quiero poder ingresar datos de servicios para generar un reporte de consultas y pagos.	Alta	8	24

A través del product backlog del spring 1 se desarrollará todo lo concerniente al lado del administrador, en de donde se abarca las historias de usuario del 1 al 4. El sprint dura 16 días, alrededor de 128 horas, en ser terminado.

Las historias de usuario que forman parte del sprint son:

- Mantenimiento de servicios: Donde el administrador puede agregar los servicios que los usuarios de la aplicación pueden utilizar para realizar sus pagos.
- Mantenimiento de catálogos: En general será para que el administrador maneje los catálogos que se van a utilizar en toda la aplicación, ya sea categorías o para realizar diferentes validaciones.
- Mantenimiento de parametrizaciones: La aplicación tendrá parametrizaciones que se podrán crear o alterar a partir de las necesidades de la misma.
- Generación de reportes: El administrador podrá revisar los servicios que más se han consultado, así mismo que más se han pagado a partir de la aplicación.

Tabla 3

Historia de Usuario 1

Usuario: Administrador	
Nombre historia: Registrar servicios	
Nivel de Prioridad: Alta	Iteración: 1
Puntaje estimado: 8	Riesgo: Baja
Descripción: Como administrador quiero poder registrar nuevos datos de servicios para mostrar los datos registrados en la aplicación.	
Criterio de aceptación: El administrador debe ingresar el nombre, el tipo de pago, el tipo de referencia, la longitud de la referencia, el servicio de consulta y pago, el tipo de servicio, si se cobra comisión por el pago y si está activo.	

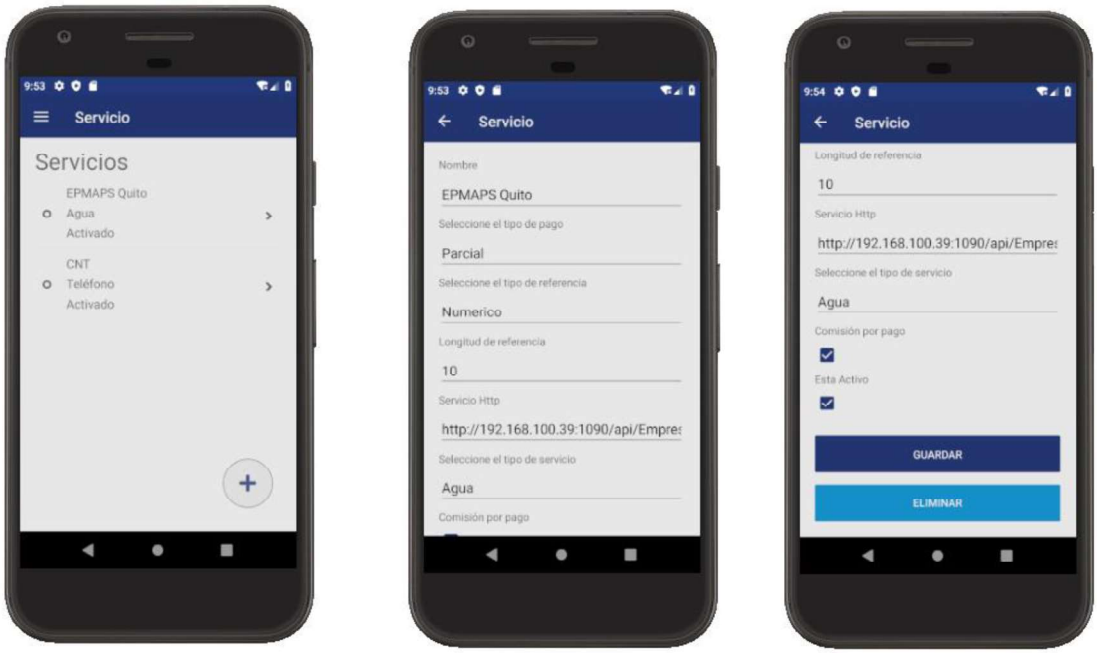


Figura 19. Pantallas mantenimiento de servicios app móvil.

Tabla 4

Historia de Usuario 2

Usuario: Administrador	
Nombre historia: Realizar mantenimiento de catálogos.	
Nivel de Prioridad: Alta	Iteración: 1
Puntaje estimado: 8	Riesgo: Baja
Descripción: Como administrador quiero poder realizar el mantenimiento de catálogos para mostrar los datos registrados en la aplicación.	
Criterio de aceptación: El administrador debe ingresar para los catálogos los datos nombre y si está activo.	

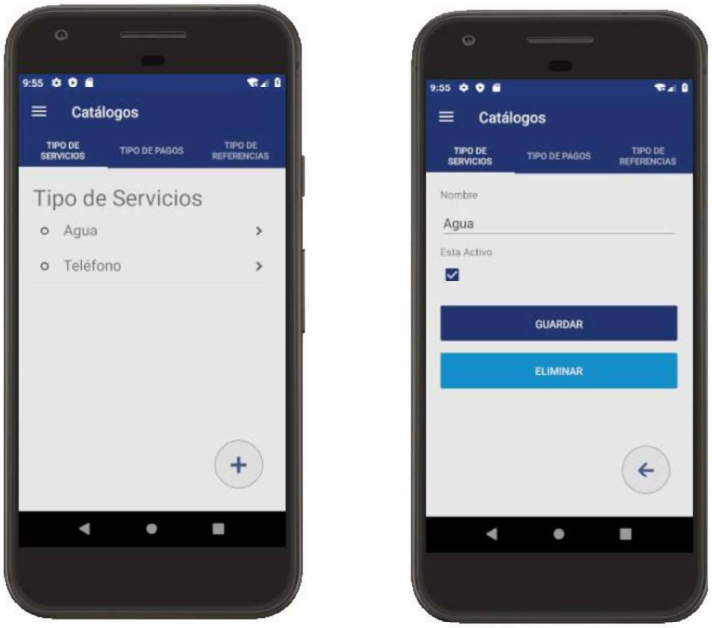


Figura 20. Pantallas mantenimiento de catálogo tipos de servicios app móvil.

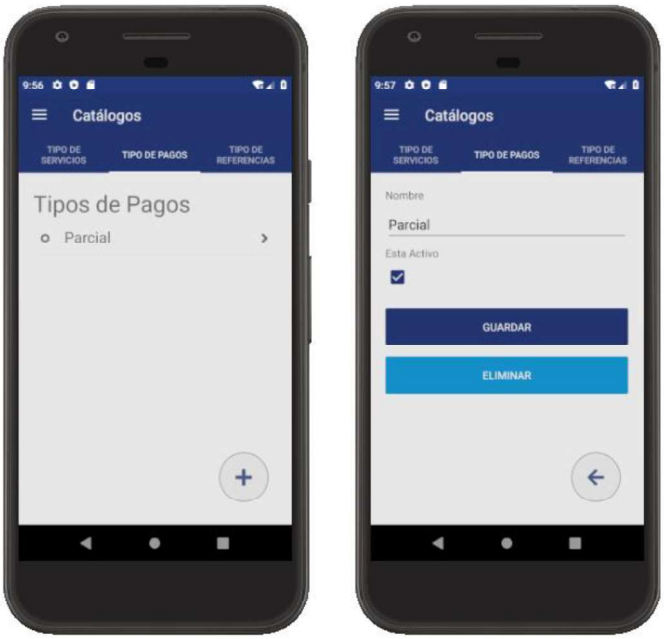


Figura 21. Pantallas mantenimiento de catálogo tipos de pagos app móvil.

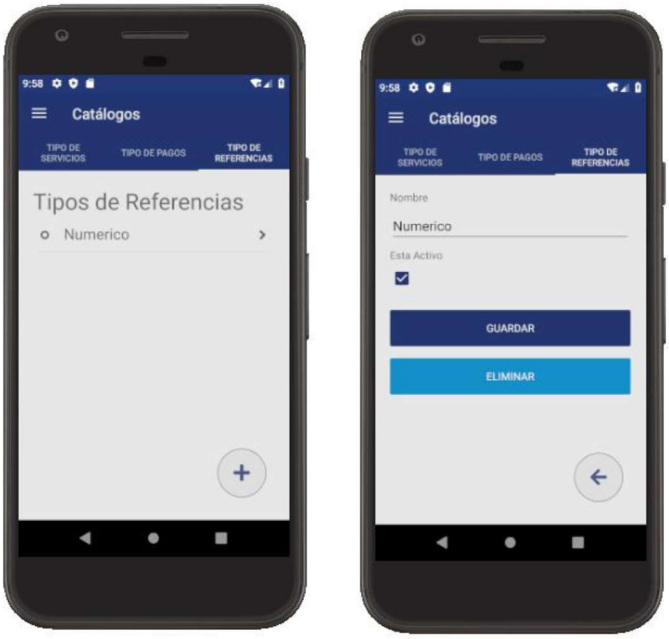


Figura 22. Pantallas mantenimiento de catálogo tipos de referencias app móvil.

Tabla 5

Historia de Usuario 3

Usuario: Administrador	
Nombre historia: Mantenimiento de parametrizaciones de la aplicación	
Nivel de Prioridad: Alta	Iteración: 1
Puntaje estimado: 8	Riesgo: Baja
Descripción: Como administrador quiero poder agregar parametrizaciones para afectación directa con la aplicación.	
Criterio de aceptación: El administrador tiene que ingresar los datos de parametrización comisión, intentos de login, correo, contraseña de correo, puerto de correo, smtp de correo, correo para pago realizado, para recuperación de cuenta y nueva cuenta.	

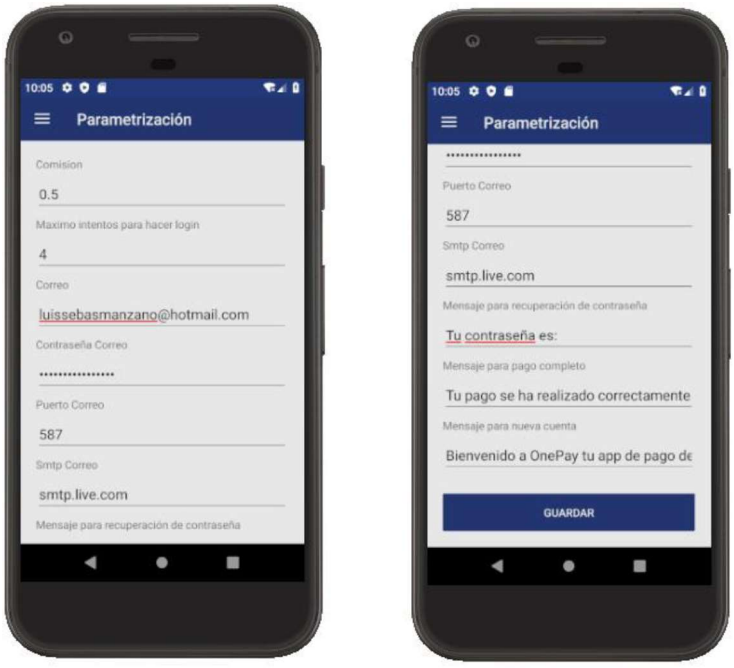


Figura 23. Pantalla de parametrizaciones app móvil.

Tabla 6

Historia de Usuario 4

Usuario: Administrador	
Nombre historia: Mostrar historial de consultas y pagos.	
Nivel de Prioridad: Alta	Iteración: 1
Puntaje estimado: 8	Riesgo: Baja
Descripción: Como administrador quiero poder ingresar datos de servicios para generar un reporte de consultas y pagos.	
Criterio de aceptación: El administrador debe agregar el tipo de servicio, el servicio un rango de fechas y seleccionar el tipo de reporte, de pagos o consultas.	

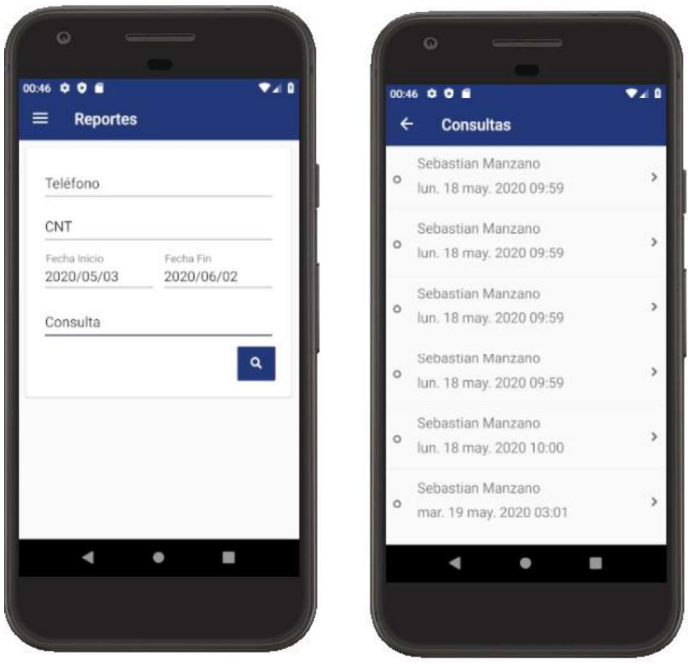


Figura 24. Pantalla de reportes consulta app móvil.

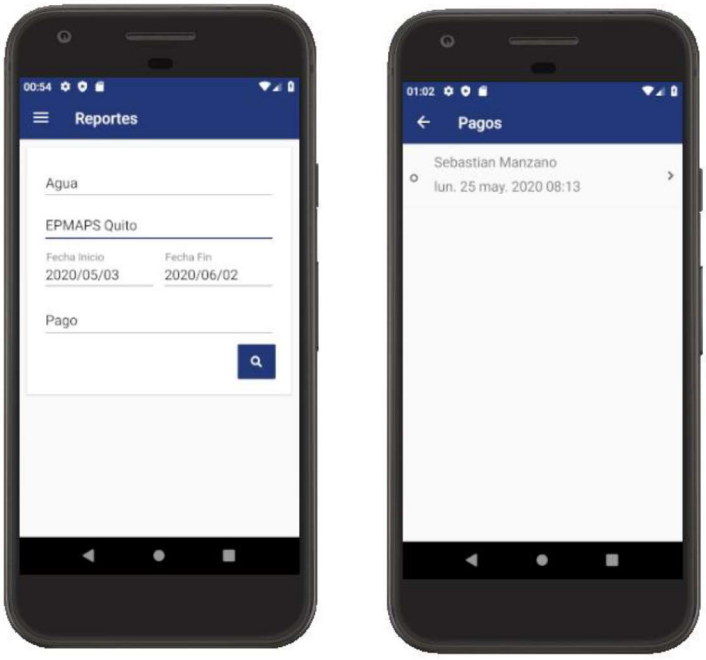


Figura 25. Pantalla de reportes pago app móvil.

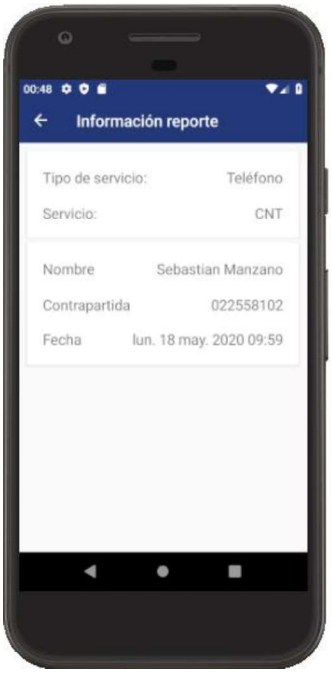


Figura 26. Pantalla de reportes información app móvil.

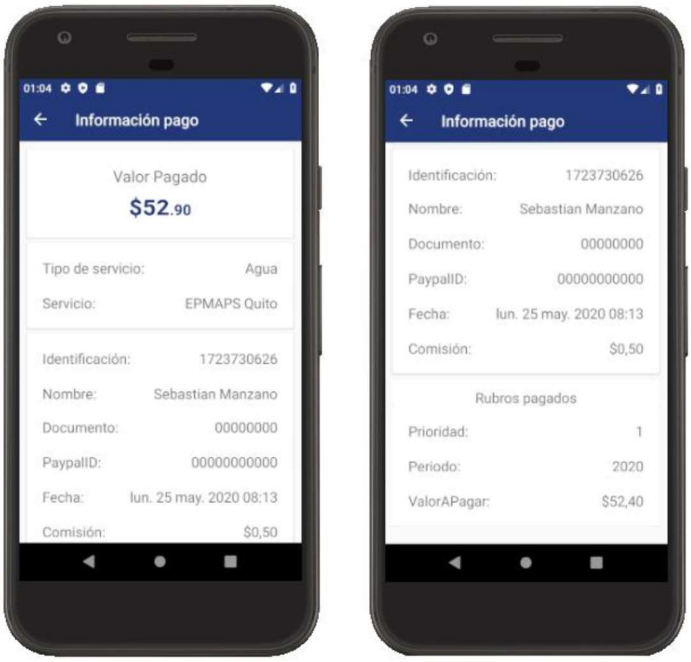


Figura 27. Pantalla de reportes pago app móvil.

4.1.2 Sprint 2

Tabla 7

Product Backlog Sprint 2

Id	Historia de Usuario	Prioridad	Estimación	Horas
5	Como usuario quiero poder registrar un usuario y contraseña para poder ingresar a la aplicación.	Alta	8	24
6	Como usuario quiero poder ingresar mis datos para realizar una consulta de información de pago de servicio.	Alta	13	53
7	Como usuario quiero poder registrar mis pagos frecuentes para realizar un pago más rápido.	Alta	8	24
8	Como usuario quiero poder conectarme con mis datos de PayPal para poder realizar el pago a través de esta plataforma.	Alta	13	53

Para el sprint 2 se desarrollará la primera parte que el usuario podrá realizar en la aplicación, perteneciendo las historias de usuario de 5 a la 9. Se ha estimado una duración de 24 días, alrededor de 192 horas, para dar por terminado.

Las historias de usuario que forman parte del sprint son:

- Registro de usuarios: El usuario podrá registrarse en la aplicación ingresando su propio usuario y contraseña, donde se manejará sus datos.
- Consulta de servicios: El usuario podrá consultar su información de pago a partir de los servicios registrados por el administrador.

- Registro de servicios: El usuario podrá registrar sus servicios para realizar consultas y pagos a mayor velocidad.
- Conexión con PayPal: Para realizar el pago se podrá registrar en Paypal y realizar el pago del servicio.

Tabla 8

Historia de Usuario 5

Usuario: Usuario	
Nombre historia: Registro de usuario	
Nivel de Prioridad: Alta	Iteración: 2
Puntaje estimado: 8	Riesgo: Baja
Descripción: Como usuario quiero poder registrar un usuario y contraseña para poder ingresar a la aplicación.	
Criterio de aceptación: El usuario puede registrarse, administrar su cuenta, y hacer un login con usuario y contraseña.	

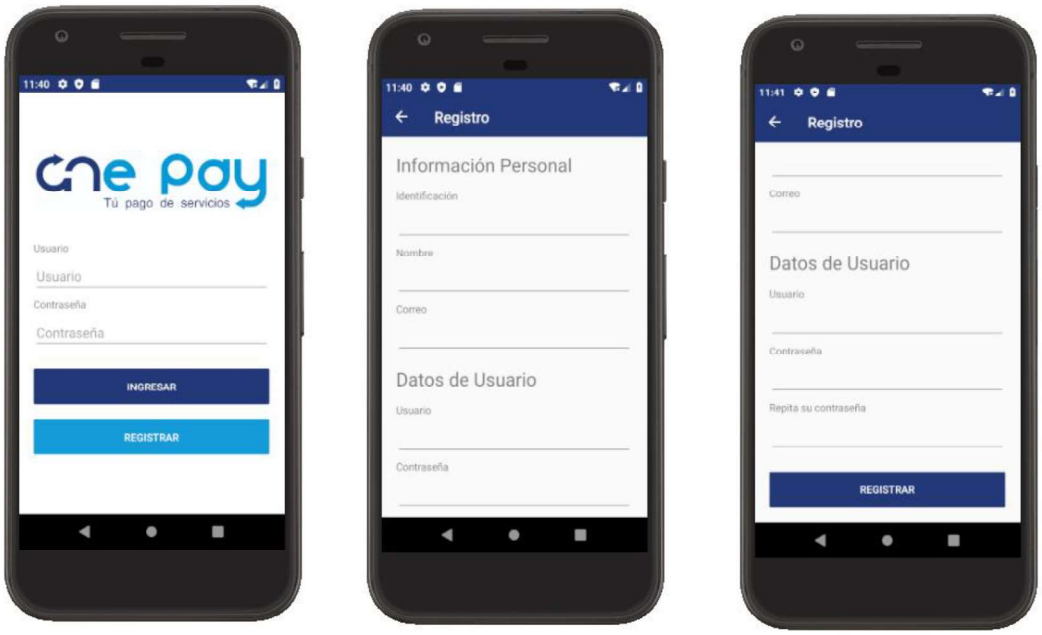


Figura 28. Pantalla de registro de app móvil.

Tabla 9

Historia de Usuario 6

Usuario: Usuario	
Nombre historia: Realizar consulta de servicios	
Nivel de Prioridad: Alta	Iteración: 2
Puntaje estimado: 13	Riesgo: Baja
Descripción: Como usuario quiero poder ingresar mis datos para realizar una consulta de información de pago de servicio.	
Criterio de aceptación: El usuario debe seleccionar los tipos de servicios principales o ir a elegirlo manualmente, ademas del servicio y la contrapartida.	

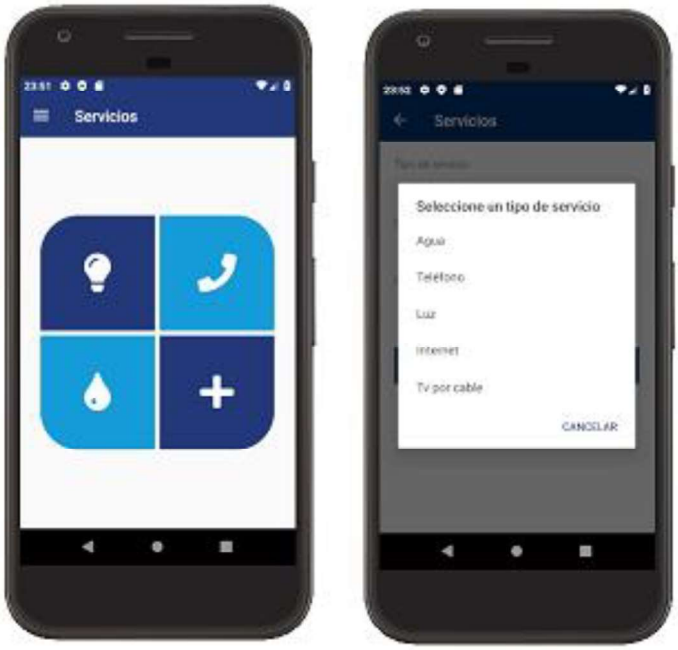


Figura 29. Pantalla información de consulta de pago app móvil.

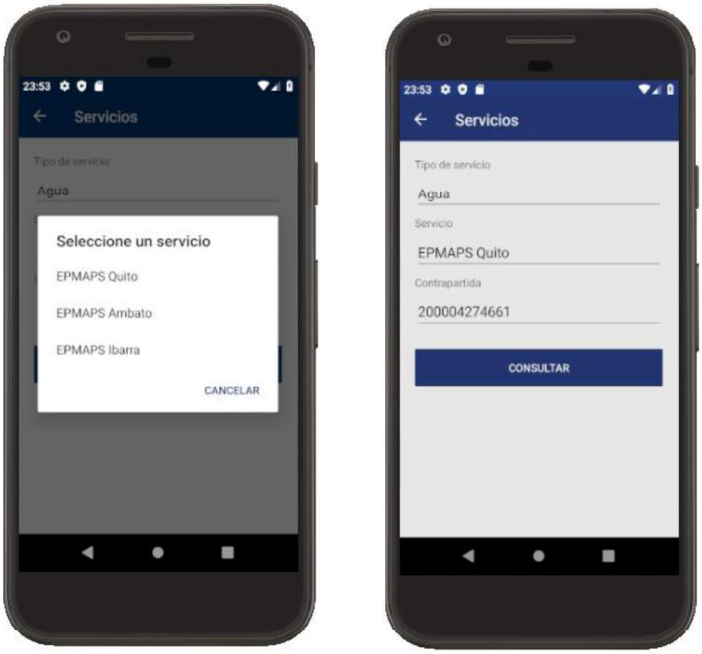


Figura 30. Pantalla información consulta de pago app móvil.

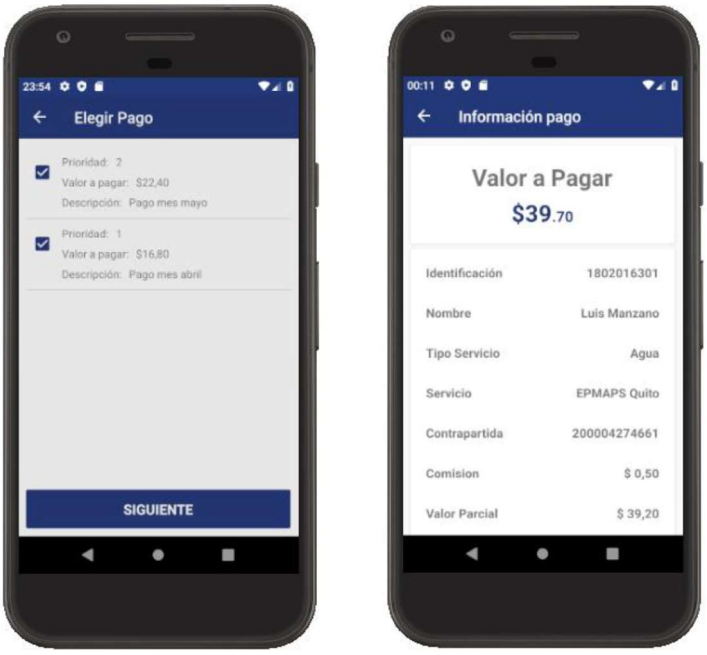


Figura 31. Pantalla consulta de información elegir pago app móvil.

Figura 32. Pantalla consulta de información de pago app móvil.

Tabla 10

Historia de Usuario 7

Usuario: Usuario	
Nombre historia: Registro de servicios frecuentes según usuario	
Nivel de Prioridad: Alta	Iteración: 2
Puntaje estimado: 8	Riesgo: Alta
Descripción: Como usuario quiero poder registrar mis pagos frecuentes para realizar un pago más rápido.	
Criterio de aceptación: El usuario debe seleccionar el tipos de servicio el servicio y la contrapartida.	

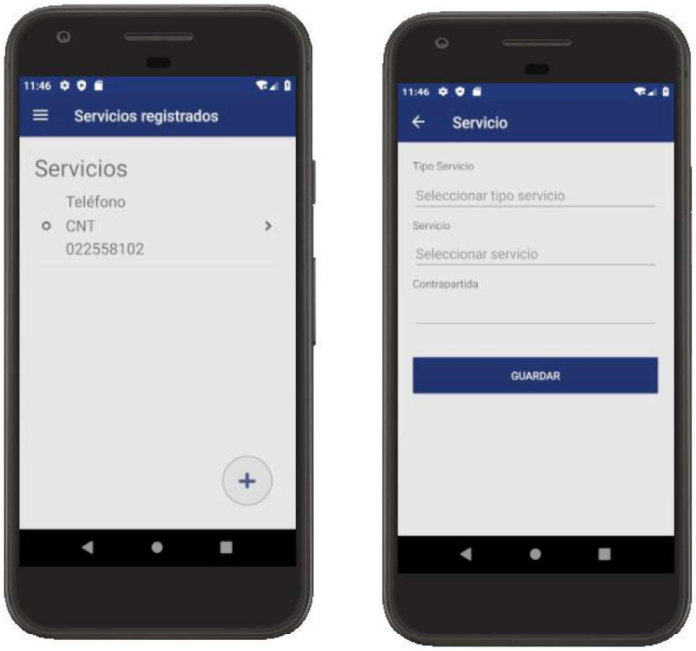


Figura 33. Pantalla registro servicios de pago app móvil.

Tabla 11

Historia de Usuario 8

Usuario: Usuario	
Nombre historia: Registro de cuenta PayPal	
Nivel de Prioridad: Alta	Iteración: 2
Puntaje estimado: 13	Riesgo: Baja
Descripción: Como usuario quiero poder conectarme con mis datos de PayPal para poder realizar el pago a través de esta plataforma.	
Criterio de aceptación: El usuario podrá contactarse con PayPal para poder realizar el pago a través de esta plataforma a través se su usuario y contraseña.	

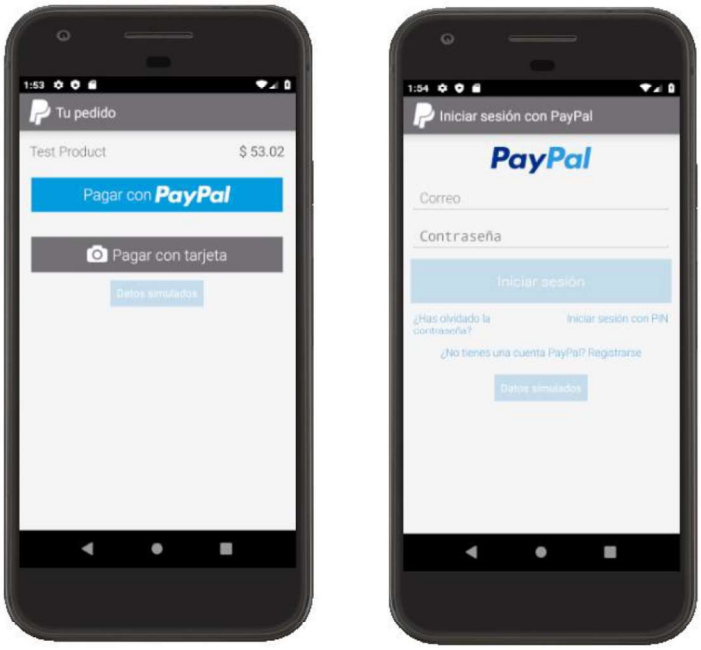


Figura 34. Pantalla registro de PayPal para pago app móvil.

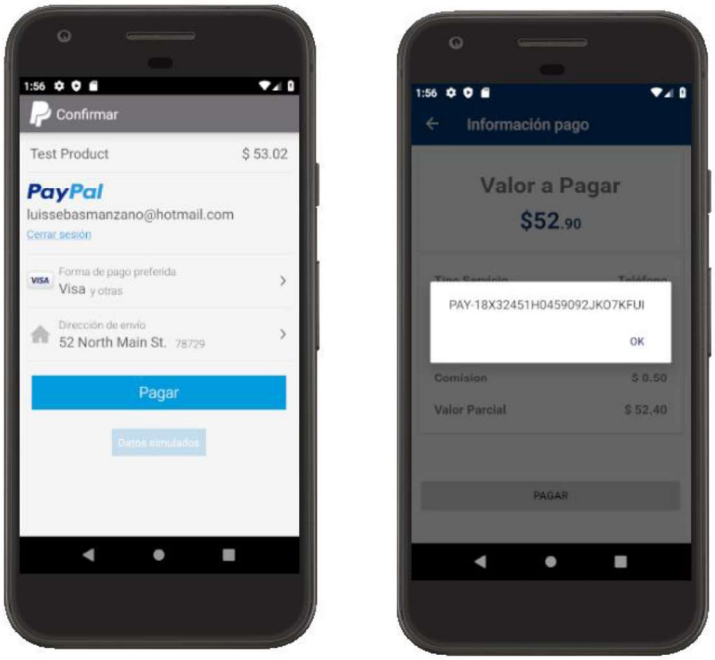


Figura 35. Pantalla pago con Paypal de pago app móvil.

4.1.3 Sprint 3

Tabla 12

Product Backlog Sprint 3

Id	Historia de Usuario	Prioridad	Estimación	Horas
9	Como usuario quiero que me llegue un código OTP para verificar el pago.	Alta	13	53
10	Como usuario quiero poder ingresar mis datos para realizar el pago de un servicio.	Media	13	53
11	Como usuario quiero que me llegue la información del pago realizado a mi mail para poder revisar y validar el mismo.	Media	8	24
12	Como usuario quiero poder ingresar mi huella para ingresar a la aplicación.	Media	5	21
13	Como usuario quiero poder ingresar mis datos para revisar el historial de pagos.	Baja	8	24

Para el sprint 3 se desarrollará la segunda parte que el usuario podrá realizar en la aplicación, perteneciendo las historias de usuario de 9 a la 12. Se ha estimado una duración de 23 días, alrededor de 184 horas, para dar por terminado.

Las historias de usuario que forman parte del sprint son:

- Recepción y verificación de código OTP.
- Simulación de pago de servicio.
- Envío de emails con información de acciones de la aplicación.
- Ingreso a la aplicación con huella dactilar

Tabla 13

Historia de Usuario 9

Usuario: Usuario	
Nombre historia: Recibir y verificar código OTP	
Nivel de Prioridad: Media	Iteración: 3
Puntaje estimado: 13	Riesgo: Baja
Descripción: Como usuario quiero que me llegue un código OTP para verificar el pago.	
Criterio de aceptación: El usuario podrá recibir un código OTP vía mail para verificar el pagos del servicio	

Figura 36. Pantalla código OTP app móvil.

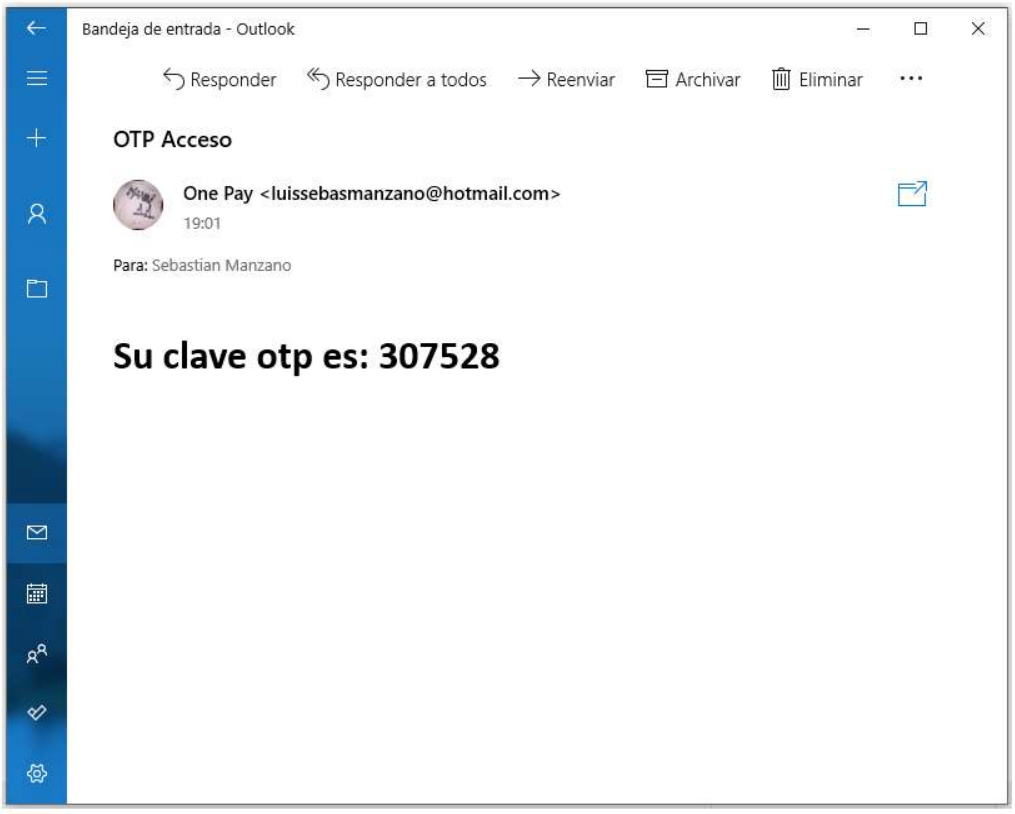


Figura 37. Pantalla código OTP mail.

Tabla 14

Historia de Usuario 10

Usuario: Usuario	
Nombre historia: Realiza el pago de servicio	
Nivel de Prioridad: Media	Iteración: 3
Puntaje estimado: 13	Riesgo: Baja
Descripción: Como usuario quiero poder ingresar mis datos para realizar el pago de un servicio.	
Criterio de aceptación: El usuario podrá registrar en la aplicación un pago de servicio.	

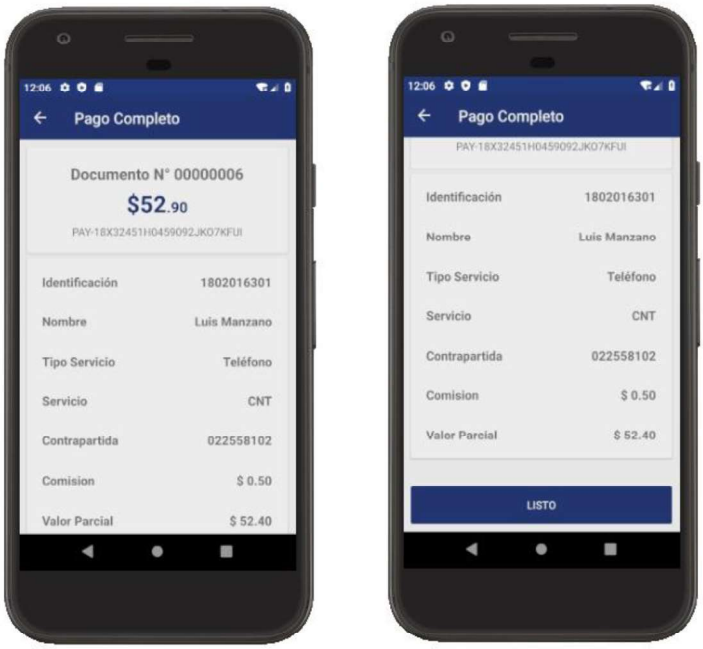


Figura 38. Pantalla pago realizado app móvil.

Tabla 15

Historia de Usuario 11

Usuario: Usuario	
Nombre historia: Enviar mail con información de pago de servicio.	
Nivel de Prioridad: Media	Iteración: 3
Puntaje estimado: 8	Riesgo: Baja
Descripción: Como usuario quiero que me llegue la información del pago realizado a mi mail para poder revisar y validar el mismo.	
Criterio de aceptación: El usuario podrá recibir un mail con la información del pago de servicio al momento en que se realizó	

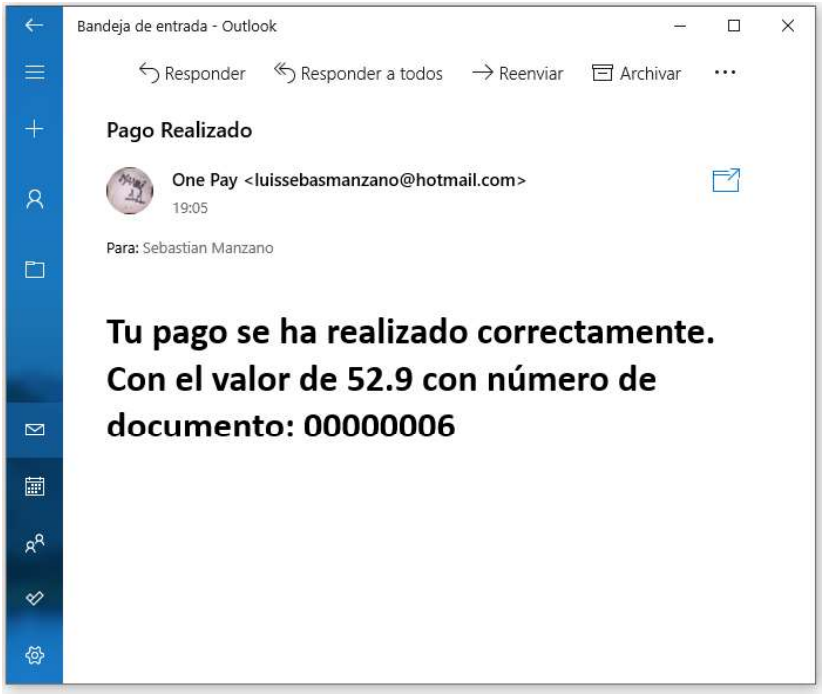


Figura 39. Pantalla pago realizado mail.

Tabla 16

Historia de Usuario 12

Usuario: Usuario	
Nombre historia: Ingresar a la aplicación por medio de huella dactilar.	
Nivel de Prioridad: Media	Iteración: 3
Puntaje estimado: 5	Riesgo: Baja
Descripción: Como usuario quiero poder ingresar mi huella para ingresar a la aplicación.	
Criterio de aceptación: El usuario podrá ingresar a la aplicación por medio de huella dactilar.	

Figura 40. Pantalla solicitud de huella dactilar para acceder a la app móvil.

Tabla 17

Historia de Usuario 13

Usuario: Usuario	
Nombre historia: Revisar el historial de pagos realizados.	
Nivel de Prioridad: Media	Iteración: 3
Puntaje estimado: 13	Riesgo: Baja
Descripción: Como usuario quiero poder ingresar mis datos para revisar el historial de pagos.	
Criterio de aceptación: El usuario debe agregar el tipo de servicio, el servicio un rango de fechas y seleccionar el tipo de reporte, de pagos o consultas.	

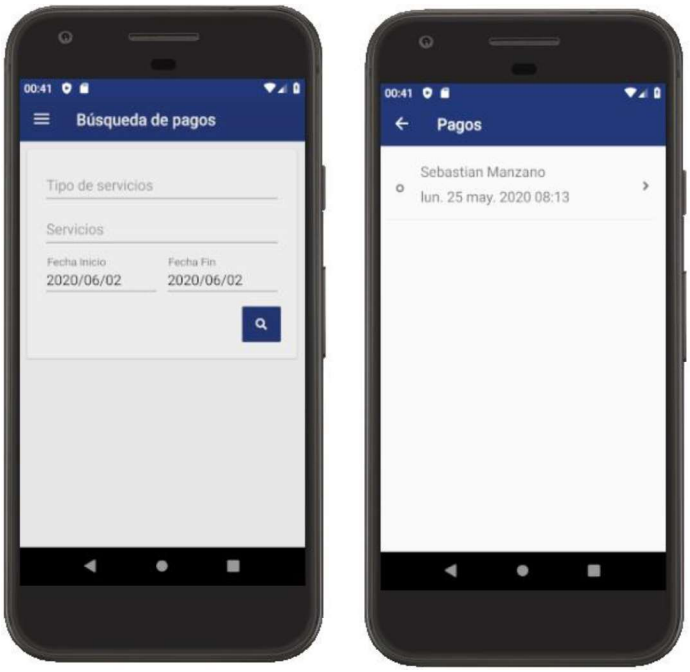


Figura 41. Pantallas historial de pagos registrados en la app móvil.

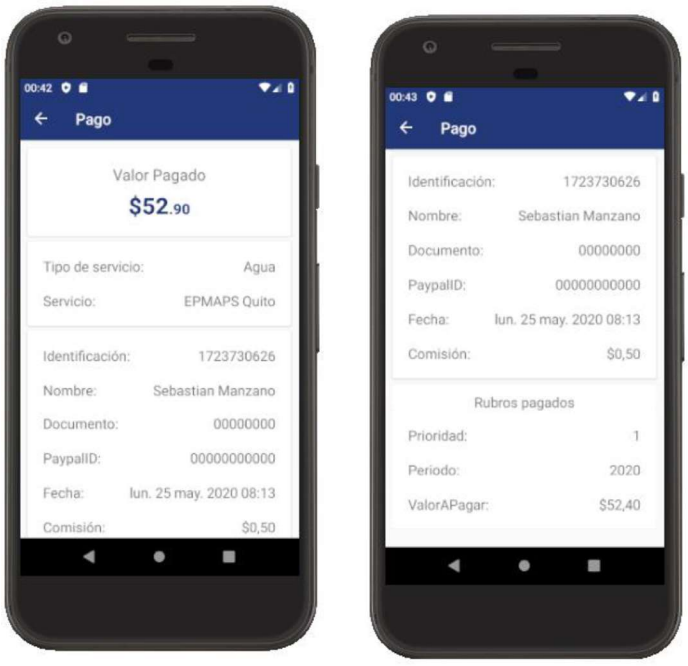


Figura 42. Pantallas historial de pagos registrados en la app.

5. Capítulo V. Pruebas De Sistema

5.1 Pruebas unitarias de funcionalidades

Como se mencionó anteriormente se utilizó el paquete NUnit. Se creó un proyecto con 3 pruebas esenciales las cuales fueron:

5.1.1 Prueba de login

A partir del diseño de la aplicación se desarrolló el código para las pruebas unitarias, donde se simula la funcionalidad de login que normalmente realizará un usuario.

```
[Test]
0 referencias | 0 cambios | 0 autores, 0 cambios
public void LoginSuccesfullTest()
{
    app.WaitForElement(c => c.Marked("Acceder"));
    app.Tap(c => c.Marked("Acceder"));
    app.WaitForElement(c => c.Marked("entUsuario"));
    app.WaitForElement(c => c.Marked("entUsuario"));
    app.EnterText("entUsuario", "luissebas");
    app.EnterText("entPassword", "Sebas123*-");
    app.DismissKeyboard();
    app.WaitForElement(c => c.Marked("Ingresar"));
    app.Tap(c => c.Marked("Ingresar"));
}
```

Figura 43. Código prueba login.

El resultado de la prueba se realizó con éxito donde los resultados se muestran directamente el IDE visual studio con sus detalles.

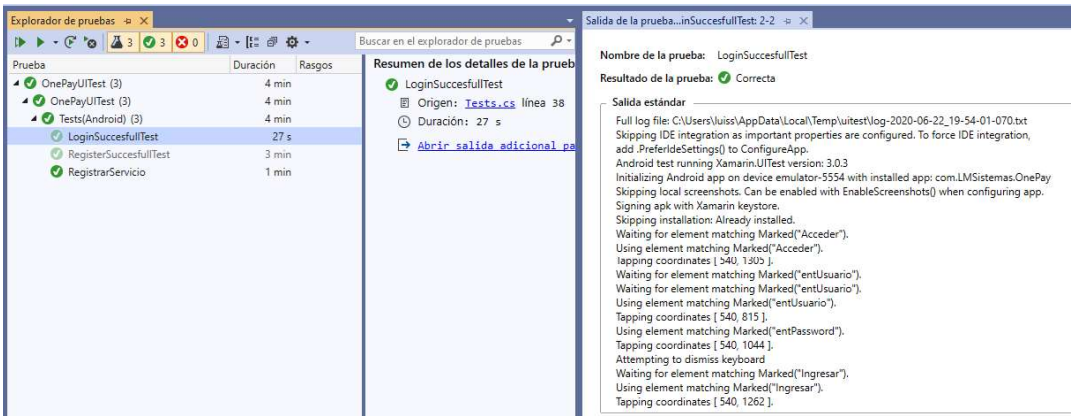


Figura 44. Resultado prueba login.

5.1.2 Prueba de registro de usuario

Así como con la prueba del login se realizó una prueba con el registro de usuario poniendo paso a paso, simulando la interacción del usuario para realizar la funcionalidad requerida

```
[Test]
0 referencias | 0 cambios | 0 autores, 0 cambios
public void RegisterSuccesfullTest()
{
    app.WaitForElement(c => c.Marked("Acceder"));
    app.Tap(c => c.Marked("Acceder"));
    app.WaitForElement(c => c.Marked("Registrar"));
    app.Tap(c => c.Marked("Registrar"));
    app.EnterText("entIdentificacion", "1723730014");
    app.EnterText("entNombre", "Juana Estrella");
    app.EnterText("entCorreo", "luis_sebastian_ldu@hotmail.com");
    app.DismissKeyboard();
    app.ScrollDown("Contraseña");
    app.ScrollDown("Contraseña");
    app.ScrollDown("Contraseña");
    app.ScrollDown("Contraseña");
    app.ScrollDown("Contraseña");
    app.EnterText("entUsuario", "Juana27");
    app.DismissKeyboard();
    app.EnterText("entContrasenia", "Juana123");
    app.DismissKeyboard();
    app.EnterText("entComprobacionContrasenia", "Juana123");
    app.ScrollDown("Repita su contraseña");
    app.ScrollDown("Repita su contraseña");
    app.ScrollDown("Repita su contraseña");
    app.ScrollDown("Repita su contraseña");
    app.ScrollDown("Repita su contraseña");
    app.DismissKeyboard();
    app.Tap(c => c.Marked("Registrar"));
}
```

Figura 45. Código prueba registro de usuario.

El resultado de la prueba se realizó con éxito donde los resultados se muestran directamente el IDE visual studio con sus detalles.

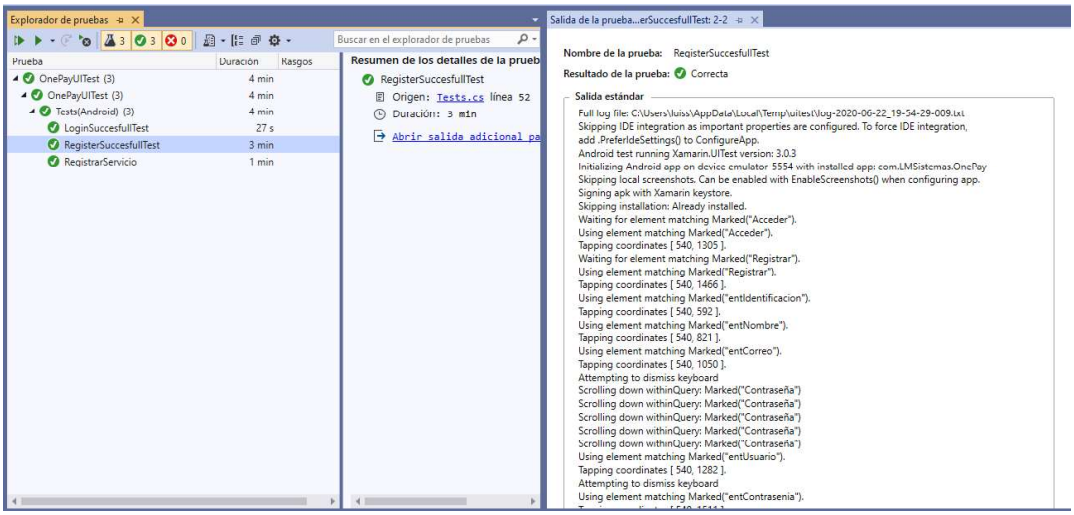


Figura 46. Resultado prueba registro de usuario.

5.1.3 Prueba de registro de servicio

Al igual que con las dos anteriores pruebas, se desarrolló el código para simular las acciones que el usuario tendrá que realizar para poder registrar su servicio para realizar pagos rápidos.

```
[Test]
0 referencias | 0 cambios | 0 autores, 0 cambios
public void RegistrarServicio()
{
    app.WaitForElement(c => c.Marked("Acceder"));
    app.Tap(c => c.Marked("Acceder"));
    app.WaitForElement(c => c.Marked("entUsuario"));
    app.WaitForElement(c => c.Marked("entUsuario"));
    app.EnterText("entUsuario", "luissebas");
    app.EnterText("entPassword", "Sebas123*-");
    app.DismissKeyboard();
    app.WaitForElement(c => c.Marked("Ingresar"));
    app.Tap(c => c.Marked("Ingresar"));
    app.WaitForElement(c => c.Marked("GrdBotones"));
    app.SwipeLeftToRight(0.99);
    app.Tap(c => c.Marked("Servicios registrados"));
    app.WaitForElement(c => c.Marked("btnAgregar"));
    app.Tap(c => c.Marked("btnAgregar"));
    app.WaitForElement(c => c.Marked("pkTipoServicio"));
    app.Tap(c => c.Marked("pkTipoServicio"));
    app.Tap(c => c.Marked("Teléfono"));
    app.Tap(c => c.Marked("pkServicio"));
    app.Tap(c => c.Marked("CNT"));
    app.WaitForElement(c => c.Marked("entCotrapartida"));
    app.ScrollDown("entCotrapartida");
    app.EnterText("entCotrapartida", "022558102");
    app.Tap(c => c.Marked("Guardar"));
}
```

Figura 47. Resultado prueba registro de usuario

El resultado de la prueba se realizó con éxito donde los resultados se muestran directamente el IDE visual studio con sus detalles.

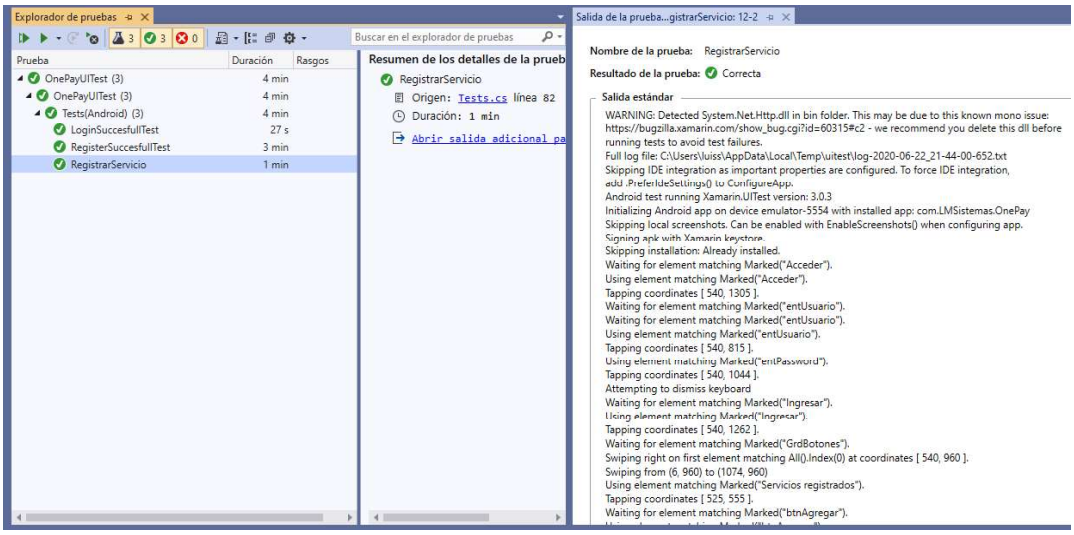


Figura 48. Código prueba registro de servicio.

5.2 Casos de prueba

5.2.1 Caso de prueba 1

Tabla 18

Caso de Prueba 1

Nombre del caso: Registro de servicio como administrador		
Descripción: Agregar todos los datos solicitados por la app para registrar el servicio del lado del administrador.		
Pasos	Resultado esperado	Resultado Obtenido
1. Ingreso a la aplicación como administrador.	Se muestra la pantalla principal del lado del administrador.	OK
2. Desde el menú se elige la opción	Se muestra el menú para administrador; una vez en la	OK

servicio y se da click en el botón “+”.	pantalla se da click en el botón para agregar el servicio.	
3. Se ingresan y verifican los datos solicitados	Ingreso de todos los datos solicitados.	OK
4. Se da click al botón guardar.	Se muestra un mensaje de confirmación de creación de servicio y regresa a la pantalla de servicios registrados.	OK

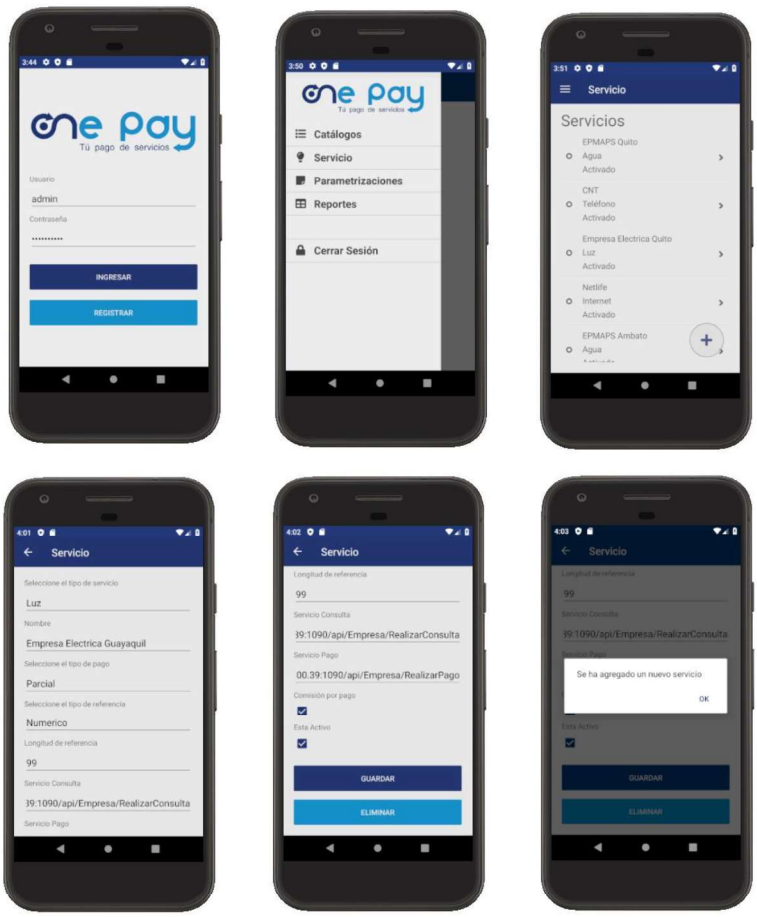


*Figura 49.*Caso de prueba registro de servicio

5.2.2 Caso de prueba 2

Tabla 19

Caso de Prueba 2

Nombre del caso: Pago de servicio		
Descripción: Agregar todos los datos solicitados por la app para realizar un pago.		
Pasos	Resultado esperado	Resultado Obtenido
1. Ingreso a la aplicación como usuario.	Se muestra la pantalla principal del lado del usuario.	OK
2. Desde las 4 imágenes a elegir se da click en “+”.	Se muestra la pantalla de formulario para pago de servicios.	OK
3. Se llena el formulario y se da click en siguiente.	Se muestra la pantalla de elección de pagos si el servicios es de tipo de pago “Parcial” caso contrario se muestra la pantalla de información de pago.	OK
4. Se eligen los pagos a realizar en caso de que el servicio tenga registrado el tipo de pago como “Parcial”. Se da click en siguiente	Se muestra la pantalla de información de pago.	OK

5. Se da click al botón pagar.	Se muestra pantalla de OTP (One Time Password)	OK
6. Se ingresa el código OTP que llega al mail. Se da click en siguiente.	Se muestra la pantalla de ingreso de PayPal.	OK
7. Se da click en pagar con PayPal. Seguido se ingresa usuario y contraseña de PayPal. Se da click en iniciar sesión.	Se muestra pantalla de selección de pago.	OK
8. Se elige el tipo de pago y la dirección y se click en pagar.	Se muestra pantalla de información de pago completado.	OK
9. Se revisa la información y se click en listo	Se muestra pantalla principal de usuario.	OK

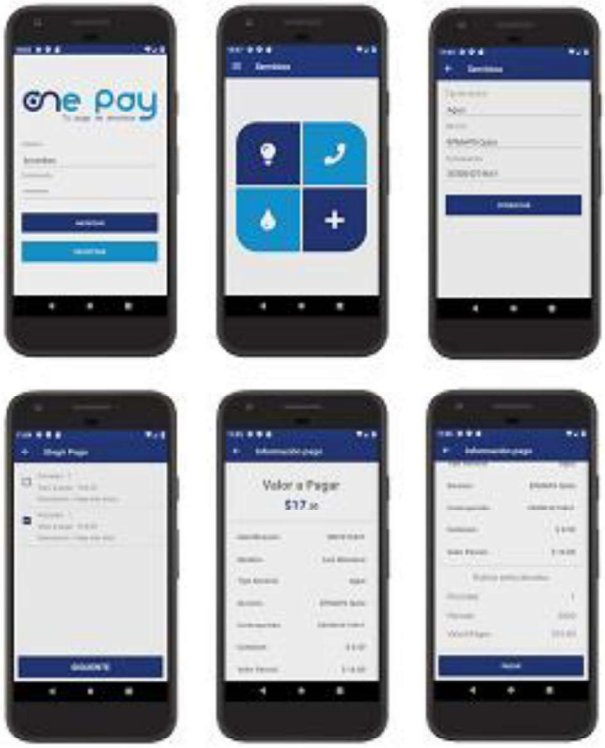


Figura 50. Caso de prueba pago de un servicio.

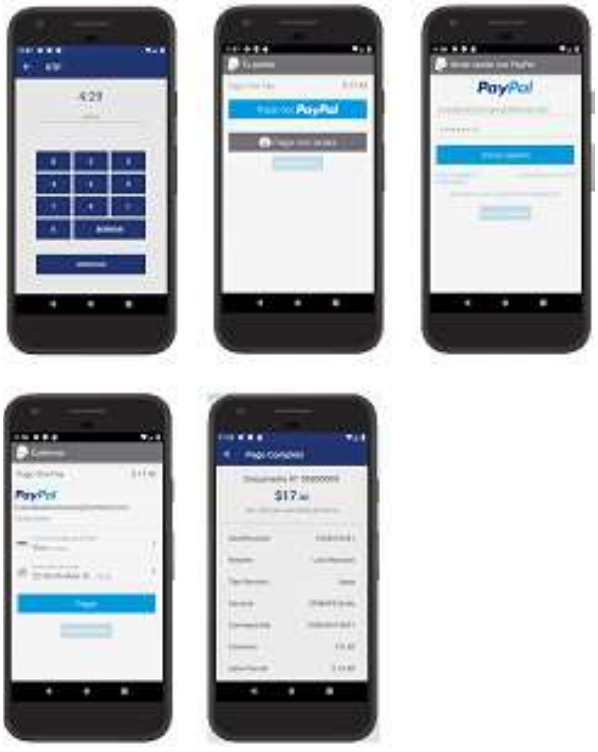


Figura 51. Caso de prueba pago de un servicio.

6. Conclusiones y Recomendaciones

6.1 Conclusiones

Los requerimientos obtenidos fueron analizados de tal manera que la aplicación pudo realizar su cometido, en donde se muestra un prototipo que evidenció la simulación de pagos de servicios a partir del consumo de web services que el administrador de la misma puede registrar según el servicio deseado.

Se diseñó una base de datos robusta y personalizada de acuerdo a la aplicación realizada en SQL 2017 ocupando claves foráneas, índices y seguridades que ofrece SQL. También protegiendo información de los servicios a utilizar.

Los resultados de las pruebas unitarias mostraron que las funcionalidades están correctamente desarrolladas y que no se demuestra errores en el código cumpliendo con la eficacia de la aplicación.

Se desplegó dos web service, donde ambas de desarrollaron .net core 3.0 con la diferencia que una funciona como capa de negocios de la aplicación y la segunda es la simulación de un web service de una empresa que permita pagos.

El diseño de la aplicación permite que sea utilizada en diferentes dispositivos, siendo esto posible gracias a la ayuda de Xamarin. Adaptada a los diferentes tamaños tanto de móviles como de tablets actuales disponibles en el mercado. Siempre buscando interfaces amigables con el usuario final y con consistencia de usabilidad de la aplicación.

La implementación con PayPal agrega un gran valor a la implementación, teniendo en cuenta que a partir del mismo se puede realizar los pagos, ademas de agregar más opciones por parte del mismo.

6.2 Recomendaciones

Para la obtención de los requerimientos se necesita retroalimentación de los usuarios que van a utilizar la aplicación, así como los clientes que van a prestar sus servicios para realizar las consultas o pagos.

Utilizar siempre la tecnología más actualizada del mercado, tomando en cuenta las actualizaciones que se tiene, siempre pensando en que hay que estar actualizado en los diferentes servicios que se puede ofrecer.

Realizar pruebas unitarias en base a casos de prueba planificados con un nivel de cobertura alto que permita mejorar la aplicación, además de las diferentes ya realizadas.

Usar el patrón MVVM para el desarrollo de aplicaciones móviles, ya que es muy utilizado por su desacoplamiento, además se encuentran las vistas más livianas debido a que casi no incluye lógica.

Se recomienda realizar una parametrización de datos que se puede enviar y recibir en los web services de terceros, donde puede llegar a ser muy ambigua la información que se obtiene de los mismos y puede causar dificultades al implementarlo.

Para aplicaciones móviles se recomienda usar frameworks cross-plataform. Ya que se puede desarrollar aplicaciones con las respectivas infraestructuras de cada sistema operativo móvil usando un único proyecto.

Referencias

Microsoft. (2017). Microsoft SQL Server 2017. Recuperado el 27 de marzo del 2020, de http://info.microsoft.com/rs/157-GQE-382/images/ES-ES-CNTNT-Whitepaper-SQLServer2017ITDMWhitePaper-ES.pdf

Oppel, A. Sheldon, R. (2009). Fundamentos de SQL. México: McGraw-Hill.

Microsoft. (2020). Breaking changes for migration from Version 2.2 to 3.1 Recuperado el 28 de marzo del 2020, de https://docs.microsoft.com/en-us/dotnet/core/compatibility/2.2-3.1

Microsoft. (2020). What is Xamarin.Forms? Recuperado el 28 de marzo del 2020, de https://docs.microsoft.com/en-us/xamarin/get-started/what-is-xamarin-forms

Microsoft. (2019). Le damos la bienvenida al IDE de Visual Studio. Recuperado el 28 de marzo del 2020, de https://docs.microsoft.com/es-es/visualstudio/get-started/visual-studio-ide?view=vs-2019

De León, A. (2019). Servidor IIS. Recuperado el 13 de junio del 2020, de https://blog.infranetworking.com/servidor-iis/

Cortés, Y. (2017). Qué es SCRUM y los roles en SCRUM. Recuperado el 28 de marzo del 2020, de https://platzi.com/blog/que-es-scrum-y-los-roles-en-scrum/

Imangulov, D. (2018). Building an ASP.NET Web API with ASP.NET Core. Recuperado el 28 de marzo del 2020, de https://www.toptal.com/asp-dot-net/asp-net-web-api-tutorial

Menzinsky, A. (2015). ¿Qué es el sprint 0?. Recuperado el 4 de abril del 2020, de https://scrum.menzinsky.com/2015/03/que-es-el-sprint-0-actualmente-hay.html

EALDE. (2017). La estimación ágil con la técnica Planning Poker para la Dirección de Proyectos. Recuperado el 4 de abril del 2020, de https://www.ealde.es/estimacion-planning-poker-direccion-de-proyectos/

Ravi, K. (2017). How does IIS SERVER work? Recuperado el 4 de abril del 2020, de https://www.quora.com/How-does-IIS-SERVER-work

Najarro, M. (2017). Planning Poker: ¿Cómo realizar la estimación inicial de un proyecto de forma rápida y fiable?. Recuperado el 21 de junio del 2020, de https://blog.interactius.com/planning-poker-c%C3%B3mo-realizar-la-estimaci%C3%B3n-inicial-de-un-proyecto-de-forma-r%C3%A1pida-y-fiable-3576e9f1a943