

UNIVERSIDAD MARIANO GÁLVEZ DE GUATEMALA
FACULTAD DE INGENIERÍA EN SISTEMAS DE INFORMACIÓN



JOSUÉ DAVID VARGAS GUTIÉREZ

GUATEMALA, octubre de 2024

UNIVERSIDAD MARIANO GÁLVEZ DE GUATEMALA
FACULTAD DE INGENIERÍA EN SISTEMAS DE INFORMACIÓN
*DESARROLLO DE APLICACIÓN WEB QUE FACILITE LA GESTIÓN
ADMINISTRATIVA DEL GIRO DEL NEGOCIO DE UNA BARBERIA
APLICADO EN STUARD BARBERSHOP
DEPARTAMENTO Y MUNICIPIO DE ZACAPA.*

TRABAJO DE GRADUACIÓN PRESENTADO POR:

JOSUÉ DAVID VARGAS GUTIÉRREZ

previo a optar al Grado Académico de

LICENCIATURA EN INGENIERÍA EN SISTEMAS DE
INFORMACIÓN Y CIENCIAS DE LA COMPUTACIÓN

y el Título Profesional de

INGENIERO EN SISTEMAS DE INFORMACIÓN
Y CIENCIAS DE LA COMPUTACIÓN

CONOCEREIS LA VERDAD
Y LA VERDAD OS HARÁ LIBRES

GUATEMALA

**AUTORIDADES DE LA FACULTAD Y ASESOR DEL
TRABAJO DE GRADUACIÓN**

DECANO DE LA FACULTAD:

ING. JORGE ALBERTO ARIAS TOBAR

SECRETARIO DE LA FACULTAD:

ING. M.A. HUGO ADALBERTO HERNÁNDEZ SANTIZO

ASESOR:

ING. WILLIAM ESTUARDO ESCOBAR ARGUETA

REGLAMENTO TRABAJO DE GRADUACIÓN

Artículo 8º: RESPONSABILIDAD

Solamente el autor es responsable de los conceptos expresados en el trabajo de tesis. Su aprobación en manera alguna implica responsabilidad de para la Universidad.

ÍNDICE

INTRODUCCIÓN	1
CAPÍTULO I - MARCO CONCEPTUAL	3
1.1 Antecedentes	3
1.2 Justificación	3
1.3. Problema	4
1.3.2 Planteamiento del problema.....	5
1.3.3 Definición del problema	5
1.4 Objetivos del proyecto	5
1.4.1 Objetivo general.....	5
1.4.2 Objetivos específicos	5
1.5 Delimitación del problema.....	6
1.5.1 Delimitación institucional.....	6
1.5.2 Delimitación geográfica.....	6
1.5.3 Delimitación temporal	6
1.5.4 Delimitación personal	6
1.5.5 Delimitación temática	6
1.5.6 Alcances y límites del problema.....	6
1.5.6.1 Alcances del problema.....	6
1.5.6.2 Limites del problema	7
1.6 Viabilidades	7
1.6.1 Viabilidad operativa.....	7
1.6.2 Viabilidad financiera.....	7
1.6.3 Viabilidad legal.....	8
1.7 Preguntas.....	8

1.7.1 Pregunta general.....	8
1.7.2 Preguntas auxiliares	9
CAPÍTULO II - MARCO TEORICO.....	11
2.1 Barbería.....	11
2.1.1 Servicios que presta	13
2.1.2 Citas programadas.....	14
2.2 Tecnologías	15
2.3 Lenguaje de programación.....	16
2.3.1 <i>PYTHON</i>	16
2.4 Base de datos.....	18
2.4.1 MYSQL.....	18
CAPÍTULO III - MARCO METODOLOGICO.....	21
3.1 Método general	21
3.2 Métodos auxiliares	21
3.2.1 Experimental	21
3.2.2 Análisis	21
3.2.3 Síntesis	21
3.3 Tipo de investigación	21
3.4 Prototipo.....	22
3.5 Metodología de desarrollo de programación	22
CAPÍTULO IV – ANALISIS DE LA APLICACIÓN O SOLUCION	23
4.1 Descripción del proyecto	23
4.2 Identificación de usuarios	23
4.3 Modelado del negocio.....	23
4.4 Metodología de desarrollo de software.....	24

4.5 Fases.....	24
4.5.1 Análisis y Requerimientos	24
4.5.2 Diseño	24
4.5.3 Desarrollo.....	25
4.5.4 Pruebas.....	25
4.5.5 Implementación y despliegue	25
4.5.6 Mantenimiento	25
4.6 Características	25
4.7 Artefactos	26
CAPÍTULO V – DISEÑO DE LA APLICACIÓN O SOLUCION	27
5.1 Actores	27
5.1.1 Administrador	27
5.1.2 Barbero.....	27
5.1.3 Cliente.....	27
5.2 Escenarios	27
5.2.1 Entorno del cliente	27
5.2.2 Entorno de instalaciones de la barbería	28
5.3 Diagrama de casos de uso	28
5.3.1 Diagramas de secuencias de <i>Stuard Barbershop</i>	28
5.3.1.1 Diagrama de secuencia para la agendación de citas	28
5.3.1.2 Diagrama de secuencia para ingresar servicios	29
5.3.1.3 Diagrama de secuencia para ingreso de cortes	30
5.3.2.1 Diagrama de actividades de <i>Stuard Barbershop</i>	31
5.3.2.2 Diagrama de actividades para agendación de citas.....	31
5.3.2.3 Diagrama de actividades para ingreso de servicios	32

5.3.2.4 Diagrama de actividades para ingreso de cortes	33
5.3.3 Diagrama de caso de uso de <i>Stuard Barbershop</i>	34
5.3.3.1 Diagrama de caso de uso para la agendación de citas	34
5.3.3.2 Diagrama de caso de uso para el ingreso de servicios	35
5.3.3.3 Diagrama de caso de uso para ingreso de cortes.....	35
5.3.4 Diagrama de estados de <i>Stuard Barbershop</i>	36
5.3.4.1 Diagrama de estados para la agendación de citas	36
5.3.4.2 Diagrama de estados para el ingreso de servicios.....	37
5.3.4.3 Diagrama de estados para el ingreso de cortes	38
5.3.5 Diagrama de colaboración para <i>Stuard Barbershop</i>	38
5.3.5.1 Diagrama de colaboración para agendacion de citas	39
5.3.5.2 Diagrama de colaboración para ingreso de servicios	39
5.3.5.3 Diagrama de colaboración para ingreso de cortes	40
5.3.6 Diagrama de objetos para <i>Stuard Barbershop</i>	40
5.3.6.1 Diagrama de objetos para agendacion de citas	41
5.3.6.2 Diagrama de objetos para ingreso de servicios.....	41
5.3.6.3 Diagrama de objetos para ingreso de cortes	42
5.3.7 Diagrama de entidad-relación de <i>Stuard Barbershop</i>	42
5.3.7.1 Diagrama de entidad-relación para agendación de citas.....	43
5.3.7.2 Diagrama de entidad-relación para ingreso de servicios	43
5.3.7.3 Diagrama de entidad-relación para ingreso de cortes	44
5.3.8 Diagrama de componentes de <i>Stuard Barbershop</i>	44
5.3.8.1 Diagrama de componentes para agendación de citas.....	45
5.3.8.2 Diagrama de componentes para ingreso de servicios	45
5.3.8.3 Diagrama de componentes para ingreso de cortes	46

5.3.9 Diagrama de infraestructura de <i>Stuard Barbershop</i>	46
5.3.9.1 Diagrama de infraestructura para agendación de citas	47
5.3.9.2 Diagrama de infraestructura para ingreso de servicios	48
5.3.9.3 Diagrama de infraestructura para agendación de cortes	48
5.3.10 Diagrama de clases de <i>Stuard Barbershop</i>	49
5.3.10.1 Diagrama de clases para agendación de citas	49
5.4 Maquetado de la aplicación web <i>Stuard Barbershop</i>	50
5.4.1 Ingreso al sistema.....	50
5.4.2 Home.....	51
5.4.3 Agendar cita	52
5.4.4 Listado de citas	53
5.4.5 Agregar servicios y listado.....	54
5.4.6 Agregar cortes y catalogo	55
5.4.7 Agregar producto y catalogo.....	56
5.4.8 Reporte de servicios.....	57
CAPÍTULO VI – DESARROLLO E IMPLEMENTACION DE LA APLICACIÓN O SOLUCION	59
6.1 Ingreso a la aplicación o login	59
6.2 Vista principal o home	60
6.3 Menú de la aplicación web	61
6.4 Listado de citas	62
6.5 Agendar citas	63
6.6 Listado de servicios.....	64
6.7 Agregar servicio.....	65
6.8 Catalogo de cortes.....	66

6.9 Agregar cortes	67
6.10 Conexión a base de datos	68
6.11 Ingreso a la aplicación	68
6.12 Menú principal de la aplicación	69
6.12.1 Código javascript	71
6.13 Modelos principales	74
6.13.1 Modelos de citas	74
6.13.2 Modelos de Estilos de cortes	74
6.13.3 Modelos de productos	75
6.13.4 Modelos de servicios	75
6.14Urls principales	76
6.14.1Urls de citas	76
6.14.2Urls de estilos de cortes	76
6.14.3Urls de productos	77
6.14.4Urls de servicios	77
6.15 Vistas principales	78
6.15.1 Vista de citas	78
6.15.2 Vista de estilos de cortes	79
6.15.3 Vista de productos	80
6.15.4 Vista de servicios	81
CONCLUSION	83
RECOMENDACIONES	84
GLOSARIO	85
REFERENCIAS	87

x_i

ÍNDICE DE TABLAS

Tabla 1 Recursos Materiales	7
Tabla 2 Recursos Humanos	7
Tabla 3 Recursos de Software.....	8
Tabla 4 Costo Total del Proyecto.....	8

ÍNDICE DE ILUSTRACIONES

Figura 1 Diagrama de secuencia para la agendación de citas.....	28
Figura 2 Diagrama de secuencia para ingresar servicios.....	29
Figura 3 Diagrama de secuencia para ingreso de cortes.....	30
Figura 4 Diagrama de actividades para agendación de citas.....	31
Figura 5 Diagrama de actividades para ingreso de servicios.....	32
Figura 6 Diagrama de actividades para ingreso de cortes.....	33
Figura 7 Diagrama de caso de uso para la agendación de citas.....	34
Figura 8 Diagrama de caso de uso para el ingreso de servicios.....	35
Figura 9 Diagrama de caso de uso para ingreso de cortes.....	35
Figura 10 Diagrama de estados para la agendación de citas.....	36
Figura 11 Diagrama de estados para el ingreso de servicios.....	37
Figura 12 Diagrama de estados para el ingreso de cortes.....	38
Figura 13 Diagrama de colaboración para agendacion de citas.....	39
Figura 14 Diagrama de colaboración para ingreso de servicios.....	39
Figura 15 Diagrama de colaboración para ingreso de cortes.....	40
Figura 16 Diagrama de objetos para agendacion de citas.....	41
Figura 17 Diagrama de objetos para ingreso de servicios.....	41
Figura 18 Diagrama de objetos para ingreso de cortes.....	42
Figura 19 Diagrama de entidad-relación para agendación de citas.....	43
Figura 20 Diagrama de entidad-relación para ingreso de servicios.....	43
Figura 21 Diagrama de entidad-relación para ingreso de cortes.....	44
Figura 22 Diagrama de componentes para agendación de citas.....	45
Figura 23 Diagrama de componentes para ingreso de servicios.....	45
Figura 24 Diagrama de componentes para ingreso de cortes.....	46
Figura 25 Diagrama de infraestructura para agendación de citas.....	47
Figura 26 Diagrama de infraestructura para ingreso de servicios.....	48
Figura 27 Diagrama de infraestructura para agendación de cortes.....	48
Figura 28 Diagrama de clases para agendación de citas.....	49
Figura 29 Ingreso al sistema	50

Figura 30 Home	51
Figura 31 Agendar cita.....	52
Figura 32 Listado de citas	53
Figura 33 Agregar servicio y listado.....	54
Figura 34 Agregar cortes y catalogo.....	55
Figura 35 Agregar producto y catalogo	56
Figura 36 Reporte de servicios	57
Figura 37 Vista principal o home.....	60
Figura 38 Menu de la aplicacion web	61
Figura 39 Listado de citas	62
Figura 40 Agendar citas	63
Figura 41 Listado de servicios	64
Figura 42 Agregar servicio	65
Figura 43 Catalogo de cortes	66
Figura 44 Agregar corte	67

INTRODUCCIÓN

En la actualidad, la tecnología ha permeado prácticamente todos los aspectos de nuestra vida cotidiana. Desde la comunicación hasta las compras y la gestión de nuestras finanzas, los dispositivos móviles se han convertido en herramientas indispensables que nos acompañan a todas partes. En este contexto, el sector de la belleza y el cuidado personal no es una excepción. Las barberías, que tradicionalmente han dependido de métodos de reserva más convencionales, ahora están reconociendo el valor de adoptar soluciones tecnológicas para mejorar sus operaciones y la experiencia del cliente.

Una de las principales áreas donde la tecnología puede marcar la diferencia es en la gestión de citas. Las barberías suelen experimentar desafíos al gestionar su agenda, con problemas como citas perdidas, retrasos y ausencias de clientes. Además, los clientes también enfrentan dificultades al intentar programar citas, ya sea debido a horarios ocupados o a la falta de opciones de reserva convenientes. Aquí es donde entra en juego una aplicación web de citas diseñada específicamente para barberías.

Esta aplicación web no solo ofrece a los clientes la comodidad de reservar citas desde la palma de su mano, sino que también proporciona a las barberías herramientas poderosas para gestionar su agenda de manera más eficiente. Desde la capacidad de ver la disponibilidad en tiempo real hasta recibir recordatorios automáticos de citas, esta aplicación tiene el potencial de transformar por completo la forma en que se gestionan las citas en las barberías. Además, al integrar características como perfiles de clientes, historiales de citas y opciones de pago móvil, esta aplicación puede ofrecer una experiencia personalizada y sin problemas para los clientes, mejorando así su satisfacción y fidelización.

Sin embargo, el desarrollo e implementación de una aplicación móvil de citas para barberías no está exento de desafíos. Desde la seguridad de los datos del cliente hasta la integración con los sistemas existentes de la barbería, hay una serie de consideraciones técnicas y prácticas que deben abordarse para garantizar el éxito de la aplicación. Además, la adopción por parte de los clientes y el personal de la barbería también son factores críticos que deben ser considerados. Será necesario implementar medidas de seguridad robustas para proteger la información personal y financiera de los usuarios, así como asegurar una experiencia de usuario intuitiva y fluida que facilite la transición desde los métodos de reserva tradicionales.

En este proyecto, exploraremos en detalle el proceso de desarrollo e implementación de una aplicación móvil de citas para barberías, desde la investigación inicial hasta la fase de lanzamiento y más allá. Analizaremos los beneficios potenciales de esta aplicación, así como los desafíos y consideraciones que deben tenerse en cuenta en cada etapa del proceso. Además, discutiremos cómo esta aplicación puede aprovechar las últimas tendencias tecnológicas y las mejores prácticas de la industria para ofrecer una solución integral y centrada en el cliente.

En última instancia, este proyecto busca demostrar el valor de la tecnología móvil en la industria de la belleza y el cuidado personal, y cómo una aplicación móvil de citas puede impulsar la eficiencia operativa, mejorar la experiencia del cliente y fomentar el crecimiento y la innovación en las barberías modernas. A medida que la tecnología continúa avanzando, la capacidad de adaptarse y adoptar nuevas herramientas se convierte en un diferenciador clave para las barberías que desean mantenerse competitivas y relevantes en un mercado en constante evolución.

CAPÍTULO I - MARCO CONCEPTUAL

Se establece la información recopilada, el porqué de nuestro proyecto y algunos de los problemas dados en los requerimientos.

1.1 Antecedentes

Actualmente *Stuard Barbershop* ofrece servicios de cortes de cabello, cortes de barba, alineado de cejas, mascarillas, líneas, entre otras, esta trabaja a través de citas, las cuales son establecidas por el administrador, las citas se hacen por medio de mensajes o llamadas, el administrador recibe el mensaje o llamada y establece la cita en el horario en el cual el cliente lo solicitó.

StyleSeat: Es una aplicación confiable y popular que conecta a clientes con estilistas y barberos en su área local. Desde su lanzamiento en 2011, ha sido utilizada por millones de personas en todo el mundo para encontrar servicios de belleza de calidad y programar citas de manera conveniente.

Booksy: Se fundó con la idea fundamental de que los dueños de pequeños negocios necesitan tiempo y espacio para dar vida a su arte. Nuestros fundadores lo saben de primera mano y crearon Booksy para devolver a los emprendedores su recurso más valioso: el tiempo. Hoy en día, nuestro equipo lleva ese propósito a través de nuestro trabajo, ya que ayudamos a los profesionales de la belleza de todo el mundo. (Booksy, 2,024)

AgendaPro: Es una herramienta indispensable en la industria de la belleza y la barbería. Al simplificar la gestión operativa, mejorar la experiencia del cliente y apoyar el crecimiento del negocio, estas soluciones tecnológicas permiten a los profesionales de la barbería alcanzar un mayor nivel de éxito y satisfacción en un mercado cada vez más competitivo. (Agendapro, s.f.)

1.2 Justificación

En la actualidad, la digitalización y la tecnología están transformando las empresas, negocios y organizaciones, incluyendo la de los servicios personales como las barberías. La implementación de una aplicación web para agendar citas en una barbería ofrece numerosas ventajas que justifican plenamente su desarrollo y adopción.

La gestión manual de citas a menudo resulta ineficiente, llevando a errores de programación y tiempo de espera innecesario tanto para los clientes como para el personal de la barbería. Una aplicación web automatiza el proceso de agendamiento, permitiendo una mejor organización del tiempo y recursos, lo cual se traduce en una operación más fluida y eficiente.

En un mercado competitivo, ofrecer una experiencia superior al cliente es crucial. La aplicación permitirá a los clientes reservar citas de manera rápida y sencilla, desde cualquier lugar y en cualquier momento. Esto no solo mejora la conveniencia para los clientes, sino que también reduce la frustración asociada con largas esperas o dificultades para agendar citas por teléfono.

Las aplicaciones móviles pueden enviar recordatorios automáticos a los clientes sobre sus próximas citas. Esto asegura que los horarios de los barberos se utilicen de manera óptima y que no haya pérdida de ingresos debido a citas perdidas. Mediante la integración de programas de fidelización, promociones exclusivas y descuentos a través de la aplicación, se puede fomentar la lealtad de los clientes existentes. Una experiencia de usuario personalizada y beneficios adicionales incrementan la retención y satisfacción del cliente.

El uso de aplicaciones web para servicios diversos es una tendencia. Adaptarse a estas tendencias no solo posiciona a la barbería como un negocio moderno, sino que también atrae a una base de clientes más joven y familiarizada con la tecnología, ampliando así el mercado objetivo.

La aplicación permite a los administradores y propietarios de la barbería gestionar horarios, personal y servicios de manera más efectiva. La centralización de la información y la posibilidad de realizar ajustes en tiempo real facilitan un giro de negocio y gestión administrativa más eficiente y proactiva, la implementación de una aplicación móvil para la gestión de citas en una barbería no solo optimiza las operaciones internas, sino que también mejora la experiencia del cliente, reduce pérdidas, y proporciona una plataforma para la recopilación de datos valiosos. Estas ventajas justifican de manera clara y contundente la inversión en el desarrollo de esta aplicación, asegurando beneficios tanto para los clientes como para la barbería misma.

1.3. Problema

Se define como un acontecimiento que ya ha sucedido, cuando un problema es identificado ya no puede ser mitigado, sino que tiene que ser resuelto, un problema es un evento actual que requiere una acción inmediata para poder minimizar el efecto negativo en los objetivos del proyecto. (Consulting, 2,022)

1.3.1 Determinación del problema

La Barberia enfrenta problemas en la gestión de citas que afectan la eficiencia y experiencia del cliente.

1.3.2 Planteamiento del problema

Este mismo enfrenta desafíos significativos en la gestión de citas, lo que impacta tanto la eficiencia interna del negocio como la percepción y la experiencia del cliente, esta pone en riesgo la retención de clientes y la reputación del negocio en el mercado, por lo tanto, es imperativo abordar estos desafíos para mejorar la eficiencia interna y la satisfacción del cliente.

1.3.3 Definición del problema

El problema en la gestión de citas de las barberías se centra en la ineficiencia operativa y la comunicación deficiente, afectando la productividad interna y la satisfacción del cliente, Las áreas específicas de preocupación incluyen la programación ineficiente que conduce a periodos de inactividad y superposición de citas.

1.4 Objetivos del proyecto

Se trata de las metas que se pretenden alcanzar por medio de este. Dicho en otras palabras, es el lugar al que se pretende llegar, para el cual es necesario estructurar alguna planificación o planteamiento que permite de manera práctica acercar el proyecto a sus fines. (Euroinnova, 2,024)

1.4.1 Objetivo general

Facilitar a la barbería *Stuard Barbershop* agendar citas, asignación de barberos y servicios, enviar notificaciones de recordatorio para evitar ausencia de citas y aglomeraciones en las instalaciones y reducir el tiempo de espera de los clientes, por medio de una aplicación web de citas.

1.4.2 Objetivos específicos

- Facilitar la realización de citas para mejorar la calendarización de horarios por medio del módulo de agendar cita.
- Permitir la selección del barbero y tipo de servicio a requerir para la asignación de barbero y estimar el tiempo asignado para el servicio a través de un módulo de agendar cita.

- Optimizar la gestión de tiempo en la barbería por medio de notificaciones de recordatorio de la cita a los clientes y así también minimizar las ausencias.

1.5 Delimitación del problema

La delimitación del problema para un sistema de citas en una barbería se centra en la ineficiencia en la programación, la comunicación deficiente con los clientes, impacto en la experiencia del cliente, efectos en la eficiencia operativa, limitaciones tecnológicas actuales y el contexto específico del salón o barbería. Este enfoque busca abordar desafíos específicos relacionados con la gestión de citas para mejorar tanto la eficacia operativa interna como la experiencia del cliente.

1.5.1 Delimitación institucional

El proyecto se realizará en las instalaciones de la barbería *Stuard Barbershop*.

1.5.2 Delimitación geográfica

El proyecto será implementado en *Stuard Barbershop* ubicada en el departamento de Zacapa.

1.5.3 Delimitación temporal

La investigación del proyecto hasta su implementación tendrá una duración de 6 meses.

1.5.4 Delimitación personal

Se obtuvo la información con el gerente de la barbería *Stuard Barbershop* ubicada en el departamento de Zacapa para la realización de la aplicación móvil de Citas y también de los integrantes de esta misma para alimentar el sistema por medio de la aplicación web.

1.5.5 Delimitación temática

PYTHON, DJANGO, MYSQL.

1.5.6 Alcances y límites del problema

Los alcances y los límites del proyecto que se llevará a cabo en la barbería *Stuard Barbershop*.

1.5.6.1 Alcances del problema

La implementación abarcará a *Stuard Barbershop* y posiblemente algunas barberías más ubicadas en el departamento de Zacapa, el cual es el lugar donde se llevará a cabo la implementación de la aplicación de citas.

1.5.6.2 Limites del problema

La aplicación web está sujeta únicamente a las funciones de citas y venta de productos de la misma barbería que aparezcan en esta misma, este requiere de acceso a internet o datos móviles para poder utilizarla de manera funcional, se tiene como objetivo que funcione en dispositivos Android y IOS.

1.6 Viabilidades

Nos referimos a qué tan probable es llevar algo a cabo, materializarlo en la realidad.
(Concepto, 2.024)

1.6.1 Viabilidad operativa

Se cuenta con el personal capacitado para poder llevar a cabo el proyecto, tenemos los procesos para poder culminar el proyecto y para darle el soporte necesario.

1.6.2 Viabilidad financiera

La barbería actualmente cuenta con el móvil necesario para la implementación de la aplicación.

Tabla 1 Recursos Materiales

Esta tabla refleja los costos en el equipo necesario para la utilización de la aplicación móvil.

Cantidad	Recursos	Costo
2	Tablet de 8 pulgadas	Q1,500.00
1	Smart TV de 32 pulgadas	Q2,000.00
Subtotal		Q3,500.00

Fuentes: Elaboración propia.

Tabla 2 Recursos Humanos

Esta tabla muestra los gastos que se tendrá para el sueldo de los desarrolladores.

Cantidad	Recursos	Costo
30 horas	Análisis de necesidades Q55.00 cada hora	Q 1,650.00
20 horas	Diseño del sistema Q55.00 cada hora	Q 1,100.00
20 horas	Diseño y desarrollo de base de datos Q55.00 cada hora	Q 1,100.00
100 horas	Desarrollador Q60.00 cada hora	Q 6,000.00
20 horas	Revisión y prueba del sistema Q55.00 cada hora	Q 1,100.00

Subtotal	Q10,950.00
<i>2 horas por 5 días a la semana, Datos obtenidos de tusalario.org/Guatemala.</i>	

Tabla 3 Recursos de Software

Esta tabla refleja los costos de software necesarios para la implementación de la aplicación móvil.

Cantidad	Recursos	Costo
1	Dominio Web	Q 160.00
1	Servidor Web Linux	Q 870.00
Subtotal		Q1,030.00

Datos obtenidos para costos del servidor web, \$9 dólares al mes, Q73.00 quetzales, lo que equivale al año Q870.00 quetzales. Datos obtenidos para el costo de dominio web, \$20 al año. Los precios están sujetos a cambios.

Tabla 4 Costo Total del Proyecto.

Esta tabla refleja el gasto total para la implementación del proyecto.

Recursos	Costo
Recursos Materiales	Q 3,500.00
Recursos Humanos	Q10,950.00
Recursos de Software	Q 1,030.00
Total	Q15,480.00

Tabla 3: Costo Total del Proyecto, Fuentes: Elaboración propia.

1.6.3 Viabilidad legal

Se cumple con la autorización requerida para obtener información e implementarla en la aplicación móvil de citas para la barbería *Stuard Barbershop*.

1.7 Preguntas

Las preguntas que guiarán el desarrollo de este proyecto y que se pretenden responder al llegar a su implementación en la barbería *Stuard Barbershop*.

1.7.1 Pregunta general

¿Cómo impacta la implementación de una aplicación móvil de citas en la gestión y la experiencia del cliente de la barbería *Stuard Barbershop*?

1.7.2 Preguntas auxiliares

- ¿Cuál es el nivel de satisfacción de los clientes con el proceso de agendación de citas actual en *Stuard Barbershop*?
- ¿Cuál es la situación del cliente cuando olvida su agendación de cita y no llega puntual?
- ¿Cuánto tiempo requiere agendar la cita a través de un mensaje o llamada?

CAPÍTULO II - MARCO TEÓRICO

Este proporciona el fundamento conceptual y la base de conocimientos sobre los temas relevantes relacionados con el desarrollo y la implementación de la aplicación.

2.1 Barbería

Una barbería es un salón de belleza enteramente dedicado a servicios para hombres. Corte de pelo, tratamientos para el cuidado de la barba, pero también tratamientos para el rostro, la piel y el cabello, un SPA dedicado exclusivamente a los hombres. (Academy, s.f.)

Es una moda que se le puede llamar negocio, este indica que la persona que atiende es un barbero y para serlo se necesita tener distintos tipos de habilidades, años de experiencia y mucho conocimiento en otras áreas de este arte, es una disciplina, es ciencia y también es arte, por lo cual no es una moda pasajera.

Para tener una barbería y ser el barbero se necesita mucho conocimiento y habilidades que se desarrollan con muchos años de experiencia, desde hace 3 mil años se transmite el conocimiento y experiencia bajo un binomio llamado barbero-aprendiz, un barbero no es quien te hace 2 o 3 cortes o rayas con una delineadora, tradicionalmente han existido estatutos, parámetros y lineamientos para calificar a alguien como barbero y a un negocio como barbería.

La barbería promueve el acicalado masculino y la interacción entre todo tipo de personas sin importar el nivel social o cultural, no porque un local este pintado de blanco, rojo y azul es una barbería o porque veamos una persona con un porta-navajas es un barbero. (Aura, s.f.)

Aunque una gran mayoría lo desconozca, los primeros barberos compartían este oficio con el de cirujanos, ya que hacían todo tipo de curas, extracción de muelas y cirugías.

Antes de entrar de lleno con el significado del término barbería, conviene partir del concepto de barbero, que procede del término barba, que no es más que el pelo que cubre el comienzo del cuello, el mentón y la zona de las mejillas en el hombre. Por ende, se llama barbero a la persona que se dedica al cuidado de la barba.

Los barberos, por lo tanto, arreglan o afeitan la barba de sus clientes. Por lo general estas personas también se encargan del cabello, es decir que, por extensión, también son peluqueros. Puede decirse de este modo, que los barberos se ocupan de todo lo concerniente con los pelos del rostro y de la cabeza, principalmente en el hombre.

Si nos centramos en el significado preciso de cada término, actualmente la mayoría de los peluqueros no son barberos, ya que no se dedican al cuidado de la barba, sino que solo peinan y

cortan el cabello. En la antigüedad, en cambio, el oficio de barbero era muy habitual. Ahora bien, aproximándonos al término de barbería propiamente dicho, tenemos que era el lugar de trabajo del barbero. Estos espacios solían funcionar como centros sociales y hasta políticos, ya que era común que el barbero y sus clientes discutieran sobre diversos temas.

Como dato curioso podemos mencionar que originalmente, en la Europa Medieval, los barberos también se desempeñaban como cirujanos y además de cortar barbas, también practicaban cirugías, amputaciones, enemas, arreglaban roturas, extraían cálculos del riñón, trataban heridas, limpiaban oídos y formulaban ungüentos.

En el siglo XV, bajo el reinado de Enrique VIII, los oficios de barberos y cirujanos fueron separados, independizados, aunque los barberos aún tenían a su cargo la extracción de muelas. De hecho, el poste blanco y rojo que tradicionalmente ha distinguido a las barberías tiene mucho que ver con la dualidad del oficio de barbero con el de cirujano y la sangre que emanaba de muchas de sus curas. En síntesis, el rojo representa la sangre y el color blanco, los vendajes que se colocaban después de alguna cirugía. (Infoguía, 2,019)

Aunque en un principio puede parecer que todas las barberías ofrecen los mismos servicios, nada más lejos de la realidad. Entre cada una de ellas existe una diferencia de servicios, precios y estilos, que marca la diferencia entre lucir un cabello sano, con un corte de tendencia y una coloración natural, entre estas barberías están:

Barberías de barrio:

Este tipo de peluquerías se caracterizan principalmente por ofrecer precios bastante económicos por sus servicios. Puedes encontrar cortes de pelo para hombre. Están ubicadas en los diferentes barrios de la ciudad y generalmente no le dan tanta importancia al diseño de interior del local. En cuanto a los servicios, es más difícil poder elegir cortes de tendencia porque no suelen prestar tanta atención a los eventos del sector.

Franquicias de Barberías:

Se caracterizan por tener un estilo común en todos los salones. Mismo diseño de interior del local, mismos conocimientos de las diferentes técnicas.

Barberías emprendedoras o de nueva apertura:

Son barberías que han surgido recientemente y que tienen a su favor una alta cualificación por parte de su personal, que generalmente suele ser reducido. En contraposición, poseen escasa experiencia en el sector.

Barberías con alta experiencia:

Por último, encontramos las barberías que cuentan con una dilatada experiencia en el sector y que tienen como norma la formación continua en las últimas tendencias para poder ofrecer a sus clientes una experiencia única y diferenciadora, tanto en el servicio que ofrecen como en su atención personalizada. (Hade, 2,023)

2.1.1 Servicios que presta

Tratamientos para la barba:

Los servicios para barba deben existir de cajón en cualquier barbería, desde lo más básico como un perfilado, hasta rituales de barba completos, le da un giro al clásico afeitado de barba personalizando el servicio de acuerdo con el tipo de piel del cliente y qué tan abundante sea su barba, en función de los factores anteriores, se sabe cuándo ofrecer un afeitado con *shave cream*, con gel de afeitar o con espuma y el tipo de navaja que se necesitará.

Y si se quiere llevar la experiencia del ritual de barba a otro nivel, se puede hidratar la barba del cliente con algún aceite o aplicarle una mascarilla para limpiar las impurezas de su rostro, también se puede aplicar tintes para barba, si es que los clientes quieren probar algo distinto o quieren mantener su tono.

Tratamientos Capilares:

Todo cliente acude a una barbería para cuidar su cabello, aprovechar el momento en que los barberos están haciendo un corte a un cliente, para detectar qué necesita su cabello, ya sea que le recomiendan y le apliquen algún tipo de mascarilla, se complementa el servicio lavando su cabello con un shampoo anticaspa o anticaída, de igual forma se podría hacer una exfoliación a su cuero cabelludo y realizar una limpieza profunda para mantener su cabello saludable, al igual que con la barba, pueden ofrecer tintes para aquellos clientes que les guste mantener vivo el color de su cabello o quieran un look distinto.

Tratamientos faciales

El secreto de una barba perfecta, es tener una piel impecable, lo mismo pasa con el cabello, cuando la piel presenta un exceso de grasa y el cabello entra en contacto con el rostro, es probable que favorezca la caída del cabello, existen muchos tipos de mascarillas, como las de carbón, las doradas, entre muchas otras, que harán que la expresión de los clientes cambie por completo cuando salen de la barbería, un masaje relajante, un tratamiento para reducir las ojeras y aplicar una crema hidratante, son excelentes formas de mejorar el servicio y diferenciarte del resto, antes de incluir cualquier servicio los barberos deben ser expertos a la hora de realizarlo, para evitar que el cliente se lleve una mala experiencia. (GoodFellas, 2,015)

2.1.2 Citas programadas

Permite establecer su horario de citas y brindar a los clientes la posibilidad de reservar espacios por su cuenta. Esto le brinda más control sobre su tiempo y lo hace más productivo. Todo lo que necesitas es un programador de citas, el software de programación de citas automatiza el proceso de citas de principio a fin. Puede establecer el horario de la cita para usted y para otros miembros de su equipo. Luego, los clientes pueden reservar, modificar o cancelar citas según los espacios disponibles y sus preferencias personales. Luego podrá realizar un seguimiento y administrar sus citas desde una única ubicación, alguna de sus ventajas es:

Evitar retrasos

Gracias a las funciones de programación y reserva, ya no necesitará depender de correos electrónicos de ida y vuelta para confirmar citas con los clientes. Esto significa que puede concentrarse mejor en su discurso y en las necesidades de sus clientes.

Ciclo de ventas más rápido

Puede programar más reuniones con quienes toman las decisiones adecuadas, cerrar acuerdos más rápido y aumentar los ingresos.

Evite la pérdida de clientes potenciales

Los programadores de citas pueden capturar datos clave como correo electrónico, teléfono y ubicación, evitando el riesgo de perder clientes potenciales de alto valor. Puede conectar estos datos a su base de datos de clientes para crear nuevos contactos lo antes posible.

Reduce las no presentaciones y cancelaciones

El software de programación de citas reduce al mínimo las posibilidades de cancelación. Envía correos electrónicos de confirmación inmediatamente después de reservar una cita y recordatorios periódicos previos a la misma. (Engagebay, s.f.)

2.2 Tecnologías

Una tecnología de desarrollo de software, a veces conocida como herramienta de generación de código o generador de códigos, es un programa informático utilizado por desarrolladores construir, mantener, modificar, apoyar y depurar otros programas, marcos o aplicaciones.

2.2.1 DJANGO

Django es un *framework* web de alto nivel basado en *Python*, diseñado para facilitar el desarrollo rápido de aplicaciones web robustas y escalables. Nació en 2005 como una herramienta para ayudar a los desarrolladores de un sitio de noticias a construir aplicaciones web de manera más eficiente. Su principal ventaja es que permite a los desarrolladores enfocarse en la creación de funcionalidades sin tener que preocuparse por aspectos comunes del desarrollo web, como la gestión de bases de datos o la seguridad, ya que estas características están integradas en el *framework*.

Una de las características fundamentales de Django es que sigue el patrón Modelo-Vista-Plantilla, una variación del patrón Modelo-Vista-Controlador, en este patrón, el modelo representa la estructura de los datos y las interacciones con la base de datos, la "Vista" contiene la lógica de negocio y la plantilla se encarga de la presentación de los datos al usuario. Este enfoque modular hace que el código sea más fácil de mantener y escalar, y permite a los desarrolladores separar claramente la lógica de la presentación.

Django también incluye un mapeador objeto-relacional que permite a los desarrolladores trabajar con bases de datos mediante objetos *Python*, en lugar de escribir consultas SQL. Esto facilita el desarrollo de aplicaciones con bases de datos relacionales, como PostgreSQL, MySQL o SQLite. Además, el ORM de Django hace que el cambio entre diferentes bases de datos sea más sencillo. Junto con su ORM, Django ofrece un panel de administración que se genera automáticamente a partir de los modelos, permitiendo a los administradores gestionar el contenido de manera eficiente y sin necesidad de desarrollar un sistema de administración desde cero.

Uno de los aspectos más destacados de Django es su enfoque en la seguridad. Incluye protecciones integradas contra vulnerabilidades comunes como la inyección SQL, el *Cross-Site Scripting* y el *Cross-Site Request Forgery*, lo que lo convierte en una opción ideal para aplicaciones web que manejan datos sensibles. Además, Django es altamente escalable, lo que lo hace adecuado para proyectos de cualquier tamaño, desde pequeñas aplicaciones hasta plataformas que manejan millones de usuarios, como Instagram y Pinterest.

Django también fomenta las buenas prácticas de desarrollo, como la reutilización de código y la reducción de la duplicación, su vasta comunidad de desarrolladores ofrece una amplia gama de bibliotecas y extensiones que facilitan aún más el desarrollo, además de contar con una excelente documentación oficial. Aunque puede tener una curva de aprendizaje algo empinada para principiantes debido a la cantidad de funcionalidades que ofrece, Django es un *framework* que, una vez dominado, permite crear aplicaciones web potentes y de alta calidad en un tiempo relativamente corto.

2.3 Lenguaje de programación

Es una herramienta que permite desarrollar software o programas para computadora. Los lenguajes de programación son empleados para diseñar e implementar programas encargados de definir y administrar el comportamiento de los dispositivos físicos y lógicos de una computadora. Lo anterior se logra mediante la creación e implementación de algoritmos de precisión que se utilizan como una forma de comunicación humana con la computadora.
(CUAED, UNAM, 2,017)

2.3.1 PYTHON

Python es un lenguaje de programación de alto nivel que ha ganado una inmensa popularidad gracias a su sintaxis simple y legible. Diseñado para ser fácil de aprender y usar, *Python* permite a los desarrolladores escribir menos código para lograr más, en comparación con otros lenguajes de programación. Su enfoque en la claridad del código lo convierte en una excelente opción tanto para principiantes como para profesionales experimentados. Además, es un lenguaje de propósito general, lo que significa que puede utilizarse en una amplia gama de aplicaciones, desde el desarrollo web hasta el análisis de datos y la automatización de tareas.

Una de las características más importantes de *Python* es su capacidad para admitir múltiples paradigmas de programación. Los desarrolladores pueden trabajar con programación orientada a objetos, programación funcional o procedural según sus necesidades. Además, *Python* es un lenguaje interpretado, lo que significa que no requiere un proceso de compilación antes de ejecutar el código. Esto permite a los programadores probar su código en tiempo real, facilitando la depuración y el desarrollo ágil.

La comunidad de *Python* es una de las más activas y colaborativas en el mundo de la programación. Hay miles de proyectos de código abierto desarrollados en *Python*, y la documentación oficial es una de las mejores disponibles, ofreciendo ejemplos claros y guías detalladas para cada característica del lenguaje. Sitios como *Stack Overflow* y *GitHub* cuentan con millones de preguntas, respuestas y proyectos que abarcan todos los aspectos del desarrollo en *Python*.

Además, *Python* tiene conferencias globales como PyCon, que reúne a desarrolladores de todo el mundo para compartir conocimiento y trabajar en proyectos colaborativos. Esta gran comunidad también fomenta el uso de *Python* en la enseñanza, convirtiéndolo en uno de los lenguajes más enseñados en universidades y plataformas de aprendizaje en línea, como Coursera y edX.

Python cuenta con un vasto ecosistema de bibliotecas y *frameworks* que extienden sus capacidades. En el desarrollo web, *frameworks* como Django y Flask hacen que sea fácil crear aplicaciones web robustas. En el ámbito de la ciencia de datos y la inteligencia artificial, librerías como NumPy, Pandas y TensorFlow permiten a los científicos de datos analizar grandes conjuntos de datos y crear modelos de aprendizaje automático de forma eficiente. Además, *Python* es una herramienta clave en la automatización de tareas, desde la administración de sistemas hasta la automatización de pruebas de software con herramientas como Selenium.

La versatilidad de *Python* es también una de sus principales fortalezas. Es compatible con una gran variedad de plataformas, incluidos Windows, macOS y Linux, y se puede usar tanto en proyectos pequeños como en grandes aplicaciones empresariales. La comunidad de *Python* es extensa y activa, proporcionando soporte constante, recursos y actualizaciones

regulares. Para instalar bibliotecas adicionales, *Python* ofrece el gestor de paquetes pip, que facilita la gestión de dependencias y la instalación de nuevas herramientas.

Python tiene dos ramas principales: *Python 2* y *Python 3*. *Python 3* es la versión moderna y recomendada, ya que *Python 2* dejó de recibir soporte oficial en 2020. Por lo tanto, todos los desarrolladores nuevos y los proyectos actuales deberían usar *Python 3* para aprovechar sus mejoras y mantener la compatibilidad a largo plazo.

2.4 Base de datos

Es una recopilación organizada de información o datos estructurados, que normalmente se almacena de forma electrónica en un sistema informático. Normalmente, una base de datos está controlada por un sistema de gestión de bases de datos (DBMS). En conjunto, los datos y el DBMS, junto con las aplicaciones asociadas a ellos, reciben el nombre de sistema de bases de datos, abreviado normalmente a simplemente base de datos. (Oracle, s.f.)

2.4.1 MYSQL

Es un sistema gestor de bases de datos muy conocido y ampliamente usado por su simplicidad y notable rendimiento. Aunque carece de algunas características avanzadas disponibles en otros SGBD del mercado, es una opción atractiva tanto para aplicaciones comerciales, como de entretenimiento precisamente por su facilidad de uso y tiempo reducido de puesta en marcha. Esto y su libre distribución en Internet bajo licencia GPL le otorgan como beneficios adicionales contar con un alto grado de estabilidad y un rápido desarrollo. MySQL está disponible para múltiples plataformas, la seleccionada para los ejemplos de este libro es GNU/Linux.

Sin embargo, las diferencias con cualquier otra plataforma son prácticamente nulas, ya que la herramienta utilizada en este caso es el cliente *mysql-client*, que permite interactuar con un servidor MySQL local o remoto. De este modo es posible realizar todos los ejercicios sobre un servidor instalado localmente o, a través de Internet, sobre un servidor remoto. Para la realización de todas las actividades, es imprescindible que dispongamos de los datos de acceso del usuario administrador de la base de datos. Aunque en algunos de ellos los privilegios necesarios serán menores, para los capítulos que tratan la administración del SGBD será imprescindible disponer de las credenciales de administrador.

MySQL es un SGBD relacional de fácil uso y alto rendimiento, dos características muy valiosas para un desarrollador de sistemas: su facilidad de uso permite la creación de bases de datos con rapidez y sin muchas complicaciones, y su alto rendimiento lo hace sumamente atractivo para aplicaciones comerciales importantes o portales web de mucho tráfico. Si a ello le añadimos la disponibilidad de código y su licencia dual, se comprende que MySQL sea atractivo y accesible para todo el mundo. Estos atributos tienen sus costes: mantenerlo con un alto rendimiento hace algo más lento su desarrollo, por lo que no es el más avanzado en cuanto a prestaciones y compatibilidad con estándares.

MySQL carece de características que muchos otros SGBD poseen. Pero no se debe olvidar que está en continuo desarrollo, por lo que futuras versiones incluirán nuevas características. Por supuesto, para MySQL es más importante la eficiencia que incluir prestaciones sólo por competir o satisfacer a algunos usuarios. (Paul, 2,003)

MySQL es un popular sistema de gestión de bases de datos SQL de código abierto desarrollado, distribuido y respaldado por *Oracle Corporation*. MySQL gestiona una colección estructurada de datos. Una base de datos MySQL le ayuda a agregar, acceder y procesar los datos almacenados en la base de datos. MySQL almacena datos en tablas separadas. Las estructuras de la base de datos están organizadas en archivos físicos optimizados para la velocidad. El modelo lógico, con objetos como bases de datos, tablas, vistas, filas y columnas, ofrece un entorno de programación flexible. La parte SQL de "MySQL" significa "Lenguaje de consulta estructurado", que es el lenguaje estandarizado más común utilizado para acceder a bases de datos. El software MySQL utiliza la licencia GPL (GNU General Public) y es un software de código abierto. (Christudas, 2,019)

Características de MySQL En este apartado enumeraremos las prestaciones que caracterizan a este SGBD, así como las deficiencias de diseño, limitaciones o partes del estándar aún no implementadas.

Prestaciones MySQL es un SGBD que ha ganado popularidad por una serie de atractivas características: Está desarrollado en C/C++, se distribuyen ejecutables para cerca de diecinueve plataformas diferentes, la API se encuentra disponible en C, C++, Eiffel , Java, Perl, PHP, Python, Ruby y TCL, está optimizado para equipos de múltiples procesadores, es muy destacable su velocidad de respuesta, se puede utilizar como cliente-servidor o

incrustado en aplicaciones, cuenta con un rico conjunto de tipos de datos, soporta múltiples métodos de almacenamiento de las tablas, con prestaciones y rendimiento diferentes para poder optimizar el SGBD a cada caso concreto, su administración se basa en usuarios y privilegios, se tiene constancia de casos en los que maneja cincuenta millones de registros, sesenta mil tablas y cinco millones de columnas, sus opciones de conectividad abarcan TCP/IP, sockets UNIX y sockets NT, además de soportar completamente ODBC, los mensajes de error pueden estar en español y hacer ordenaciones correctas con palabras acentuadas o con la letra 'ñ', es altamente confiable en cuanto a estabilidad se refiere.

Para todos aquellos que son adeptos a la filosofía de UNIX y del lenguaje C/C++, el uso de MySQL les será muy familiar, ya que su diseño y sus interfaces son acordes a esa filosofía: "crear herramientas que hagan una sola cosa y que la hagan bien". MySQL tiene como principal objetivo ser una base de datos fiable y eficiente. Ninguna característica es implementada en MySQL si antes no se tiene la certeza que funcionará con la mejor velocidad de respuesta y, por supuesto, sin causar problemas de estabilidad.

La influencia de C/C++ y UNIX se puede observar de igual manera en su sintaxis. Por ejemplo, la utilización de expresiones regulares, la diferenciación de funciones por los paréntesis, los valores lógicos como 0 y 1, la utilización del tabulador para completar sentencias, por mencionar algunos.

Sus limitaciones al comprender sus principios de diseño, se puede explicar mejor las razones de algunas de sus carencias. Por ejemplo, el soporte de transacciones o la integridad referencial (la gestión de claves foráneas) en MySQL está condicionado a un esquema de almacenamiento de tabla concreto, de forma que, si el usuario no va a usar transacciones, puede usar el esquema de almacenamiento "tradicional" (MyISAM) y obtendrá mayor rendimiento, mientras que si su aplicación requiere transacciones, deberá usar el esquema que lo permite (InnoDB), sin ninguna otra restricción o implicación (Dubois, 2,003)

CAPÍTULO III - MARCO METODOLOGICO

Se establece información recopilada, el porqué de nuestro proyecto y algunos de los problemas dados en los requerimientos.

3.1 Método general

Se realizaron investigaciones, análisis, informes, pruebas y validaciones para la problemática que se planteó en el proyecto de la aplicación web de citas para la barbería *Stuard Barbershop*.

3.2 Métodos auxiliares

- Pruebas con aplicación de agendación.
- Estudio de clientes.
- Entrevistas a expertos en tecnologías y barbería.
- Análisis de competencia.

3.2.1 Experimental

Se realizó una prueba con un modelo de aplicación de agendación de citas para barbería, se experimentó y el resultado fue positivo para el dueño del negocio, ya que solo tenía un módulo para el administrador.

3.2.2 Análisis

Se realizó un estudio en algunas barberías para ver si se tiene implementado esta innovación para el área de barberías, también se estudiaron y se hicieron pruebas con aplicaciones beta que hay en internet para dar una idea de lo que sería la nuestra al dueño de la barbería *Stuard Barbershop*.

3.2.3 Síntesis

Se estudiaron comentarios y reseñas sobre algunas aplicaciones similares a la nuestra para tener idea del impacto que se tiene con los clientes, también se analizó como puede afectar el rendimiento de la barbería en su uso.

3.3 Tipo de investigación

Se utilizó la investigación de tipo aplicada, ya que el objetivo es la solución del problema que se tiene con la agendación de citas para poder mejorar la eficiencia y la experiencia del cliente con la aplicación web de citas.

3.4 Prototipo

El prototipo será la representación inicial simplificada y funcional del proyecto que se crea durante la fase del desarrollo para hacer pruebas y validar las ideas, funcionalidades, diseños y usabilidad antes de llegar a tener el producto final e implementarlo.

3.5 Metodología de desarrollo de programación

El proyecto se está realizando con el método cascada, el cual enfoca en cada una de las fases de procesos de desarrollo de software, que son secuenciales y progresivas.

- Requisitos
- Diseño
- Implementación
- Pruebas
- Mantenimiento

CAPÍTULO IV – ANALISIS DE LA APLICACIÓN O SOLUCION

4.1 Descripción del proyecto

El objetivo de este proyecto es desarrollar una aplicación web que facilite la gestión de citas en una barbería, tanto para los clientes como para los barberos. Los clientes podrán registrarse, acceder al sistema y reservar citas en tiempo real con su barbero de preferencia, mientras que los barberos podrán gestionar sus horarios, disponibilidad y citas. El sistema también proporcionará notificaciones para confirmar o recordar las citas y permitirá a los usuarios calificar y reseñar los servicios recibidos. La aplicación estará basada en tecnologías web modernas como HTML5, CSS y JavaScript en el *frontend*. En el *backend*, se empleará una base de datos SQL para almacenar información sobre usuarios, citas y servicios.

4.2 Identificación de usuarios

La aplicación tendrá tres tipos de usuarios principales. Los clientes son aquellos que utilizarán la aplicación para buscar barberos disponibles y reservar citas según su conveniencia. Podrán gestionar sus citas, modificar o cancelar reservas, y dejar reseñas y calificaciones después de cada servicio. Los barberos tendrán acceso a una interfaz donde podrán gestionar su disponibilidad, ver y confirmar sus citas, y actualizar sus servicios. Finalmente, el administrador será responsable de gestionar tanto a los clientes como a los barberos, así como de controlar los servicios que ofrece la barbería, asegurando que el sistema funcione de manera eficiente.

4.3 Modelado del negocio

El modelo de negocio de esta aplicación se centra en las siguientes entidades principales: Clientes, Barberos, Citas, y Servicios. Los clientes podrán registrarse y ver los diferentes barberos disponibles, seleccionar un servicio específico como corte de cabello o afeitado, y reservar una cita en la fecha y hora disponibles. Los barberos podrán gestionar sus citas y disponibilidad a través del sistema, ofreciendo una experiencia organizada y fluida. Cada cita tiene un estado (pendiente, confirmada, cancelada) que se actualiza según las acciones realizadas por el cliente o barbero. Además, el sistema gestionará los servicios que se ofrecen, especificando la duración y el precio de cada uno.

4.4 Metodología de desarrollo de software

Para el desarrollo de esta aplicación web se utilizará el método cascada, ya que trabaja con un enfoque secuencial y estructurado en el que el desarrollo del proyecto avanza de manera lineal a través de distintas fases, donde cada una de estas debe completarse antes de pasar a la siguiente. Las fases del proceso, como la recolección de requisitos, el diseño, la implementación, las pruebas, el despliegue y el mantenimiento, estas se llevan a cabo de manera consecutiva, lo que minimiza la flexibilidad para realizar modificaciones una vez que una fase ha sido completada. Es útil por que tiene la planificación detallada y la previsibilidad ya que son esenciales y así esta proporciona una clara estructura y control sobre cada etapa del proceso.

4.5 Fases

El desarrollo de la aplicación seguirá varias fases durante su inicio, desarrollo e implementación.

4.5.1 Análisis y Requerimientos

Se lleva a cabo una recopilación exhaustiva de las necesidades tanto de los clientes como de los barberos. Se documentan los requisitos funcionales y no funcionales de la aplicación, como la funcionalidad de reserva de citas, la gestión de horarios y la capacidad de que los barberos administren sus servicios. La prioridad es asegurarse de que todos los requisitos estén claramente especificados y documentados, ya que en este enfoque no se permite volver atrás una vez que se pasa a la siguiente fase.

4.5.2 Diseño

Se diseña tanto la arquitectura del sistema como la interfaz de usuario. Esto incluye la creación de diagramas de clases, diagramas de flujo y diseños detallados de cada página de la aplicación. Se definen las tecnologías a utilizar HTML5, CSS3, JavaScript, base de datos SQL y se estructura el flujo de la aplicación, desde el inicio de sesión hasta la gestión de citas.

4.5.3 Desarrollo

Se desarrolla la aplicación basada en los requisitos y los diseños aprobados en las fases anteriores. Se codifica el *frontend* con tecnologías web y el *backend* con una arquitectura adecuada que permita la interacción con la base de datos para gestionar clientes, barberos, citas y servicios, aquí el código es implementado y probado internamente para asegurar que cumple con los requisitos establecidos.

4.5.4 Pruebas

Una vez que el sistema ha sido desarrollado, se somete a pruebas exhaustivas. Se realizan pruebas unitarias, de integración y funcionales para asegurarse de que cada parte del sistema funcione correctamente. También se llevan a cabo pruebas de rendimiento y de seguridad para garantizar que la aplicación web sea robusta y segura.

4.5.5 Implementación y despliegue

La aplicación está lista para ser lanzada en un entorno de producción. Se despliega el sistema para que esté disponible para los usuarios finales. La infraestructura debe estar preparada para soportar el tráfico y se implementan las herramientas necesarias para el monitoreo y la gestión de incidencias.

4.5.6 Mantenimiento

Una vez que la aplicación está en uso, se entra en la fase de mantenimiento, donde el equipo de desarrollo está encargado de realizar mejoras, corregir errores y mantener el sistema actualizado conforme a las nuevas necesidades o tecnologías que puedan surgir. Esta fase es crucial para garantizar que la aplicación siga siendo útil y funcional a largo plazo.

4.6 Características

La aplicación ofrecerá una interfaz de usuario amigable e intuitiva que facilitará la interacción de los clientes y barberos. Se implementarán notificaciones automáticas que enviarán recordatorios a los clientes sobre sus citas próximas por medio de una notificación *push* o un mensaje de *whatsapp*. Los usuarios podrán gestionar sus perfiles, modificar información personal y consultar su historial de citas. Además, se implementará un sistema de calificación para que los clientes puedan evaluar la calidad del servicio recibido y así ayudar a mejorar la experiencia general.

4.7 Artefactos

A lo largo del proyecto, se generarán varios artefactos para documentar y respaldar el desarrollo. Los documentos de requerimientos detallarán las funcionalidades de la aplicación y los casos de uso de cada tipo de usuario. Se creará un diagrama de casos de uso que describirá las interacciones entre los usuarios y el sistema, proporcionando una visión clara de cómo funcionará la aplicación. También se desarrollará un diagrama de clases para modelar las principales entidades del sistema y sus relaciones. Un diagrama de arquitectura describirá la estructura de la aplicación, incluyendo el modelo cliente-servidor y las capas tecnológicas involucradas.

CAPÍTULO V – DISEÑO DE LA APLICACIÓN O SOLUCIÓN

En el presente capítulo se muestran las representaciones gráficas de la aplicación web de la barbería *Stuard Barbershop*, éstas no solo proporcionarán una visión inicial de su funcionamiento, sino que también servirán como guía y punto de partida para los pasos subsiguientes en el proceso de desarrollo.

5.1 Actores

En el contexto de sistemas, los “Actores” se refiere a las entidades que interactúan con la aplicación o los procesos dentro de la misma.

5.1.1 Administrador

El administrador puede gestionar los usuarios y grupos que acceden al inicio de la aplicación y supervisarla como también a la base de datos. Este se define como un super usuario dentro de la aplicación. (IBM Corporation, 2,021)

5.1.2 Barbero

Los barberos son los encargados de interactuar con los clientes. Sus funciones principales son atender y dar los servicios que la barbería ofrece a las personas.

5.1.3 Cliente

Los clientes son las personas que reciben los servicios de la barbería. Sus funciones son agendar las citas para luego recibir el servicio por parte de los barberos.

5.2 Escenarios

Los escenarios representan los lugares donde se llevará a cabo la participación de la aplicación web. La importancia de tener el conocimiento de estos radica que las necesidades varían de acuerdo con la ubicación y por lo tanto tienen un impacto significativo en la creación de acciones por parte de los actores.

5.2.1 Entorno del cliente

La ubicación de este es el lugar o residencia de cada cliente, interactúa con la aplicación de manera que pueda realizar la cita el día y la hora que desee y así no tener que enviar un mensaje o hacer una llamada.

5.2.2 Entorno de instalaciones de la barbería

Este es el lugar donde se verifican las citas realizadas por los clientes y así validar su disponibilidad de acuerdo con lo seleccionado por el cliente.

5.3 Diagrama de casos de uso

Los diagramas de comportamiento permiten capturar, de manera gráfica y detallada, las dinámicas y las relaciones entre los componentes de una aplicación. A través de ellos, podemos visualizar como los objetos, clases o componentes de la aplicación interactúan en respuesta a eventos o estímulos externos, cómo evolucionan a lo largo del tiempo y como se coordinan para lograr un objetivo común.

5.3.1 Diagramas de secuencias de *Stuard Barbershop*

En este apartado, se muestran diagramas de secuencias que tienen como objetivo visualizar y describir la interacción entre los diferentes actores y componentes en una aplicación o proceso específico.

5.3.1.1 Diagrama de secuencia para la agendación de citas

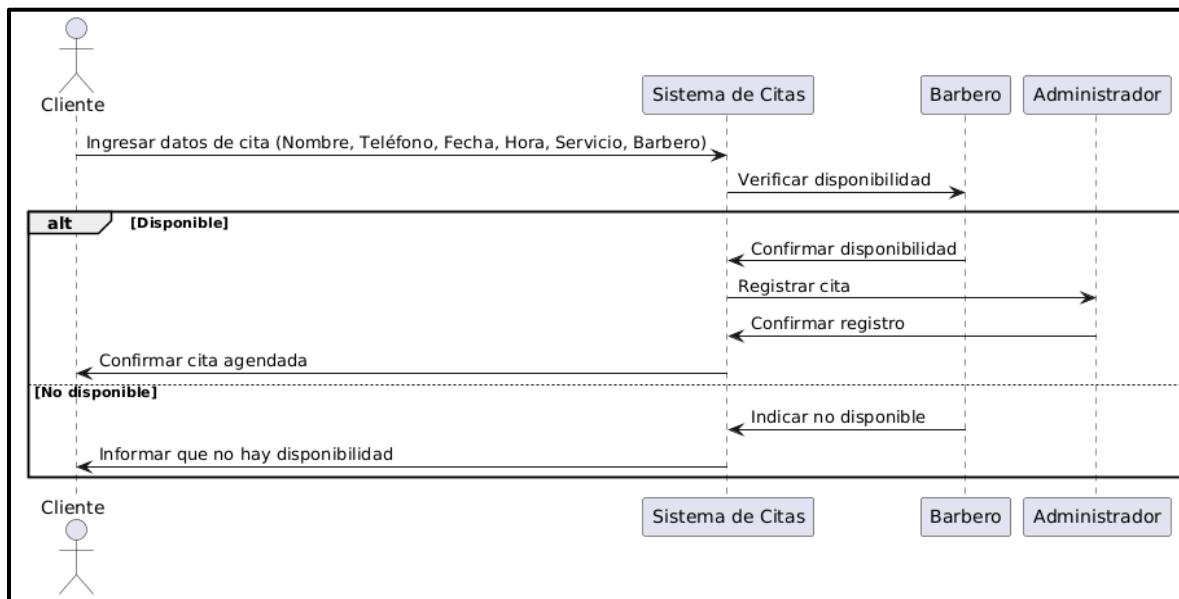


Figura 1 Diagrama de secuencia para la agendación de citas.

Fuente: Elaboración propia.

Este diagrama de secuencia representa un proceso esencial en la agendación de las citas en la barbería. Su importancia radica en que visualiza la interacción entre los actores clave en este proceso, como el cliente y la aplicación web.

5.3.1.2 Diagrama de secuencia para ingresar servicios

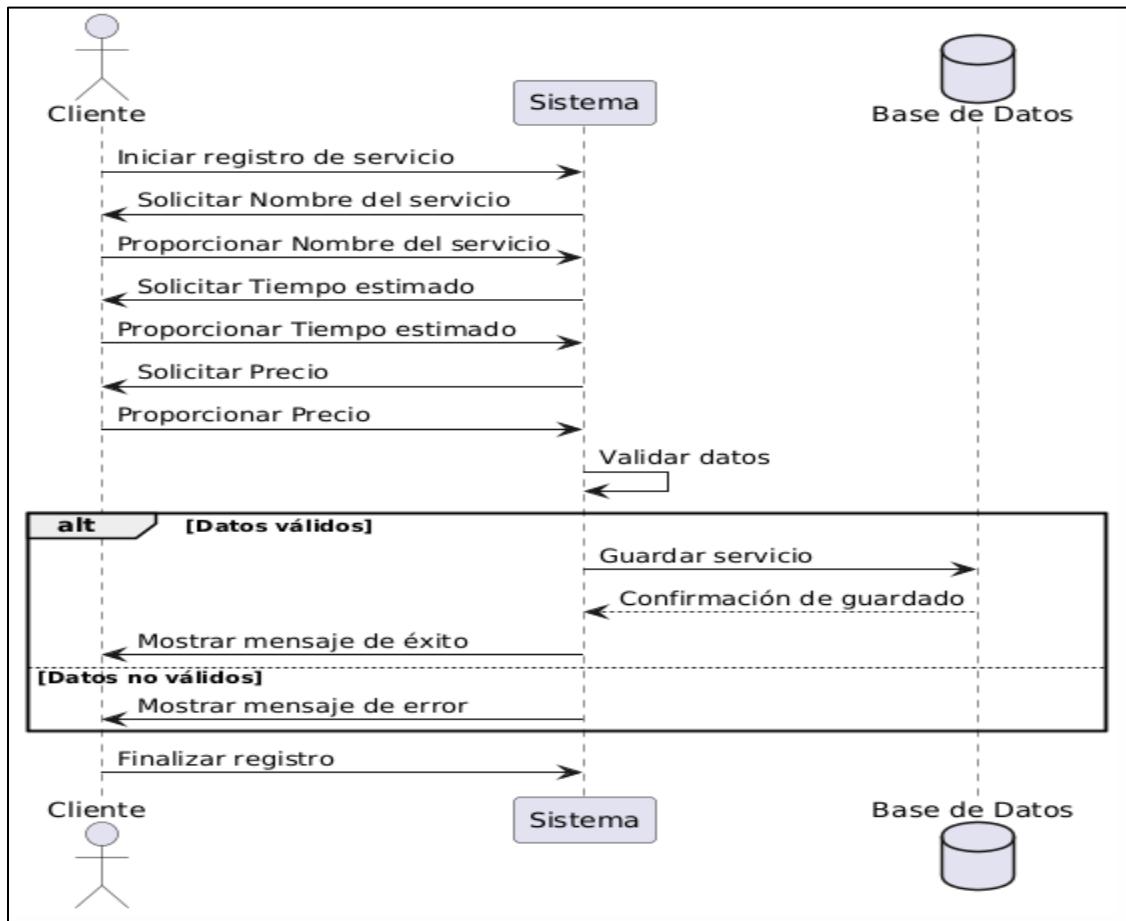


Figura 2 Diagrama de secuencia para ingresar servicios

Fuente: Elaboración propia.

Este diagrama representa el cómo se ingresa un servicio a la aplicación, desde el inicio del registro del servicio, hasta que la aplicación confirma y guarda la información ingresada.

5.3.1.3 Diagrama de secuencia para ingreso de cortes

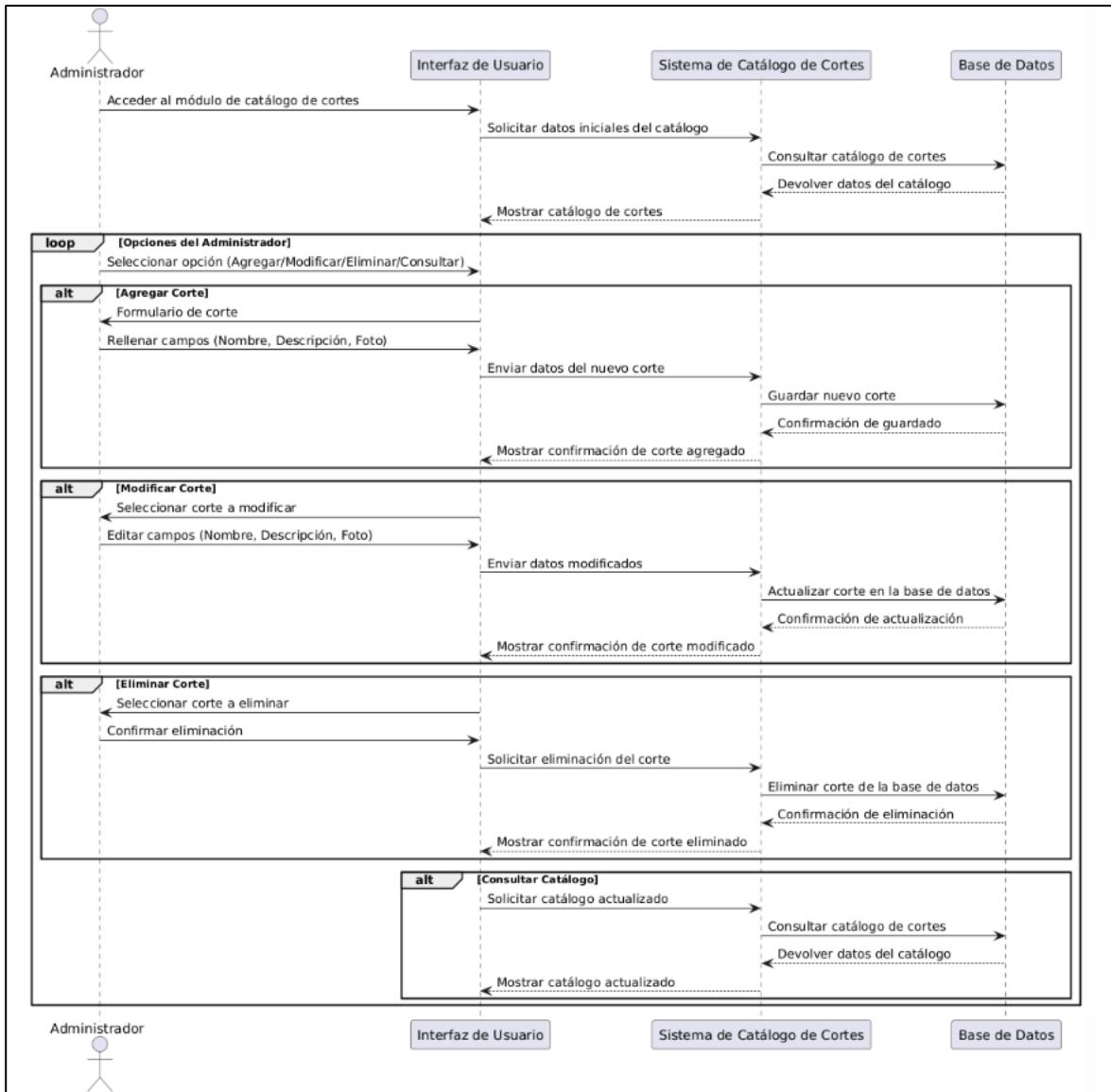


Figura 3 Diagrama de secuencia para ingreso de cortes

Fuente: Elaboración propia.

Este diagrama muestra la secuencia del proceso de ingreso de cortes a la base de datos, y así mostrarlo en el catálogo de cortes que la aplicación posee.

5.3.2.1 Diagrama de actividades de *Stuard Barbershop*

Los diagramas de actividades son una herramienta muy esencial en el campo de la modelización y gestión de procesos. Las representaciones visuales describen de manera clara y concisa el proceso de las acciones, decisiones y flujos de información que conforman un proceso, sistema o aplicación.

5.3.2.2 Diagrama de actividades para agendación de citas

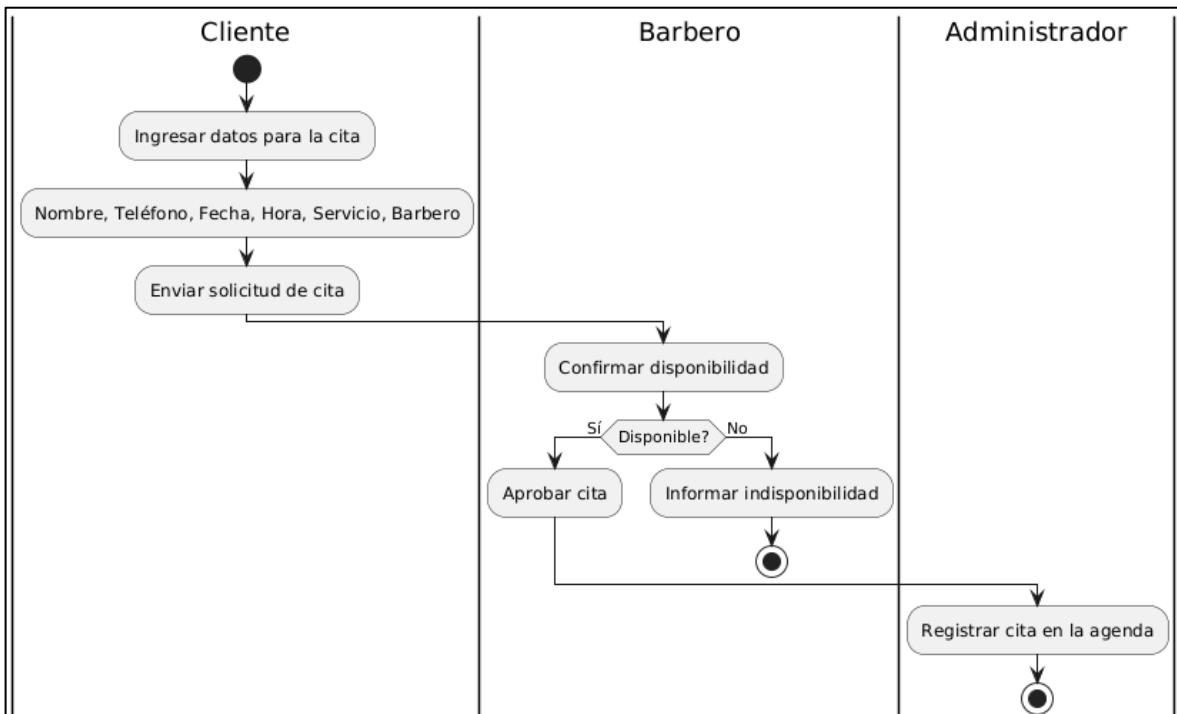


Figura 4 Diagrama de actividades para agendación de citas.

Fuente: Elaboración propia.

El diagrama figura el proceso de agendación de las citas y las funciones que tiene cada actor durante su ejecución, hasta finalizar el registro y se guarde en la base de datos.

5.3.2.3 Diagrama de actividades para ingreso de servicios

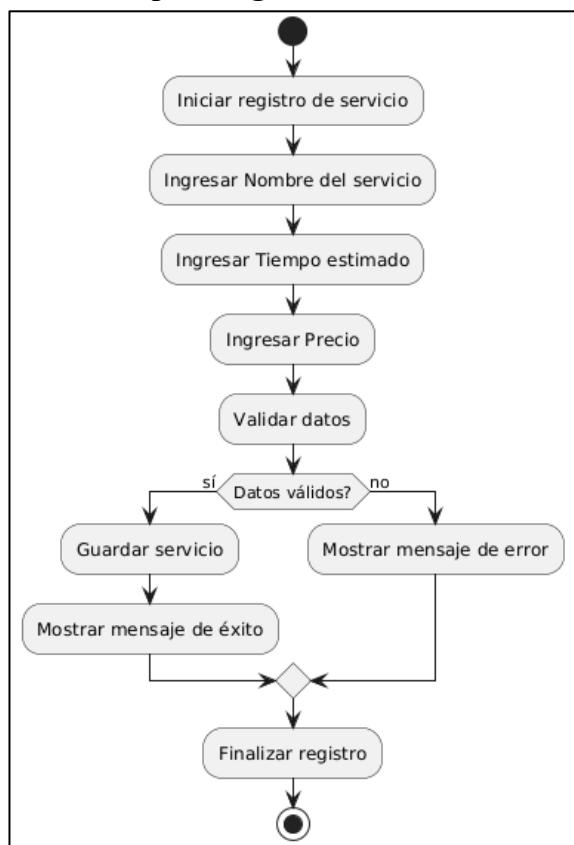


Figura 5 Diagrama de actividades para ingreso de servicios.

Fuente: Elaboración propia.

Este diagrama de actividad nos ayuda a entender como debe ser el proceso de ingreso de servicios a la aplicación, lo muestra de manera simple y sin partes que nos hagan dudar del mismo.

5.3.2.4 Diagrama de actividades para ingreso de cortes

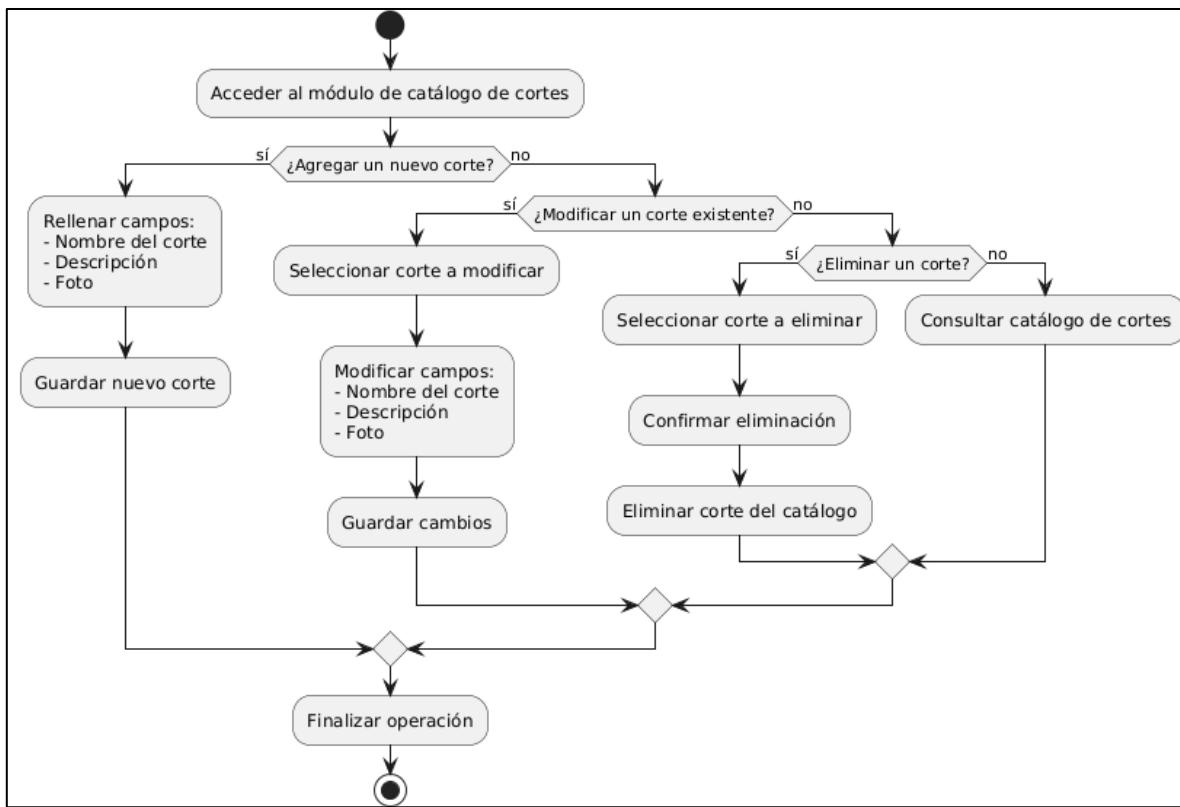


Figura 6 Diagrama de actividades para ingreso de cortes.

Fuente: Elaboración propia.

El diagrama muestra el proceso y los caminos que tomaría la aplicación de acuerdo con las decisiones que se tomen durante la ejecución antes de llenar los campos que se solicitan para poder llegar a finalizar la operación de este módulo.

5.3.3 Diagrama de caso de uso de Stuard Barbershop

En este apartado, veremos la importancia de los diagramas de casos de uso a través de ejemplos concretos. Exploraremos como estos diagramas pueden ayudar a identificar los requisitos funcionales de una aplicación, la prioridad de características más importantes y a establecer una base sólida para el diseño y la implementación.

5.3.3.1 Diagrama de caso de uso para la agendación de citas

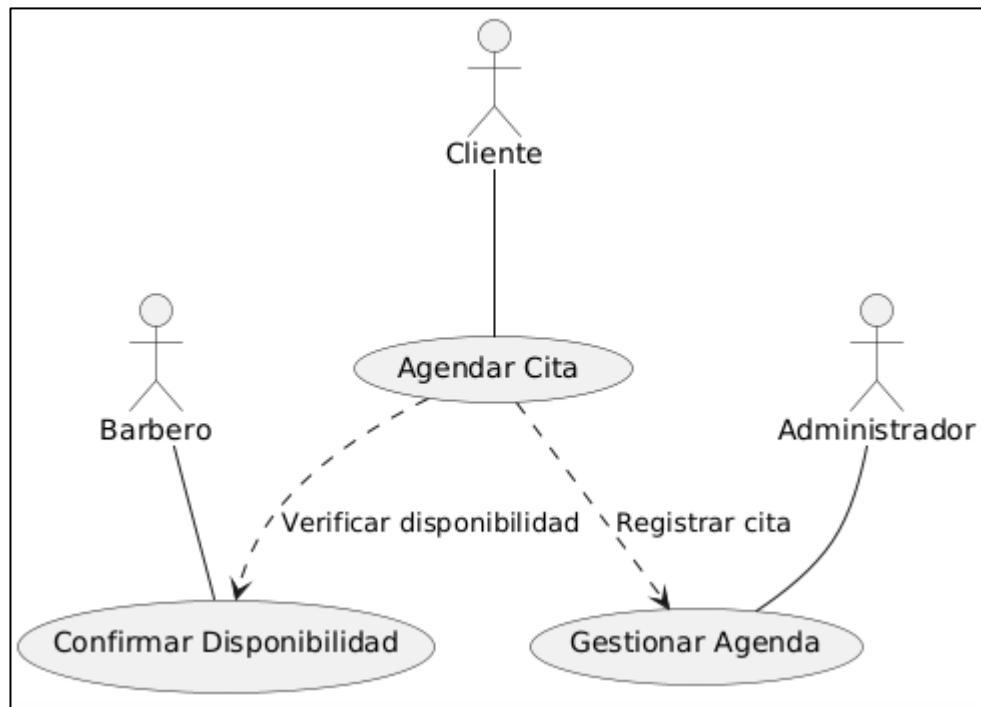


Figura 7 Diagrama de caso de uso para la agendación de citas.

Fuente: Elaboración propia.

Este diagrama muestra de manera organizada como se gestiona la agendación de citas de la barbería. Es importante para mantener las citas de manera precisa y eficiente.

5.3.3.2 Diagrama de caso de uso para el ingreso de servicios

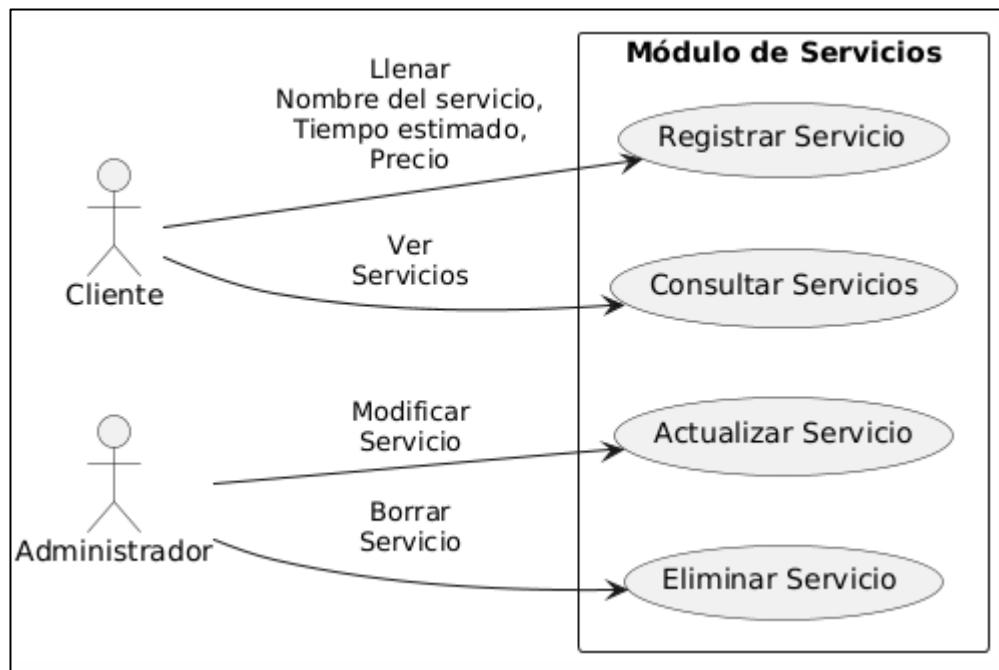


Figura 8 Diagrama de caso de uso para el ingreso de servicios

Fuente: Elaboración propia.

En este diagrama podemos entender que actor es el que realiza cada función de la aplicación web. Es importante entender, para no existan confusiones de uso entre los actores.

5.3.3.3 Diagrama de caso de uso para ingreso de cortes

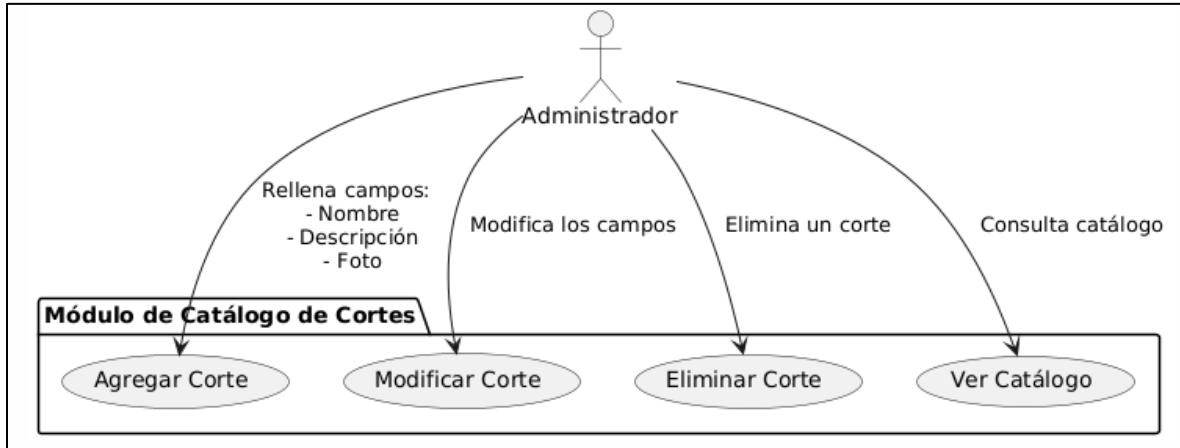


Figura 9 Diagrama de caso de uso para ingreso de cortes.

Fuente: Elaboración propia.

En el diagrama se muestra que solo un actor puede insertar, modificar y eliminar los datos que se obtengan en el módulo del catálogo de cortes. En la figura observamos las funciones que tiene el actor con los procesos que requiere la aplicación.

5.3.4 Diagrama de estados de Stuard Barbershop

Los diagramas de estados son una herramienta crucial en la modelización y el diseño de las aplicaciones, estos permiten visualizar y comprender como un objeto o una aplicación se comporta y responde a diferentes eventos y condiciones en el paso del tiempo.

5.3.4.1 Diagrama de estados para la agendación de citas

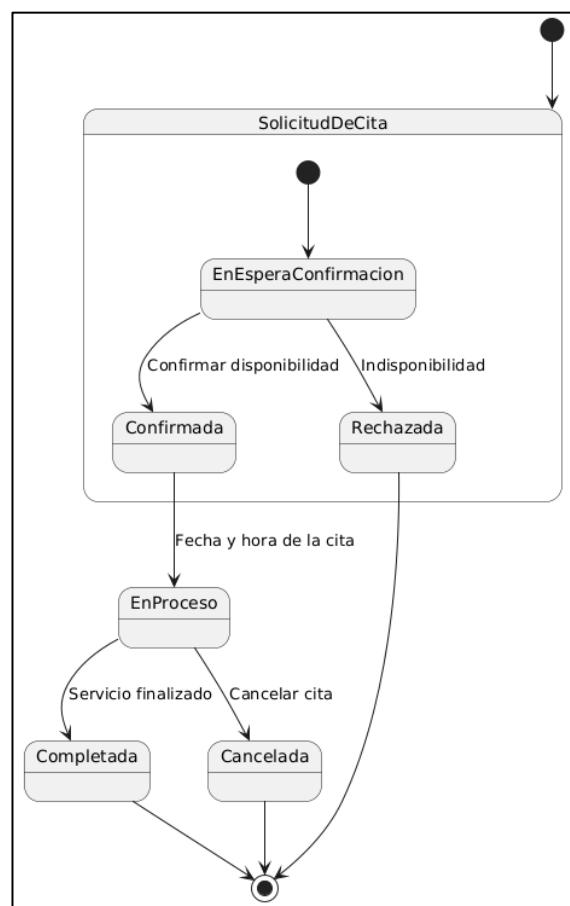


Figura 10 Diagrama de estados para la agendación de citas.

Fuente: Elaboración propia.

El diagrama de estado actual representa el proceso de la agendación de citas, mostrando como las funciones avanzan por los estados, a medida que avanza en el proceso. Estos diagramas son útiles para observar y comprender los cambios de estado y las transiciones en la aplicación, esto facilita el diseño.

5.3.4.2 Diagrama de estados para el ingreso de servicios

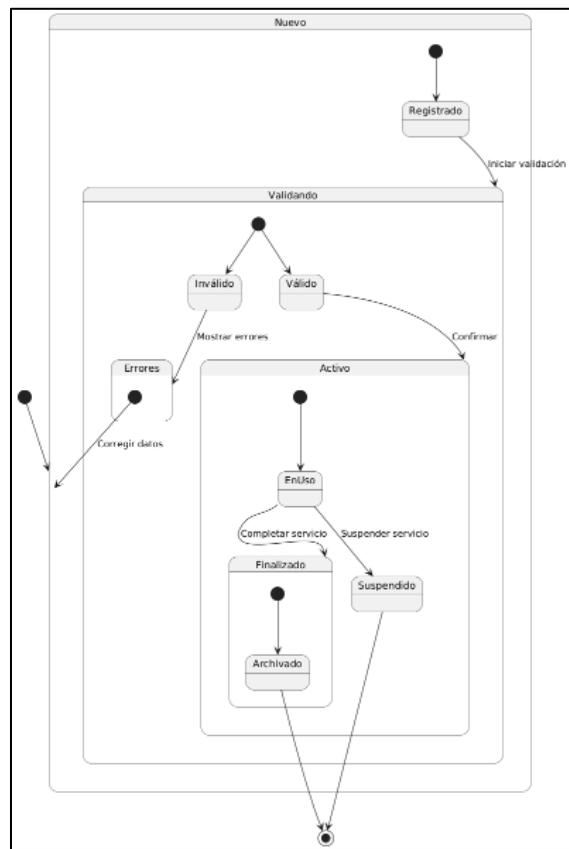


Figura 11 Diagrama de estados para el ingreso de servicios.

Fuente: Elaboración propia.

Este diagrama de estados de servicios proporciona una vista clara de cómo evolucionan los servicios a lo largo del tiempo. Enfatiza la relevancia del registro de servicios ingresados. Facilita el seguimiento y la gestión eficiente de los servicios de la barbería.

5.3.4.3 Diagrama de estados para el ingreso de cortes

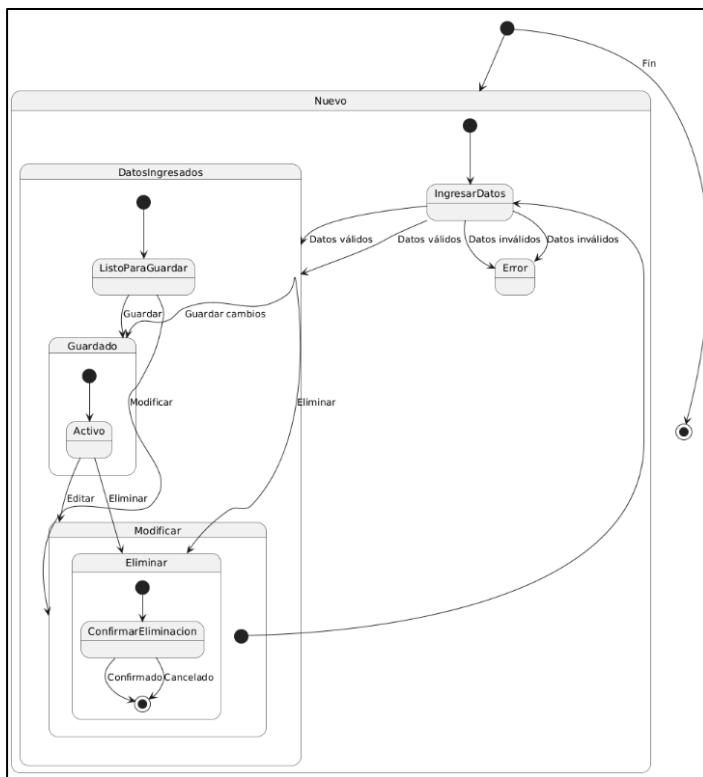


Figura 12 Diagrama de estados para el ingreso de cortes.

Fuente: Elaboración propia.

El diagrama muestra como los procesos avanzan mediante estados y así finalizar con el objetivo de ingreso del corte. Es importante saber cuál es la función de cada estado para mejor uso de la aplicación.

5.3.5 Diagrama de colaboración para Stuard Barbershop

En este apartado, se muestra la importancia de los diagramas de colaboración a través de ejemplos concretos y aplicaciones prácticas. Veremos como estos diagramas pueden ayudar a los desarrolladores y diseñadores a visualizar y planificar la interacción entre objetos en una aplicación, lo que a su vez facilita la detección de posibles problemas de diseño, como ambigüedades en la comunicación o la necesidad de optimizar la eficiencia en la transmisión de información.

5.3.5.1 Diagrama de colaboración para agendacion de citas

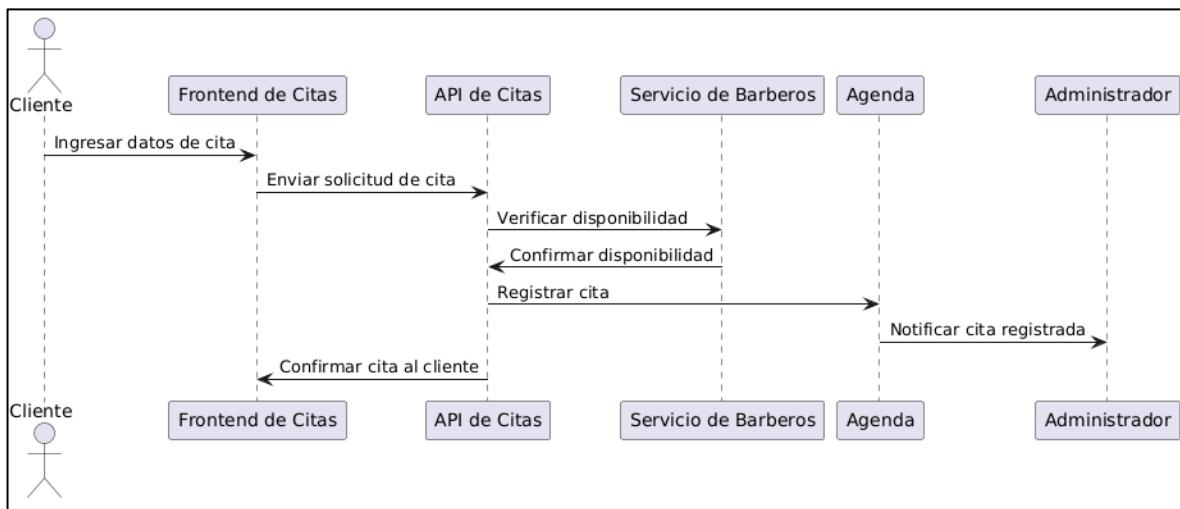


Figura 13 Diagrama de colaboración para agendacion de citas.

Fuente: Elaboración propia.

El siguiente diagrama de colaboración puede utilizarse como herramienta para analizar y mejorar el proceso de agendación de citas al identificar ineficiencias o áreas de optimización.

5.3.5.2 Diagrama de colaboración para ingreso de servicios

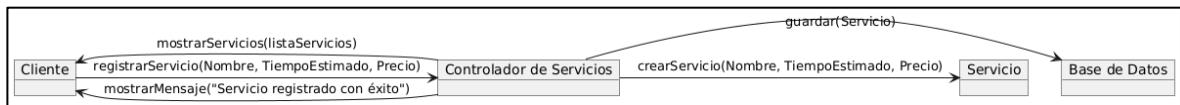


Figura 14 Diagrama de colaboración para ingreso de servicios.

Fuente: Elaboración propia.

Un diagrama de colaboración para el ingreso de servicios es una herramienta versátil que ayuda a comprender, optimizar y comunicar el proceso de ingreso de servicios. Ayuda a una gestión más eficiente, una mayor satisfacción del cliente y una mayor transparencia en el flujo de trabajo relacionado con los servicios.

5.3.5.3 Diagrama de colaboración para ingreso de cortes

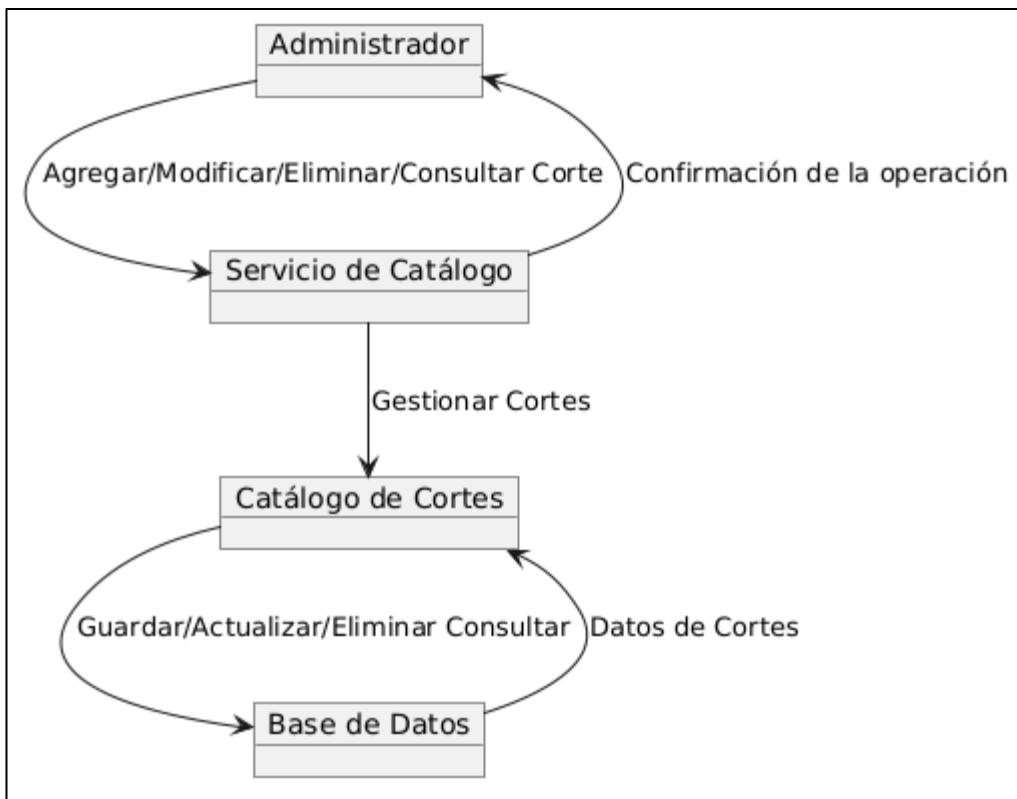


Figura 15 Diagrama de colaboración para ingreso de cortes.

Fuente: Elaboración propia.

Un diagrama de colaboración en el ingreso de cortes es esencial para visualizar y comprender eficazmente el proceso de este módulo. Contribuye a un ingreso más eficiente, una mejor satisfacción y operación en general.

5.3.6 Diagrama de objetos para *Stuard Barbershop*

El diagrama de objetos nos ilustra la estructura estática de *Stuard Barbershop*, incluyendo sus objetos, sus relaciones y estados. Ayuda en la comunicación, el diseño, el desarrollo y el mantenimiento del sistema, lo que contribuye a un desarrollo más eficiente y una mejor comprensión del sistema en su conjunto.

5.3.6.1 Diagrama de objetos para agendacion de citas

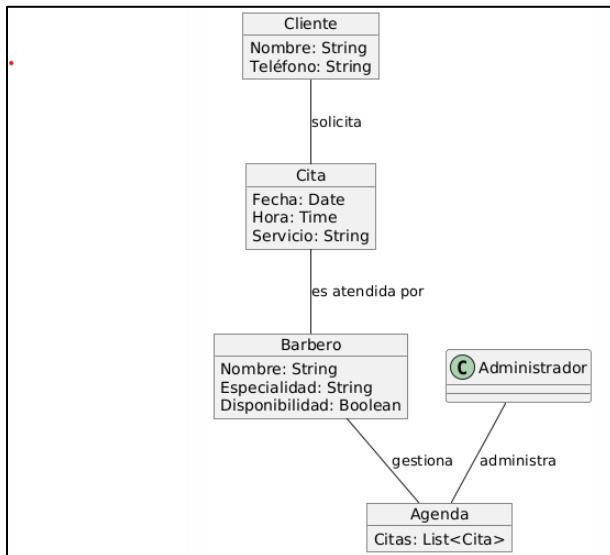


Figura 16 Diagrama de objetos para agendacion de citas.

Fuente: Elaboración propia.

Este diagrama nos muestra la estructura del módulo agendación de citas.

5.3.6.2 Diagrama de objetos para ingreso de servicios

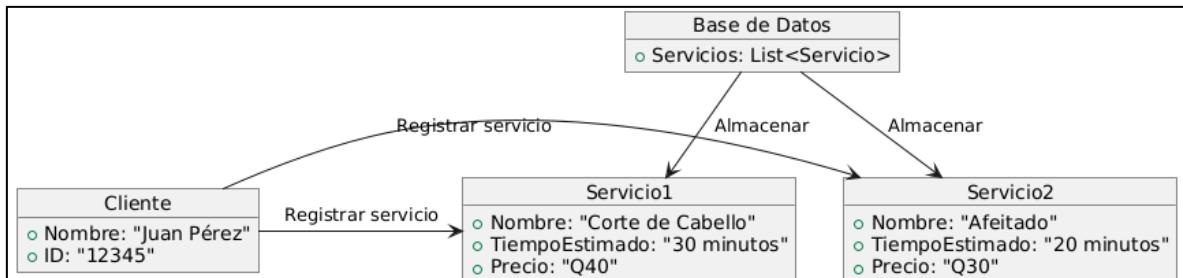


Figura 17 Diagrama de objetos para ingreso de servicios.

Fuente: Elaboración propia.

El diagrama nos ayuda a comprender la estructura exacta del módulo, es importante entender que datos poseen los servicios.

5.3.6.3 Diagrama de objetos para ingreso de cortes

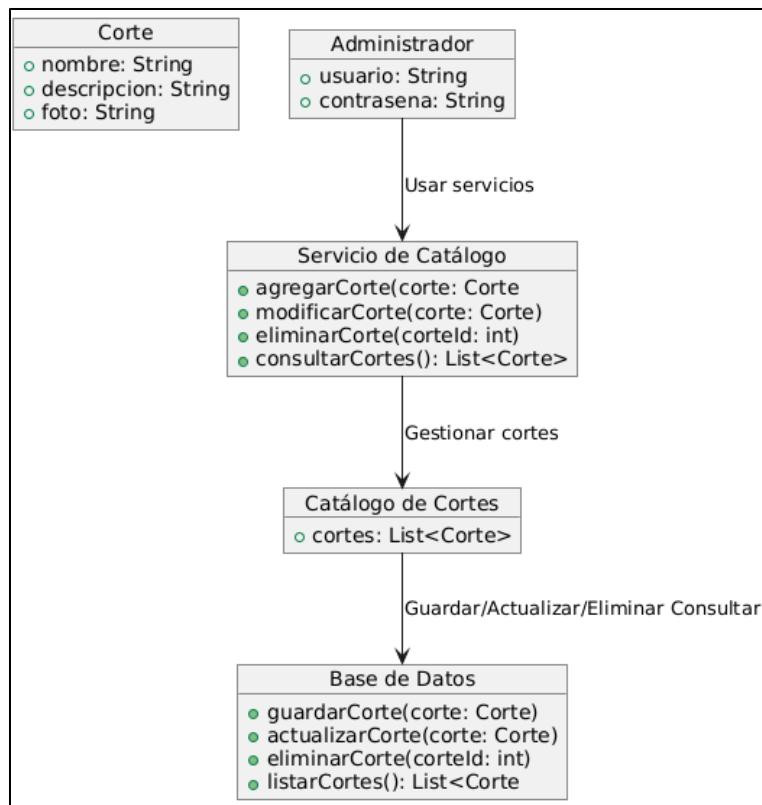


Figura 18 Diagrama de objetos para ingreso de cortes.

Fuente: Elaboración propia.

5.3.7 Diagrama de entidad-relación de *Stuard Barbershop*

El diagrama de entidad-relación visualiza y define las entidades clave involucradas en el proceso de la agendación, servicios, catálogo de cortes, productos, eventos. Entre otros. Esto facilita la comprensión de la estructura de datos subyacente.

5.3.7.1 Diagrama de entidad-relación para agendación de citas

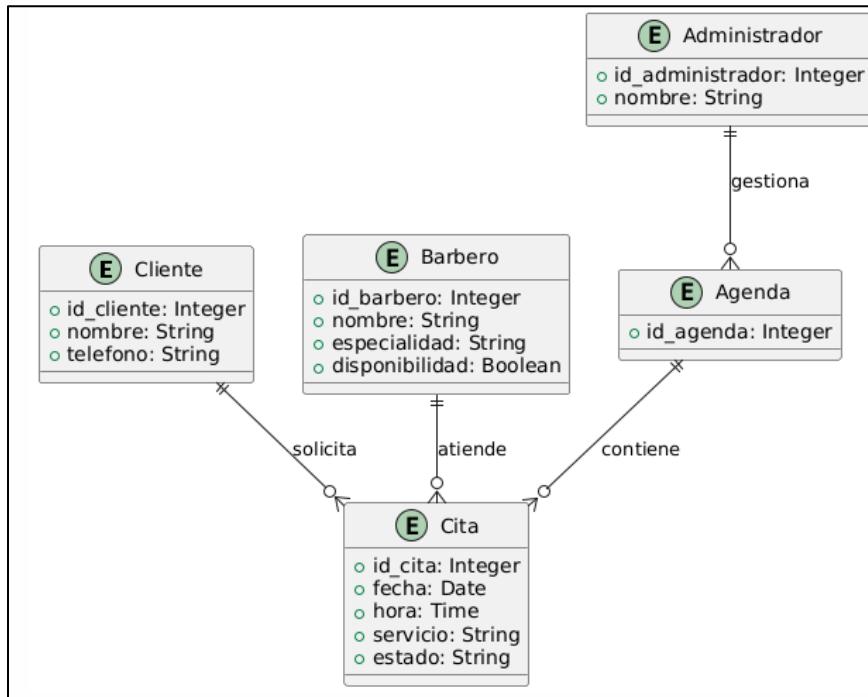


Figura 19 Diagrama de entidad-relación para agendación de citas.

Fuente: Elaboración propia.

El diagrama de entidad-relación muestra las distintas entidades que entran en función, es una forma de visualizar una base de datos al visualizar las tablas creadas.

5.3.7.2 Diagrama de entidad-relación para ingreso de servicios

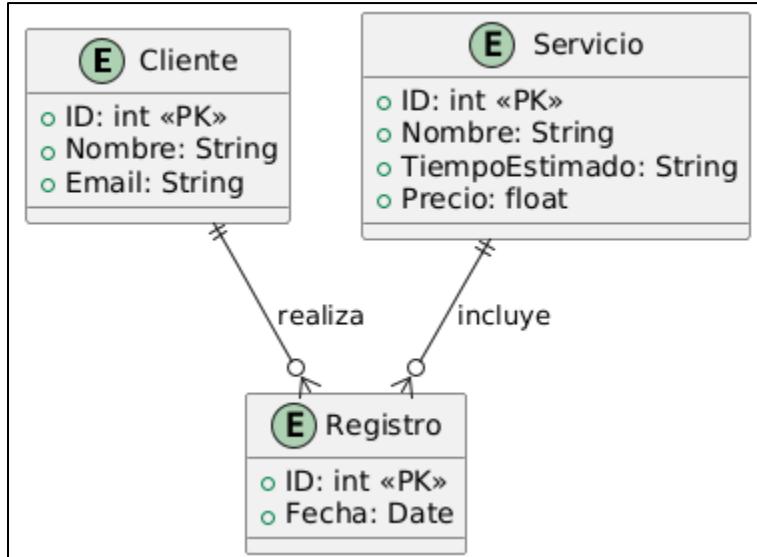


Figura 20 Diagrama de entidad-relación para ingreso de servicios.

Fuente: Elaboración propia.

En este diagrama podemos visualizar los datos que se obtienen de cada entidad que participa en el proceso del ingreso de servicios.

5.3.7.3 Diagrama de entidad-relación para ingreso de cortes

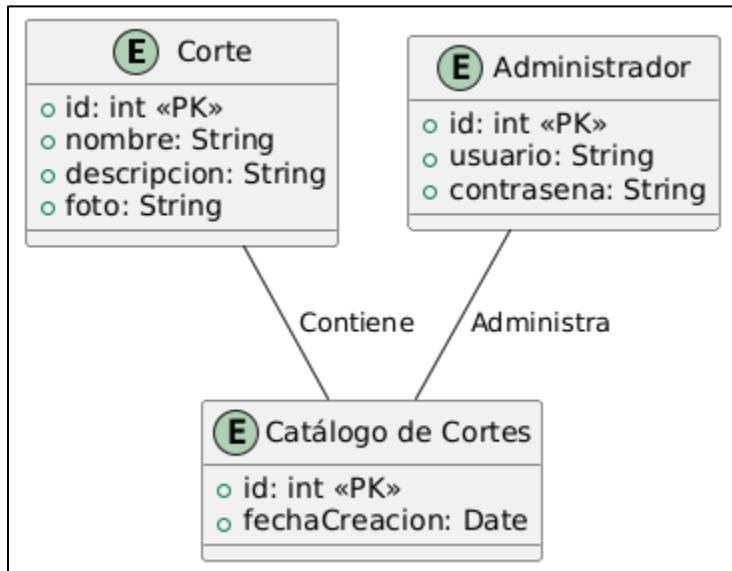


Figura 21 Diagrama de entidad-relación para ingreso de cortes.

Fuente: Elaboración propia.

Este diagrama es el que muestra la base de datos modelada, es importante conocer los datos que requiere cada tabla.

5.3.8 Diagrama de componentes de *Stuard Barbershop*

El diagrama de componentes se utiliza para ilustrar las aplicaciones complejas, muestra el cableado, los artefactos y los componentes de una aplicación física.

5.3.8.1 Diagrama de componentes para agendación de citas

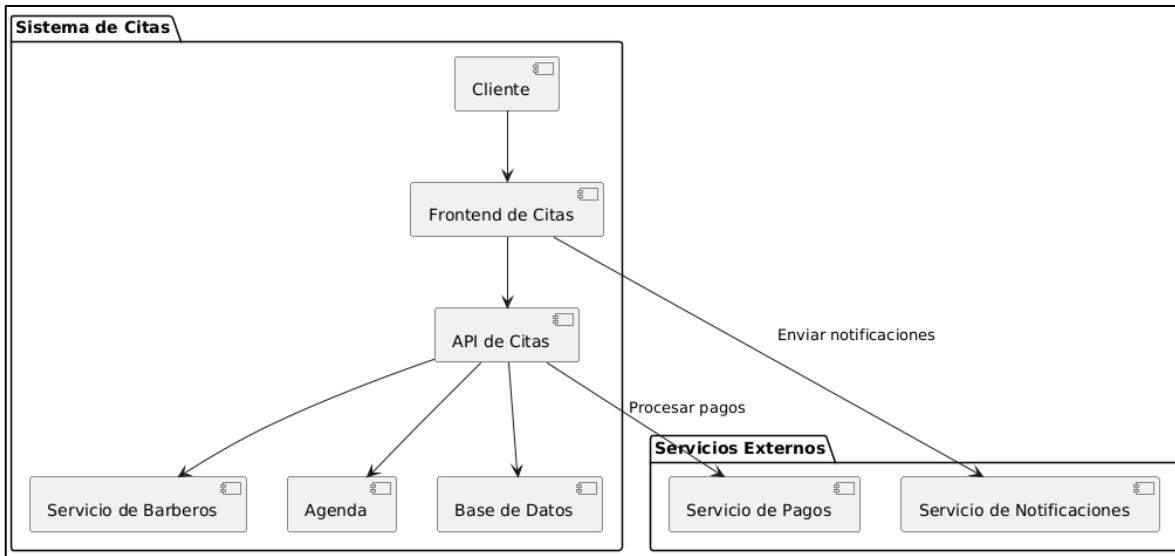


Figura 22 Diagrama de componentes para agendación de citas. Fuente: Elaboración propia.

Este diagrama de componentes muestra la estructura principal de la aplicación de la barbería, destacando los componentes y sus interacciones.

5.3.8.2 Diagrama de componentes para ingreso de servicios

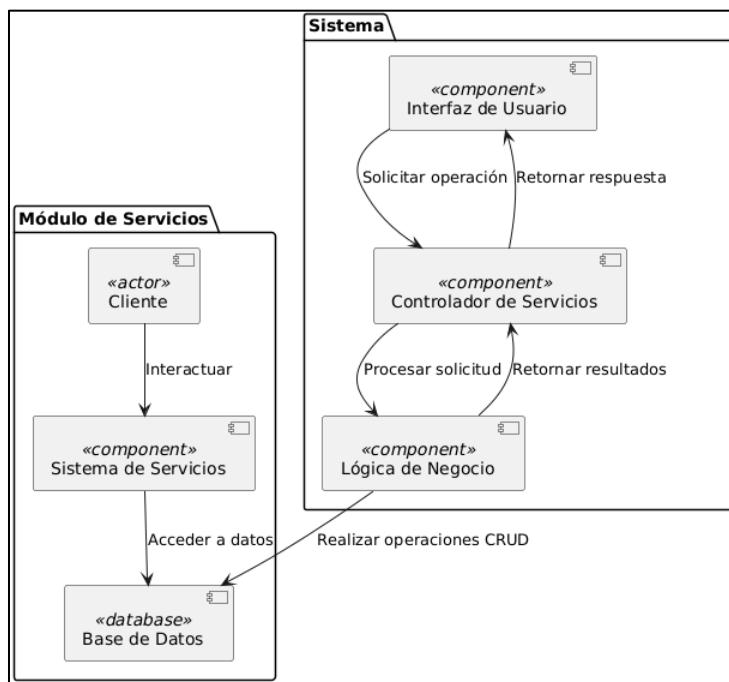


Figura 23 Diagrama de componentes para ingreso de servicios.

Fuente: Elaboración propia.

En el diagrama se visualiza la estructura del módulo de servicios, sus componentes y cada interacción en este proceso.

5.3.8.3 Diagrama de componentes para ingreso de cortes

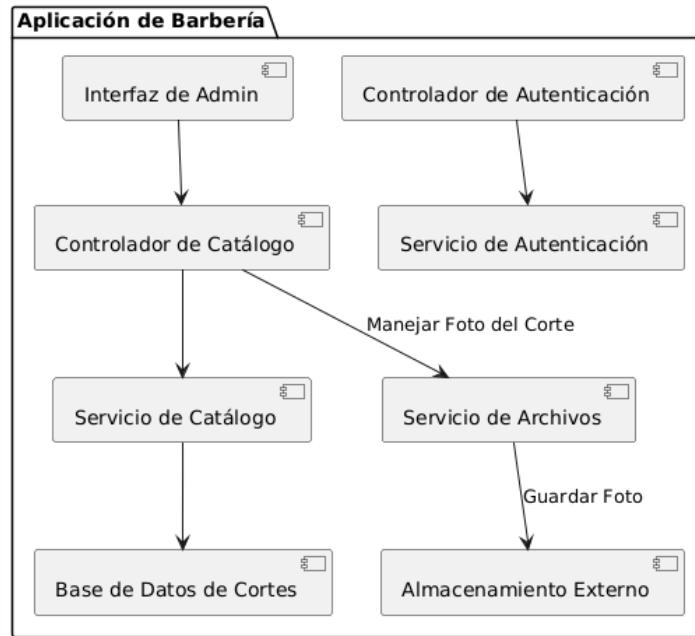


Figura 24 Diagrama de componentes para ingreso de cortes.

Fuente: Elaboración propia.

Este diagrama de componentes refleja la arquitectura de la aplicación y como interactúan los diferentes módulos para el manejo de catálogos de cortes.

5.3.9 Diagrama de infraestructura de *Stuard Barbershop*

Este diagrama representa visualmente los componentes de una aplicación y su interacción, las relaciones entre ellos.

5.3.9.1 Diagrama de infraestructura para agendación de citas

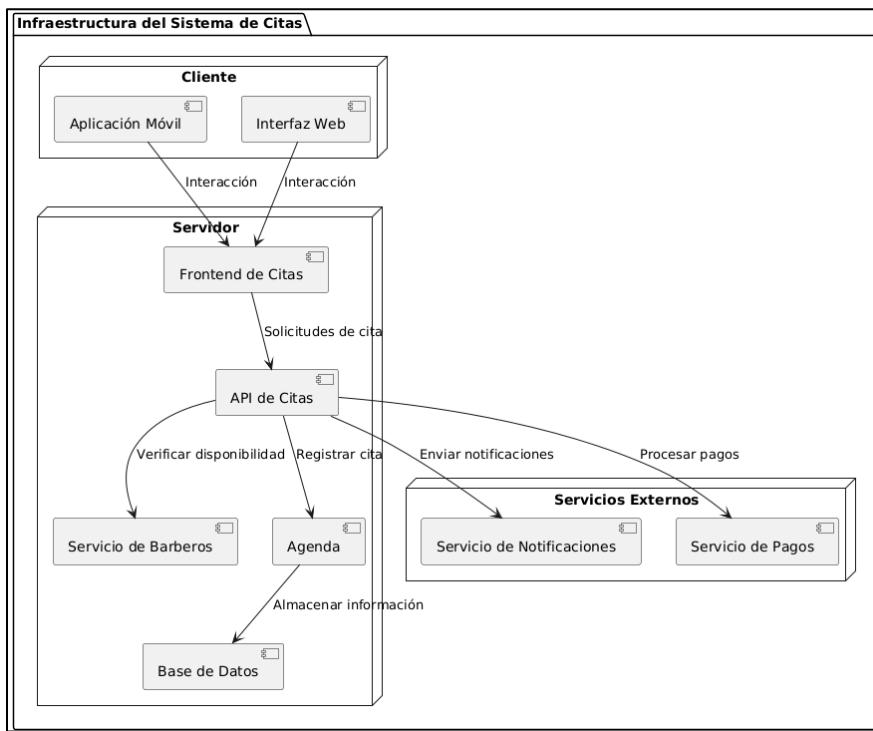


Figura 25 Diagrama de infraestructura para agendación de citas.

Fuente: Elaboración propia.

Este diagrama es esencial para comprender la arquitectura del módulo y planificar su despliegue, identificar sus puntos débiles en la seguridad, facilitar el mantenimiento y la actualización de la aplicación.

5.3.9.2 Diagrama de infraestructura para ingreso de servicios

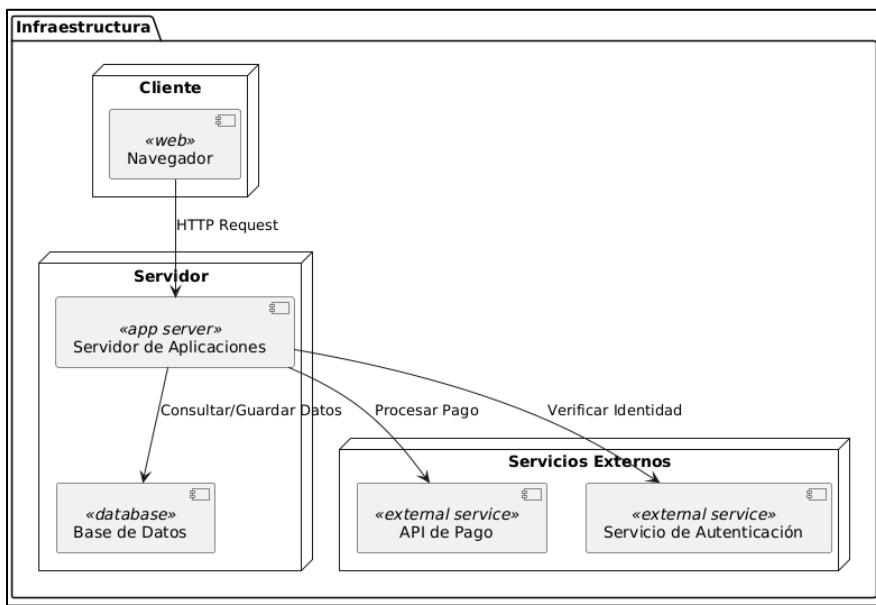


Figura 26 Diagrama de infraestructura para ingreso de servicios.

Fuente: Elaboración propia.

El diagrama muestra la comprensión del módulo de servicios, su arquitectura y el despliegue. Identificamos lo que utilizamos, tanto físico como lógico.

5.3.9.3 Diagrama de infraestructura para agendación de cortes

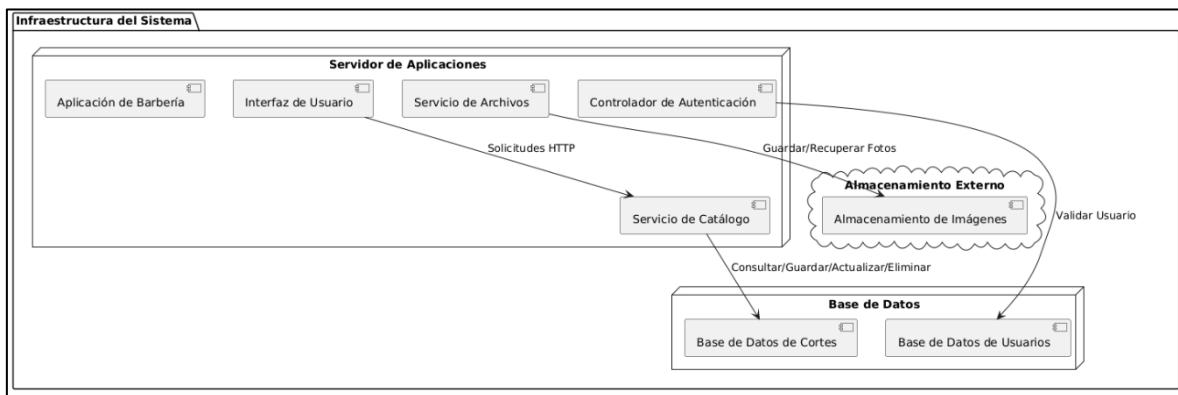


Figura 27 Diagrama de infraestructura para agendación de cortes.

Fuente: Elaboración propia.

Se muestra el diagrama con los elementos tanto físicos como virtuales que son utilizados en el proceso de ingreso al catálogo de cortes. Es esencial conocer estos para tener conocimiento de lo que vamos a ejecutar.

5.3.10 Diagrama de clases de *Stuard Barbershop*

El diagrama de clases es en el que se representan las clases, atributos, métodos y las relaciones entre ellas dentro de la aplicación web. Es clave como elemento de modelado en el desarrollo orientado a objetos para visualizar la estructura estática de la aplicación.

5.3.10.1 Diagrama de clases para agendación de citas

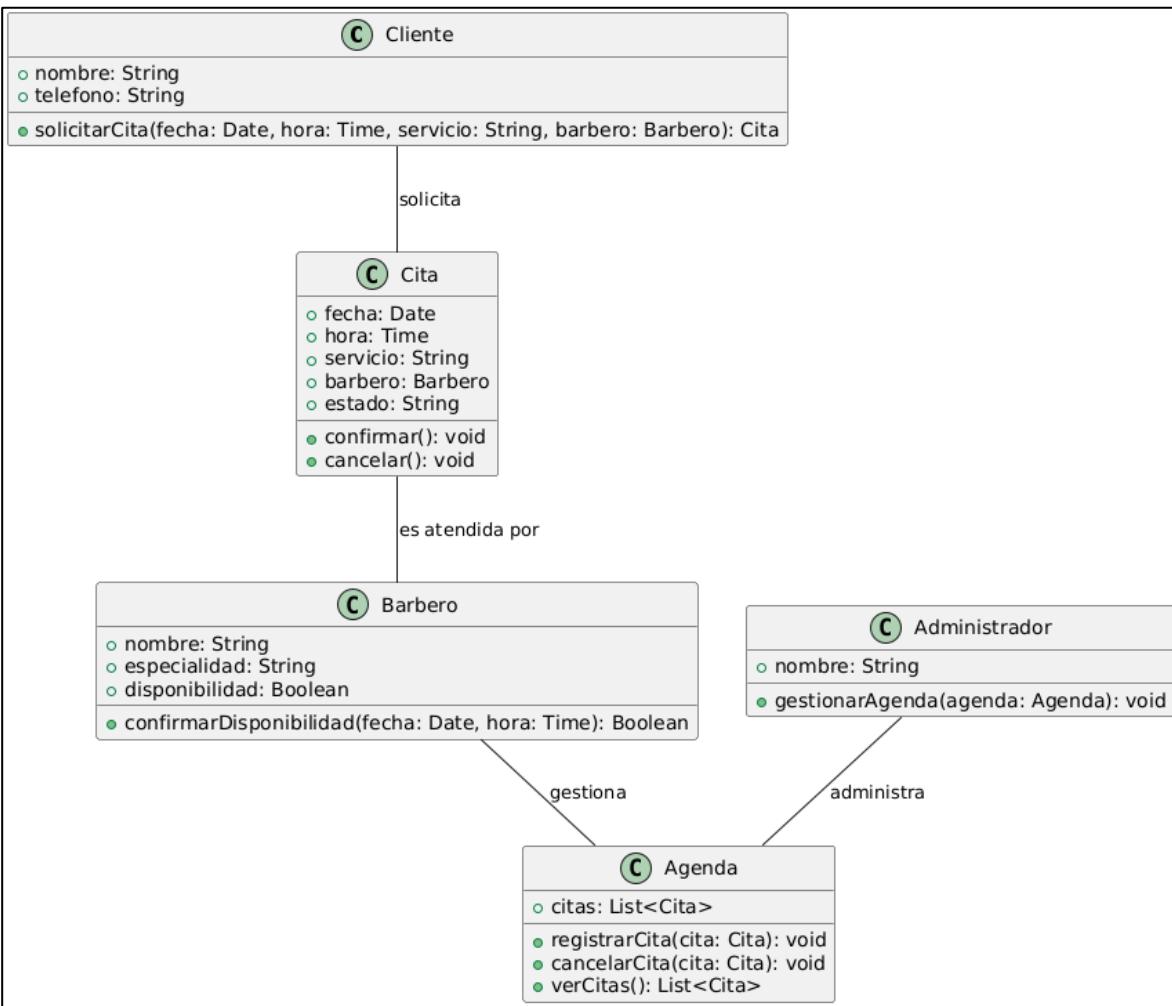


Figura 28 Diagrama de clases para agendación de citas. Fuente: Elaboración propia.

El diagrama es útil para definir la estructura de la aplicación y planificar como interactúan las distintas partes del sistema.

5.4 Maquetado de la aplicación web *Stuard Barbershop*

Tomando en cuenta las necesidades de la barbería *Stuard Barbershop* se define el diseño del software para cada actividad que lo requiera. Siendo participes cada uno de los actores en los distintos escenarios que se puedan encontrar.

5.4.1 Ingreso al sistema



Figura 29 Ingreso al sistema

Fuente: Elaboración propia.

En la presente figura podemos observar los requerimientos para acceder a la aplicación web y los datos requeridos para registrar una cuenta de usuario cliente.

5.4.2 Home

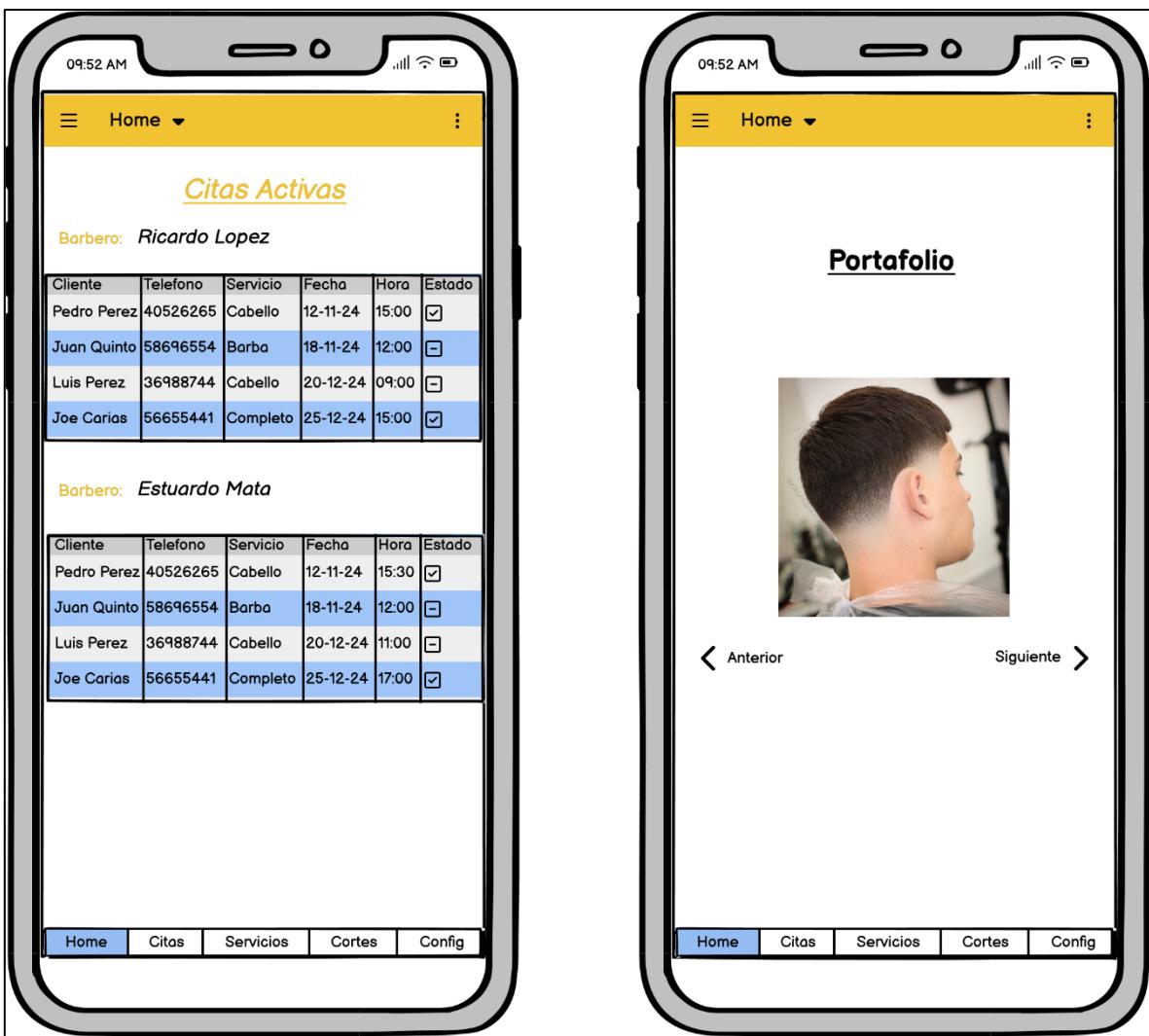


Figura 30 Home

Fuente: Elaboración propia.

Una vez el usuario cliente o administrador ingreso a la aplicación, se muestra el módulo de home según sea el tipo de usuario.

5.4.3 Agendar cita

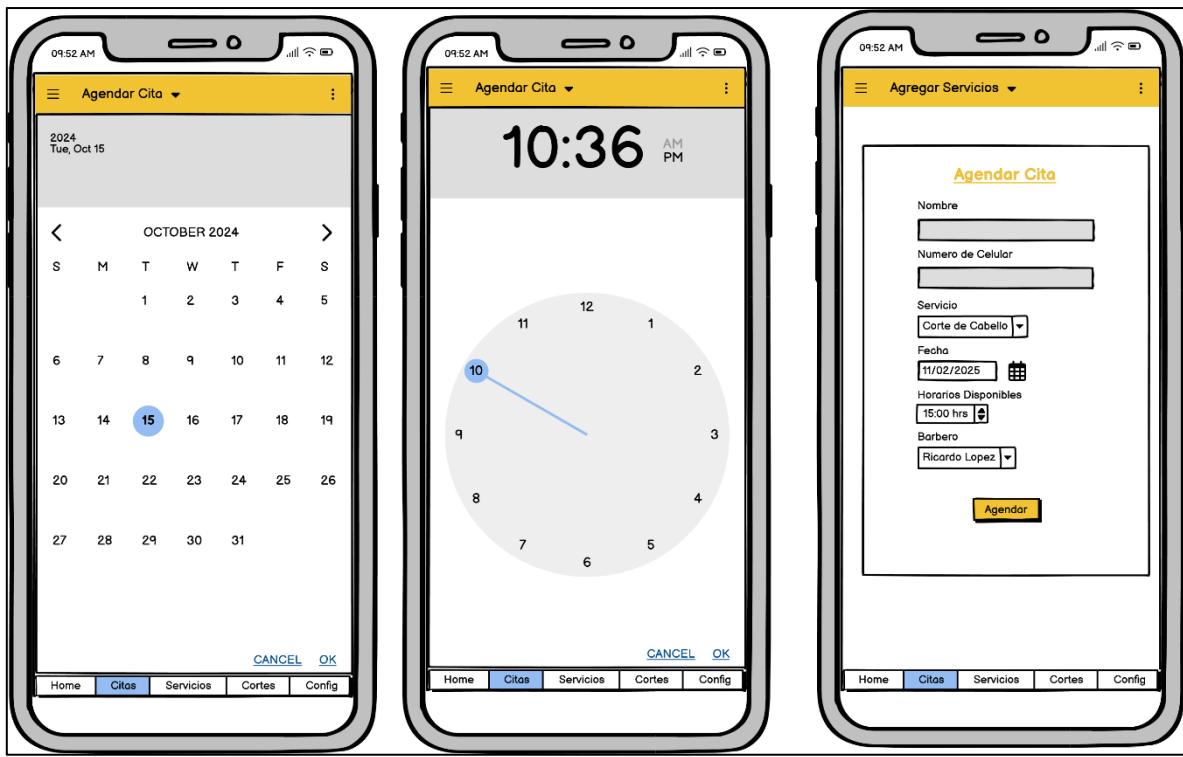


Figura 31 Agendar cita

Fuente: Elaboración propia

En esta vista podemos agendar la cita que requerimos, llenando los datos solicitados como, que servicio requerimos, la fecha y hora en la que queremos recibirla y por último el barbero de nuestra preferencia.

5.4.4 Listado de citas

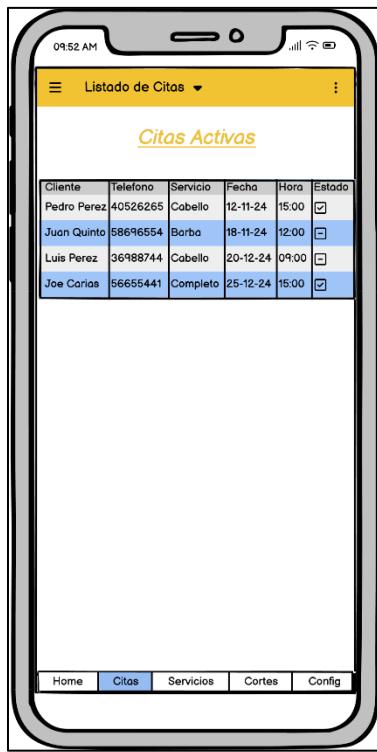


Figura 32 Listado de citas

Fuente: Elaboración propia.

La vista de lista de citas en una aplicación web ayuda a mantener un registro detallado de los clientes y sus interacciones con la agendacion, lo que puede tener un impacto positivo en la reserva de citas y satisfacción del cliente al agendarla.

5.4.5 Agregar servicios y listado

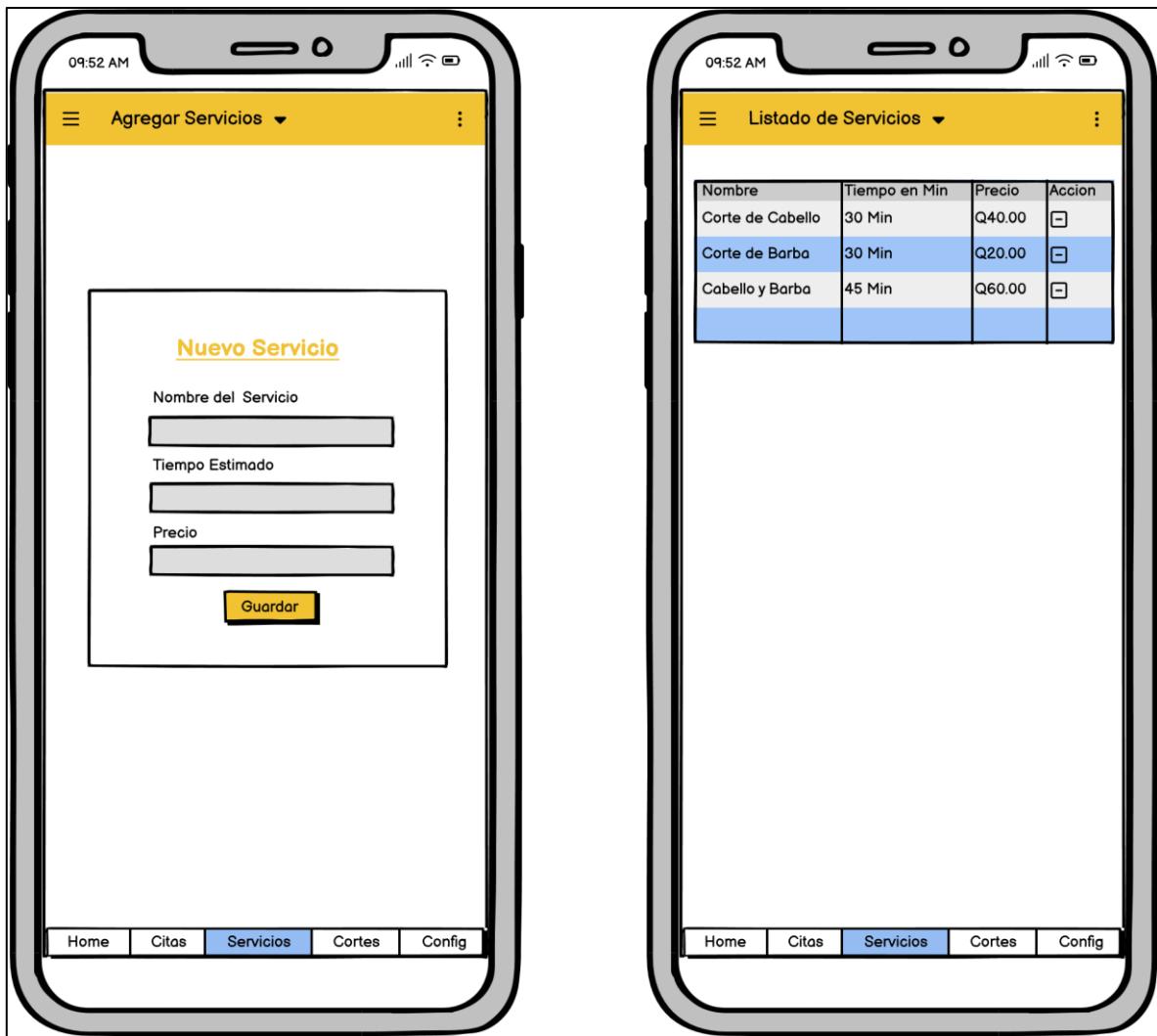


Figura 33 Agregar servicio y listado

Fuente: Elaboración propia.

Este módulo nos muestra dos vistas las cuales nos permiten agregar nuevos servicios, guardarlos y mostrarlos en el listado de este.

5.4.6 Agregar cortes y catalogo

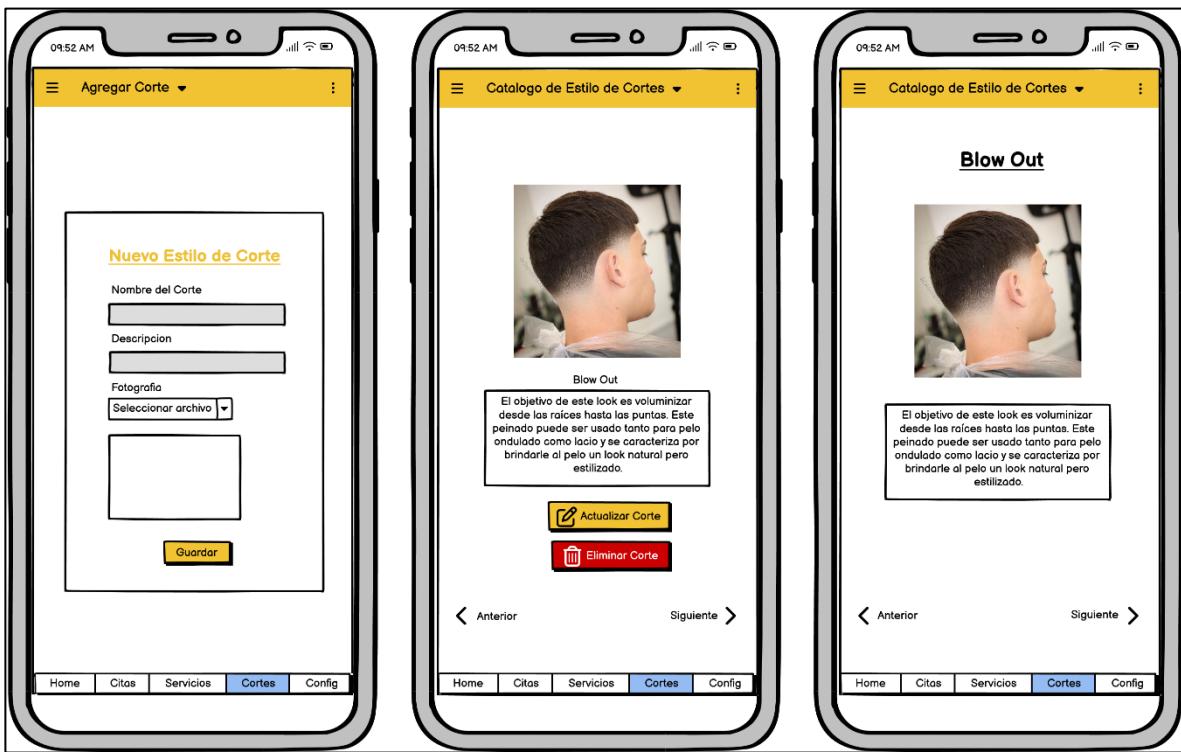


Figura 34 Agregar cortes y catalogo

Fuente: Elaboración propia.

El módulo de cortes nos permite agregar cortes y actualizarlos, para mostrarlos en un catálogo, con su nombre y una descripción que nos ayuda a entender mejor el porqué de este estilo de corte.

5.4.7 Agregar producto y catalogo

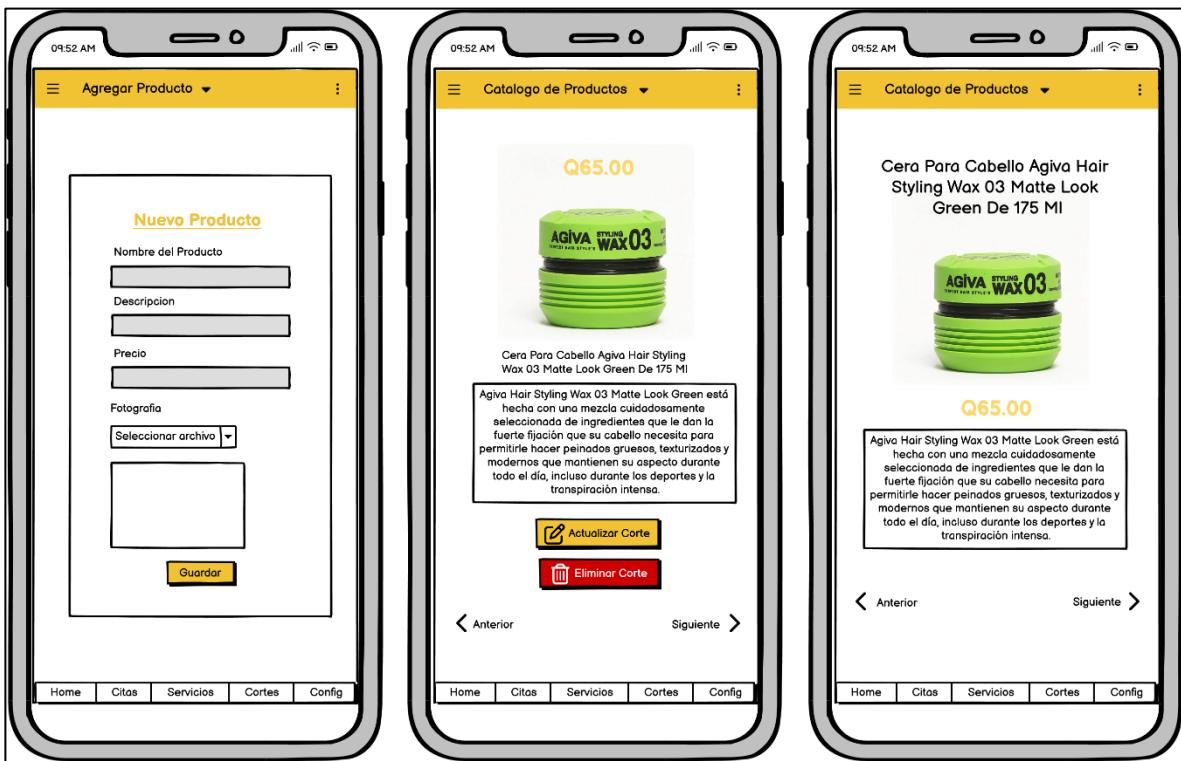


Figura 35 Agregar producto y catalogo

Fuente: Elaboración propia.

El módulo de productos nos permite agregar, actualizar cada uno de estos, el catálogo nos muestra el nombre, el precio, una imagen y una descripción de este.

5.4.8 Reporte de servicios



Figura 36 Reporte de servicios

Fuente: Elaboración propia.

Este módulo reporta todos los servicios que fueron brindados en fechas específicas y por barbero, se ven los totales de ingresos.

CAPÍTULO VI – DESARROLLO E IMPLEMENTACION DE LA APLICACIÓN O SOLUCION

En este apartado, se proporcionarán las pruebas e ilustraciones que sostienen el funcionamiento completo del sistema que se desarrolló y ha sido finalizado. En éstas se recaudan diversas evidencias de distintas formas que certifican que el sistema es operativo y cumple con todos los requisitos y funcionalidades que fueron definidos previamente. Es esencial que los usuarios que estén interesados en maximizar la capacidad y las funcionalidades de un sistema revisen cuidadosamente el capítulo puesto que puede proporcionar una guía detallada sobre cómo utilizar el sistema de una manera más efectiva.

Las vistas son la interfaz principal por la cual los usuarios interactúan con un sistema de información. Permiten a los usuarios visualizar y manipular los datos, realizar acciones, y acceden a las funcionalidades del sistema. La forma en que están diseñadas y presentadas las vistas influye directamente en la experiencia del usuario. Estas también actúan como un medio de comunicación entre el sistema y el usuario.

6.1 Ingreso a la aplicación o login



Figura 37 Ingreso a la aplicación o login Fuente: Elaboración propia.

Para el ingreso del sistema se deberá colocar las credenciales de inicio de sesión que fueron previamente creadas y luego confirmar el ingreso.

6.2 Vista principal o home

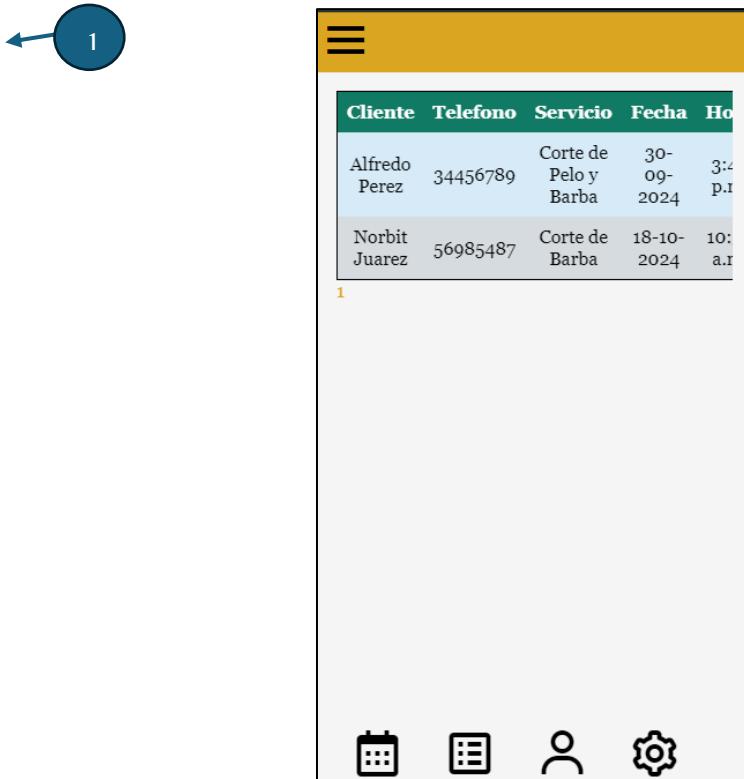


Figura 37 Vista principal o home

Fuente: Elaboración propia.

La vista principal luego de acceder mediante el login muestra los siguientes elementos.

1. Botón para extender el menú, funciona para visualizar las opciones del menú.

Se visualizan las citas activas en el apartado del home.

6.3 Menú de la aplicación web



Figura 38 Menu de la aplicacion web

Fuente: Elaboración propia

En la vista del menú se muestra las siguientes opciones:

Citas: En esta se muestra la opción para agendar cita, ver el listado de citas, modificarlo y eliminarlo.

Productos: Se muestran las opciones para agregar productos y el listado de producto.

Servicios: En esta se muestran las opciones del listado de servicios y agregar servicios.

Estilos: Este es la sección de cortes, donde se muestra el catálogo de cortes y la opción de agregar cortes.

Reportes: Se muestran los reportes de los servicios que fueron realizados y el total de ganancias.

6.4 Listado de citas

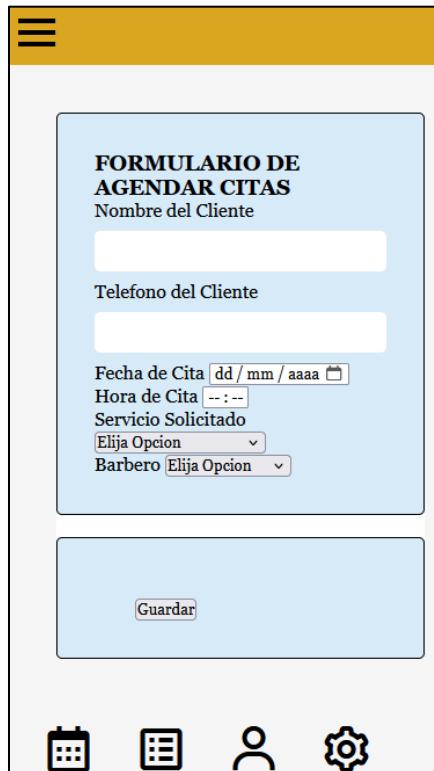
Cliente	Telefono	Servicio	Fecha
Alfredo Perez	34456789	Corte de Pelo y Barba	30-09-2024
Norbit Juarez	56985487	Corte de Barba	18-10-2024
Antonio Perez	52322545	Corte de Pelo y Barba	19-12-2024

Figura 39 Listado de citas

Fuente: Elaboración propia.

En este apartado podemos visualizar las citas que han agendado los clientes, en la tabla se muestra el nombre del cliente, el número de teléfono, el servicio seleccionado por el cliente, la fecha para la que se agendo la cita, la hora de la cita en la fecha seleccionada y el estado de la cita, si ya fue confirmada o fue rechazada.

6.5 Agendar citas

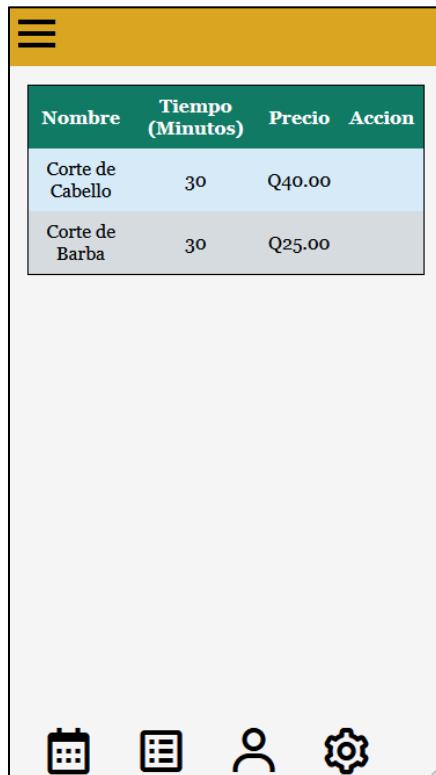


The image shows a mobile application interface titled "FORMULARIO DE AGENDAR CITAS". At the top, there is a yellow header bar with three horizontal lines on the left. Below the header, the title "FORMULARIO DE AGENDAR CITAS" is centered in a blue box. Inside this box, there are several input fields: "Nombre del Cliente" with a text input field, "Telefono del Cliente" with a text input field, "Fecha de Cita" with a date input field showing "dd / mm / aaaa" and a calendar icon, "Hora de Cita" with a time input field showing "-- : --", "Servicio Solicitado" with a dropdown menu showing "Elija Opcion", and "Barbero" with another dropdown menu showing "Elija Opcion". Below these fields is a large blue button labeled "Guardar". At the bottom of the screen, there are four icons: a calendar, a list, a person, and a gear.

Figura 40 Agendar citas Fuente: Elaboración propia.

Se muestran los campos requeridos para agendar la cita, todos los campos deben ser llenados, se eligen las opciones que se prefieren y se le da clic en el botón de guardar.

6.6 Listado de servicios



The screenshot shows a mobile application interface. At the top is a yellow header bar with three horizontal lines on the left side. Below it is a white table with a green header row containing columns for 'Nombre', 'Tiempo (Minutos)', 'Precio', and 'Accion'. Two rows of data are listed: 'Corte de Cabello' with time 30 and price Q40.00, and 'Corte de Barba' with time 30 and price Q25.00. At the bottom of the screen are four icons: a calendar, a list, a person, and a gear.

Nombre	Tiempo (Minutos)	Precio	Accion
Corte de Cabello	30	Q40.00	
Corte de Barba	30	Q25.00	

Figura 41 Listado de servicios

Fuente: Elaboración propia.

El listado de los servicios muestra una tabla donde están los datos que se necesitan para crear los servicios que se darán en la barbería, los campos que muestra son el nombre del servicio, el tiempo que requiere el servicio, el precio establecido para el servicio.

6.7 Agregar servicio



Figura 42 Agregar servicio

Fuente: Elaboración propia.

Para crear un servicio, se deben llenar los campos requeridos por la aplicación, los cuales son el nombre del servicio, el tiempo estimado que tardara el servicio y el precio que se establece al servicio que se está creando, se da clic en el botón guardar para registrar el servicio.

6.8 Catalogo de cortes



Figura 43 Catalogo de cortes

Fuente: Elaboración propia.

Se muestran los cortes de moda, para una visualización de ideas en el cliente mientras llega la hora en la que agendo su cita. y así tenga más claro que corte decide que le realicen, si se dese actualizar los datos o la imagen del estilo de corte se da clic en el botón “Actualizar” y si se desea eliminarlo se da clic en el botón “Eliminar”.

6.9 Agregar cortes

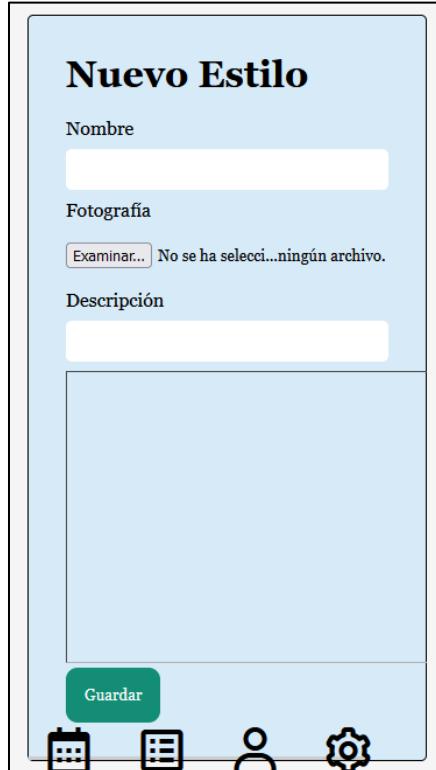


Figura 44 Agregar corte

Fuente: Elaboración propia.

Se deben llenar los campos requeridos por la aplicación para poder ingresar un nuevo corte, los campos son nombre del estilo de corte, se selecciona la imagen haciendo clic en el botón “Examinar” y se selecciona el corte deseado, se ingresa la descripción del corte para tener una idea de donde proviene y cuáles son sus características, se da clic en el botón “guardar” para ingresar el estilo de corte y que se muestre en el catálogo.

6.10 Conexión a base de datos

```

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'dbbarbershop',
        'USER': 'root',
        'PASSWORD': '',
        'HOST': 'localhost',
        'PORT': '3306',
    }
}

```

Este código tiene el propósito de configurar la conexión a la base de datos en un proyecto de django. Establece que motor de base de datos se usara, la ubicación de la base de datos, las credenciales de autenticación y el nombre de la base de datos a la que la aplicación se conectara. Esta configuración es esencial para que django pueda interactuar con la base de datos, almacenar la base de datos y recuperar información de esta.

6.11 Ingreso a la aplicación

```

<html lang="en">
  {% load static %}
  <head>
    <title>Login | Barbershop</title>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link rel="stylesheet" type="text/css" href="{% static 'Login/style.css' %}" />
  </head>

  <body>
    <section>
      <span></span>
    </section>
  </body>

```

```

<span></span>
<div class="signin">
  <div class="content">
    <h2> STUARD BARBERSHOP</h2>
    <div class="form">
      <div class="inputBox">
        <input type="text" required />
        <i>Nombre de Usuario</i>
      </div>
      <div class="inputBox">
        <input type="password" required />
        <i>Contraseña</i>
      </div>
      <div class="links">
        <a href="#">Olvide mi Contraseña</a>
        <a href="#">Registrarse</a>
      </div>
      <div class="inputBox">
        <input type="submit" value="INICIAR SESION" />
      </div>
    </div>
  </div>
</section>
</body>
</html>

```

Este código sirve para crear la página de inicio de sesión de la aplicación web. Permite a los usuarios autenticarse en el sistema utilizando su nombre de usuario y contraseña.

6.12 Menú principal de la aplicación

```

<!DOCTYPE html>
{% load static %}
<html lang="en">

  <head>
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <title>{% block title %}{% endblock %}</title>
    <meta content="width=device-width, initial-scale=1.0, shrink-to-fit=no" name="viewport" />
    <link rel="icon" href="{% static 'icon.ico' %}" type="image/x-icon" />

    <!-- CSS Files -->
    <link href='https://unpkg.com/boxicons@2.1.4/css/boxicons.min.css' rel='stylesheet'>
    <link rel="stylesheet" href="{% static 'Base/styles.css' %}" type="text/css">

  </head>

```

```
<body>
  <header>

    <div class="sidenav" id="mySidenav">
      <a href="{% url 'Inicio' %}"></a>
      <a href="javascript:void(0)" class="closebtn" onclick="closeNav()">&times;</a>

      <button class="dropdown-btn">Citas
      </button>
      <div class="dropdown-container">
        <a href="{% url 'ListadoCita' %}"> Listado Citas </a>
        <a href="{% url 'NuevaCita' %}"> Nueva Citas </a>
      </div>

      <a href="#">Eventos</a>

      <button class="dropdown-btn">Productos
      </button>
      <div class="dropdown-container">
        <a href="{% url 'ListadoProducto' %}">Listado Productos</a>
        <a href="{% url 'NuevoProducto' %}">Agregar Productos</a>
      </div>
      <button class="dropdown-btn">Servicios
      </button>
      <div class="dropdown-container">
        <a href="{% url 'ListadoServicio' %}">Listado Servicios</a>
        <a href="{% url 'NuevoServicio' %}">Agregar Servicios</a>
      </div>
      <button class="dropdown-btn">Estilos
      </button>
      <div class="dropdown-container">
        <a href="{% url 'ListadoEstilo' %}">Listado Estilos</a>
        <a href="{% url 'NuevoEstilo' %}">Agregar Estilos</a>
      </div>
      <a href="#">Reportes</a>
    </div>

    <div id="openBtn">
      <span style="font-size: 30px; cursor: pointer;" onclick="openNav()"><i class='bx bx-menu'></i>
    </div>
  </header>

  <main id="main">
    <div class="contenedor">
      {% block content %}{% endblock %}
    </div>
  </main>
```

```

<footer>
  <div class="menu-footer">
    <a class="menu-icons" href="#"></i></a>
    <a class="menu-icons" href="#"></i></a>
    <a class="menu-icons" href="#"></i></a>
    <a class="menu-icons" href="#"></i></a>
  </div>
</footer>

<!--Js Files-->
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js"></script>
<script src="https://unpkg.com/boxicons@2.1.4/dist/boxicons.js"></script>
<script type="text/javascript" src="#">{% static 'Base/scripts.js' %}"></script>
</body>

</html>

```

Este código se utiliza para crear una barra lateral de navegación en la aplicación web, que ofrece diferentes opciones de menú según el estado de autenticación del usuario, se desarrolló como un menú base que es estático. Proporciona una interfaz de usuario para navegar por las diferentes secciones de la aplicación y realizar específicas, como administrar la aplicación web.

6.12.1 Código javascript

```

let anchoVentana = window.innerWidth
//Funcion que realiza cambios al momento de abrir y cerrar el menu
function openNav() {
  if (anchoVentana < 768) {
    document.getElementById("mySidenav").style.width = "100%";
    document.getElementById("openBtn").style.hidden = true;
    document.getElementById("main").style.hidden = true;
  } else {
    document.getElementById("mySidenav").style.width = "12em";
    document.getElementById("openBtn").style.hidden = true;
    document.getElementById("main").style.marginLeft = "12em";
  }
}

function closeNav() {
  document.getElementById("mySidenav").style.width = "0";
  document.getElementById("openBtn").style.marginLeft = "0";
  document.getElementById("main").style.marginLeft = "0";
  document.getElementById("mySidenav").style.paddingLeft = "0";
}

```

```

//Funcion para los susbmenus
let dropdown = document.getElementsByClassName("dropdown-btn");
let i;

for (i = 0; i < dropdown.length; i++) {
  dropdown[i].addEventListener("click", function () {
    this.classList.toggle("active");
    let dropdownContent = this.nextElementSibling;
    if (dropdownContent.style.display === "block") {
      dropdownContent.style.display = "none";
    } else {
      dropdownContent.style.display = "block";
    }
  });
}

//Funcion de la paginación
$(document).ready(function () {
  $('#data').after('<div id="nav"></div>');
  if (anchoVentana < 768) {
    let rowsShown = 6;
    let rowsTotal = $('#data tbody tr').length;
    let numPages = rowsTotal / rowsShown;
    for (i = 0; i < numPages; i++) {
      let pageNum = i + 1;
      $('#nav').append('<a href="#" rel="' + i + '">' + pageNum + '</a> ');
    }
    $('#data tbody tr').hide();
    $('#data tbody tr').slice(0, rowsShown).show();
    $('#nav a:first').addClass('active');
    $('#nav a').bind('click', function () {

      $('#nav a').removeClass('active');
      $(this).addClass('active');
      let currPage = $(this).attr('rel');
      let startItem = currPage * rowsShown;
      let endItem = startItem + rowsShown;
      $('#data tbody tr').css('opacity', '0.0').hide().slice(startItem, endItem).
        css('display', 'table-row').animate({ opacity: 1 }, 300);
    });
  }
  else{
    let rowsShown = 10;
    let rowsTotal = $('#data tbody tr').length;
    let numPages = rowsTotal / rowsShown;
    for (i = 0; i < numPages; i++) {
      let pageNum = i + 1;
      $('#nav').append('<a href="#" rel="' + i + '">' + pageNum + '</a> ');
    }
    $('#data tbody tr').hide();
    $('#data tbody tr').slice(0, rowsShown).show();
    $('#nav a:first').addClass('active');
    $('#nav a').bind('click', function () {

      $('#nav a').removeClass('active');
      $(this).addClass('active');
      let currPage = $(this).attr('rel');
      let startItem = currPage * rowsShown;
      let endItem = startItem + rowsShown;
      $('#data tbody tr').css('opacity', '0.0').hide().slice(startItem, endItem).
        css('display', 'table-row').animate({ opacity: 1 }, 300);
    });
  }
});

```

```

    }
  else{
    let rowsShown = 10;
    let rowsTotal = $('#data tbody tr').length;
    let numPages = rowsTotal / rowsShown;
    for (i = 0; i < numPages; i++) {
      let pageNum = i + 1;
      $('#nav').append('<a href="#" rel="' + i + '">' + pageNum + '</a> ');
    }
    $('#data tbody tr').hide();
    $('#data tbody tr').slice(0, rowsShown).show();
    $('#nav a:first').addClass('active');
    $('#nav a').bind('click', function () {

      $('#nav a').removeClass('active');
      $(this).addClass('active');
      let currPage = $(this).attr('rel');
      let startItem = currPage * rowsShown;
      let endItem = startItem + rowsShown;
      $('#data tbody tr').css('opacity', '0.0').hide().slice(startItem, endItem).
        css('display', 'table-row').animate({ opacity: 1 }, 300);
    });
  }
};

//Funcion para visualizar imagen
const $seleccionArchivos = document.querySelector("#foto"),
  $imagenPrevisualizacion = document.querySelector("#imagenPrevisualizacion");

// Escuchar cuando cambie
$seleccionArchivos.addEventListener("change", () => {
  // Los archivos seleccionados, pueden ser muchos o uno
  const archivos = $seleccionArchivos.files;
  // Si no hay archivos salimos de la función y quitamos la imagen
  if (!archivos || !archivos.length) {
    $imagenPrevisualizacion.src = "";
    return;
  }
  // Ahora tomamos el primer archivo, el cual vamos a previsualizar
  const primerArchivo = archivos[0];
  // Lo convertimos a un objeto de tipo objectURL
  const objectURL = URL.createObjectURL(primerArchivo);
  // Y a la fuente de la imagen le ponemos el objectURL
  $imagenPrevisualizacion.src = objectURL;
});

// Escuchar cuando cambie
$seleccionArchivos.addEventListener("change", () => {
  // Los archivos seleccionados, pueden ser muchos o uno
  const archivos = $seleccionArchivos.files;
  // Si no hay archivos salimos de la función y quitamos la imagen
  if (!archivos || !archivos.length) {
    $imagenPrevisualizacion.src = "";
    return;
  }
  // Ahora tomamos el primer archivo, el cual vamos a previsualizar
  const primerArchivo = archivos[0];
  // Lo convertimos a un objeto de tipo objectURL
  const objectURL = URL.createObjectURL(primerArchivo);
  // Y a la fuente de la imagen le ponemos el objectURL
  $imagenPrevisualizacion.src = objectURL;
});

```

6.13 Modelos principales

Su función principal es definir los modelos que describen los datos y sus relaciones en tu aplicación web. Estos modelos permiten a Django manejar de manera automática la creación, consulta, actualización y eliminación (CRUD) de los registros en la base de datos.

6.13.1 Modelos de citas

```
from django.db import models

class Cita(models.Model):
    cliente = models.CharField(max_length=250, blank=False, null=False)
    tel = models.CharField(max_length=8, blank=False, null=False, default=0)
    cita = models.DateField(blank=False, null=False)
    hora = models.TimeField(blank=False, null=False)
    servicio = models.CharField(max_length=250, blank=False, null=False)
    barbero = models.CharField(max_length=200, blank=False, null=False)
    fecha = models.DateTimeField(blank=False, null=False, auto_now_add=True)
    estado = models.IntegerField(blank=False, null=False, default=0)

    class Meta:
        ordering = ["id"]

    def __str__(self):
        return self.cliente
```

Este código contiene toda la información relevante sobre una cita en la barbería, como el cliente, el número de teléfono, el tipo de servicio, el barbero, la fecha y hora, y el estado de la cita. Además, con el método `__str__`, cuando se impriman los objetos de este modelo, se mostrará el nombre del cliente.

6.13.2 Modelos de Estilos de cortes

```
from django.db import models

class Estilos(models.Model):
    nombre = models.CharField(max_length=250, blank=False, null=False)
    descripcion = models.CharField(max_length=250, blank=False, null=False)
    foto = models.ImageField(upload_to='estilos/', blank=True, null=True, default='S/F')
    fecha = models.DateField(blank=False, null=False, auto_now_add=True)

    class Meta:
        ordering = ["id"]

    def __str__(self):
        return self.nombre
```

Este modelo se usa para almacenar información sobre los estilos disponibles en la barbería, como el nombre, descripción, imagen y la fecha de creación. La configuración adicional ordena las instancias por id y permite que, cuando se visualice el modelo, se muestre el nombre del estilo.

6.13.3 Modelos de productos

```
from django.db import models

class Productos(models.Model):
    nombre = models.CharField(max_length=250, blank=False, null=False)
    descripcion = models.CharField(max_length=250, blank=False, null=False)
    precio = models.DecimalField(max_digits=4, decimal_places=2, blank=False, null=False, default=0)
    foto = models.ImageField(upload_to='productos/', blank=True, null=True, default='S/F')

    class Meta:
        ordering = ["id"]

    def __str__(self):
        return self.nombre
```

Se utiliza para almacenar información sobre productos en una base de datos, incluyendo su nombre, descripción, precio y una imagen asociada. La configuración adicional asegura que los registros se ordenen por su id, y al mostrar el modelo, se visualiza el nombre del producto.

6.13.4 Modelos de servicios

```
from django.db import models

class Servicios(models.Model):
    nombre = models.CharField(max_length=250, blank=False, null=False)
    tiempo = models.IntegerField(blank=False, null=False, default=0)
    precio = models.DecimalField(max_digits=4, decimal_places=2, blank=False, null=False, default=0)

    class Meta:
        ordering = ["id"]

    def __str__(self):
        return self.nombre
```

Define un modelo servicios, que representa los servicios que una barbería puede ofrecer como cortes, se definen con 3 datos de los cuales son nombre, tiempo estimado y precio, estos se definen en la programación y se crean en la base de datos una vez se hace la migración de datos.

6.14Urls principales

Es un componente fundamental encargado de gestionar el enrutamiento de las solicitudes *requests* dentro de una aplicación web. Este es el archivo donde se definen las URLs que van a estar asociadas con vistas específicas *views* o funciones de la aplicación.

6.14.1Urls de citas

```
from django.urls import path,include
from Citas import views

urlpatterns = [
    path('nueva-cita/',views.nueva,name="NuevaCita"),
    path('listado-cita/',views.listado,name="ListadoCita"),
    path('actualizar-cita/<int:id>',views.actualizar,name="UpdateCita"),
    path('eliminar-cita/<int:id>',views.eliminar,name="DeleteCita"),
]
```

Este parte del código define un conjunto de rutas URL para la aplicación Citas, que están relacionadas con la gestión de citas dentro del proyecto web. El archivo urls.py actúa como el mapa que conecta las URLs ingresadas por el usuario con las vistas correspondientes que procesarán esas solicitudes.

6.14.2Urls de estilos de cortes

```
from django.urls import path,include
from Estilos import views

urlpatterns = [
    path('nuevo-estilo/',views.nuevo,name="NuevoEstilo"),
    path('listado-estilo/',views.listado,name="ListadoEstilo"),
    path('actualizar-estilo/<int:id>',views.actualizar,name="UpdateEstilo"),
    path('eliminar-estilo/<int:id>',views.eliminar,name="DeleteEstilo"),
]
```

Este código define las rutas URLs de la aplicación estilos para gestionar la creación, listado, actualización y eliminación de estilos dentro de la aplicación web. Utiliza la función path para asociar cada URL con una vista correspondiente ubicada en el archivo views.py de la aplicación. Las rutas incluyen: crear un nuevo estilo (nuevo-estilo/), listar estilos (listado-estilo/), actualizar un estilo (actualizar-estilo/<int:id>), y eliminar un estilo (eliminar-estilo/<int:id>), donde el parámetro <int:id> representa el identificador de cada estilo en las operaciones de actualización y eliminación.

6.14.3Urls de productos

```
from django.urls import path,include
from Productos import views

urlpatterns = [
    path('nuevo Producto/',views.nuevo,name="NuevoProducto"),
    path('listado Producto/',views.listado,name="ListadoProducto"),
    path('actualizar Producto/<int:id>',views.actualizar,name="UpdateProducto"),
    path('eliminar Producto/<int:id>',views.eliminar,name="DeleteProducto"),
]
```

Esta sección de código define las rutas URLs para manejar las operaciones de crear, leer, actualizar y eliminar de productos en una aplicación web. Usa el módulo path para mapear las URLs a las funciones correspondientes en el archivo views.py de la aplicación productos.

6.14.4Urls de servicios

```
from django.urls import path,include
from Servicios import views

urlpatterns = [
    path('nuevo Servicio/',views.nuevo,name="NuevoServicio"),
    path('listado Servicio/',views.listado,name="ListadoServicio"),
    path('actualizar Servicio/<int:id>',views.actualizar,name="UpdateServicio"),
    path('eliminar Servicio/<int:id>',views.eliminar,name="DeleteServicio"),
]
```

Esta sección de código define las rutas URLs para gestionar los servicios en la aplicación. Cada ruta está vinculada a una vista dentro del archivo views.py de la aplicación servicios, lo que permite realizar operaciones de crear, leer, actualizar y eliminar sobre los servicios.

6.15 Vistas principales

Es donde se definen las vistas de la aplicación. Las vistas son funciones o clases que reciben solicitudes HTTP y devuelven respuestas HTTP.

6.15.1 Vista de citas

```
from django.shortcuts import render,redirect
from Citas.models import Cita
from Estilos.models import Estilos
from django.contrib import messages

def nueva(request):
    if request.method == "POST":
        c = Cita()
        c.cliente = request.POST['cliente']
        c.tel = request.POST['tel']
        c.cita = request.POST['cita']
        c.hora = request.POST['hora']
        c.servicio = request.POST['servicio']
        c.barbero = request.POST['barbero']
        c.save()
        messages.success(request,f'Cita Agregada para el dia {c.cita} a las {c.hora} hrs Espere Confirmacion!')
        return redirect('NuevaCita')

    return render(request,'Citas/nueva.html')

def listado(request):
    citas = Citas.objects.all()
    return render(request,'Citas/listado.html',{'citas':citas})

def actualizar(request,id):
    c = Citas.objects.get(id=id)
    if request.method == "POST":
        Citas.objects.filter(id=id).update(cliente=request.POST['cliente'],telefono=request.POST['tel'],cita=request.POST['cita'])
        messages.success(request,f'Citas {c.cliente} Actualizado!')
        return redirect('ListadoCitas')

    return render(request,'Citas/actualizar.html',{'c':c})

def eliminar(request,id):
    Citas.objects.filter(id=id).delete()
    messages.success(request,f'Cita Eliminada!')
    return redirect('ListadoCitas')
```

Este código define cuatro funciones para manejar las citas. La función nueva permite crear una nueva cita a través de un formulario POST, almacenando los datos en el modelo Cita y mostrando un mensaje de éxito. La función listado obtiene todas las citas almacenadas y las pasa al template listado.html para su visualización. La función actualizar permite modificar una cita existente, identificándola por su id y actualizando los campos necesarios, mostrando un mensaje cuando la actualización es exitosa y la función eliminar borra una cita específica por su id y redirige a la lista de citas con un mensaje de eliminación exitosa.

6.15.2 Vista de estilos de cortes

```
from django.shortcuts import render,redirect
from Estilos.models import Estilos
from django.contrib import messages

def nuevo(request):
    if request.method == "POST":
        e = Estilos()
        e.nombre = request.POST['nombre']
        e.descripcion = request.POST['descripcion']
        e.foto = request.FILES['foto']
        e.save()
        messages.success(request,f'Estilo {e.nombre} Ingresado!')
        return redirect('NuevoEstilo')

    return render(request,'Estilos/nuevo.html')

def listado(request):
    cortes = Estilos.objects.all()

    return render(request,'Estilos/listado.html',{'cortes':cortes})

def actualizar(request,id):
    e = Estilos.objects.get(id=id)

    if request.method == "POST":
        Estilos.objects.filter(id=id).update(nombre=request.POST['nombre'],
                                             descripcion=request.POST['descripcion'],foto=f"estilos/{request.FILES['foto']}")
        messages.success(request,f'Estilo {e.nombre} Actualizado!')
        return redirect('ListadoEstilo')

    return render(request,'Estilos/actualizar.html',{'e':e})

def eliminar(request,id):
    Estilos.objects.filter(id=id).delete()
    messages.success(request,f'Estilo Eliminado!')
    return redirect('ListadoEstilo')
```

Esta sección de código define cuatro funciones en para gestionar estilos mediante operaciones CRUD: nuevo crea un nuevo estilo al recibir datos de un formulario y redirige a la página de creación, listado recupera y muestra todos los estilos existentes en la plantilla correspondiente, actualizar permite editar un estilo específico, actualizando sus datos si se envía un formulario, y redirigiendo a la lista de estilos, eliminar elimina un estilo seleccionado y redirige a la lista, también mostrando un mensaje de confirmación.

6.15.3 Vista de productos

```
from django.shortcuts import render,redirect
from Productos.models import Productos
from django.contrib import messages

def nuevo(request):
    if request.method == "POST":
        p = Productos()
        p.nombre = request.POST['nombre']
        p.descripcion = request.POST['descripcion']
        p.precio = request.POST['precio']
        p.foto = request.FILES['foto']
        p.save()
        messages.success(request,f'Producto {p.nombre} Ingresado!')
        return redirect('NuevoProducto')

    return render(request,'Productos/nuevo.html')

def listado(request):
    producto = Productos.objects.all()
    return render(request,'Productos/listado.html',{'productos':producto})

def actualizar(request,id):
    p = Productos.objects.get(id=id)

    if request.method == "POST":
        Productos.objects.filter(id=id).update(nombre=request.POST['nombre'],
                                                descripcion=request.POST['descripcion'],foto=f"productos/{request.FILES['foto']}")  

        messages.success(request,f'Producto {p.nombre} Actualizado!')
        return redirect('ListadoProducto')

    return render(request,'Productos/actualizar.html',{'p':p})

def eliminar(request,id):
    Productos.objects.filter(id=id).delete()
    messages.success(request,f'Producto Eliminado!')
    return redirect('ListadoProducto')
```

Este código define funciones para manejar las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) de productos en una aplicación. La función nuevo gestiona la creación de un nuevo producto, guardando sus datos (nombre, descripción, precio y foto) al recibir una solicitud POST y redirigiendo a la página de creación con un mensaje de éxito. La función listado recupera todos los productos y los envía a la plantilla correspondiente para su visualización. La función actualizar permite editar un producto específico, actualizando su información si se envía un formulario POST, y luego redirige a la lista de productos. Por último, la función eliminar elimina un producto según su id y muestra un mensaje de confirmación antes de redirigir a la lista de productos. En conjunto, estas funciones facilitan la gestión de productos en la aplicación.

6.15.4 Vista de servicios

```

from django.shortcuts import render,redirect
from Servicios.models import Servicios
from django.contrib import messages

def nuevo(request):
    if request.method == "POST":
        s = Servicios()
        s.nombre = request.POST['nombre']
        s.tiempo = request.POST['tiempo']
        s.precio = request.POST['precio']
        s.save()
        messages.success(request,f'Servicio {s.nombre} Ingresado!')
        return redirect('NuevoServicio')

    return render(request,'Servicios/nuevo.html')

def listado(request):
    servicios = Servicios.objects.all()
    return render(request,'Servicios/listado.html',{'servicios':servicios})

def actualizar(request,id):
    s = Servicios.objects.get(id=id)

    if request.method == "POST":
        Servicios.objects.filter(id=id).update(nombre=request.POST['nombre'],
                                                tiempo=request.POST['tiempo'],precio=request.POST['precio'])
        messages.success(request,f'Servicio {s.nombre} Actualizado!')
        return redirect('ListadoServicio')

    return render(request,'Servicios/actualizar.html',{'s':s})

def eliminar(request,id):
    Servicios.objects.filter(id=id).delete()
    messages.success(request,f'Servicio Eliminado!')
    return redirect('ListadoServicio')

```

Este fragmento de código implementa funciones para gestionar servicios mediante operaciones de crear, leer, actualizar, eliminar. La función nuevo permite crear un nuevo servicio al recibir datos a través de un formulario POST, guarda el nombre, tiempo y precio del servicio, muestra un mensaje de éxito y redirige a la página de creación. La función listado obtiene todos los servicios y los envía a la plantilla correspondiente para su visualización. La función actualizar busca un servicio específico por su id y, si se envían nuevos datos mediante POST, actualiza el servicio y redirige a la lista de servicios mostrando un mensaje de éxito. La función eliminar elimina el servicio indicado por el id y redirige a la lista de servicios, también mostrando un mensaje de confirmación.

CONCLUSION

1. El módulo de agendación de citas facilita la gestión y planificación de horarios tanto para los clientes como para el personal de la barbería. Al permitir que los usuarios seleccionen sus preferencias de fecha y hora de forma directa, se mejora la precisión de la calendarización y se reduce la posibilidad de errores o conflictos de horarios. Esta automatización no solo ahorra tiempo al personal, sino que también brinda comodidad a los clientes, quienes pueden agendar citas a su conveniencia, lo que resulta en una experiencia más organizada y eficiente para ambas partes.
2. Permitir la selección de barberos y tipos de servicios en el módulo de agendación de citas brinda a los clientes una experiencia más personalizada, mientras que optimiza la asignación de recursos dentro de la barbería. Este sistema permite calcular el tiempo requerido para cada servicio, mejorando la planificación y asegurando que el cliente reciba el servicio adecuado de acuerdo con sus preferencias. De este modo, la barbería logra un uso más eficiente de su equipo y tiempo, mientras mejora la satisfacción del cliente al ofrecer más control sobre la experiencia de servicio.
3. La implementación de notificaciones de recordatorio para citas contribuye significativamente a la optimización del tiempo en la barbería, al reducir el riesgo de ausencias o cancelaciones tardías. Este sistema automatizado no solo ayuda a los clientes a mantenerse organizados, sino que también asegura que la barbería pueda gestionar su flujo de trabajo de manera más eficiente. Minimizar las ausencias mejora la productividad y permite un mejor uso de los recursos disponibles, lo que a su vez maximiza la rentabilidad y la satisfacción tanto de los clientes como del personal.

RECOMENDACIONES

1. Se recomienda implementar un sistema que permita a los clientes acceder a su historial de citas y servicios realizados. Esto no solo mejora la experiencia del usuario al facilitar la repetición de servicios anteriores, sino que también permite a los barberos ofrecer un servicio más personalizado basado en preferencias pasadas. Además, esto puede fomentar la fidelización al recordar a los clientes sobre los servicios que más les gustaron.
2. Es crucial mantener una agenda en tiempo real que refleje la disponibilidad de barberos y servicios, y que se actualice de inmediato cuando se realicen nuevas reservas. Esto reduce el riesgo de duplicaciones o conflictos en las citas y proporciona una experiencia de usuario más fluida. Además, un control eficiente de la disponibilidad ayuda al personal de la barbería a gestionar mejor su tiempo y recursos.
3. Se recomienda implementar un sistema de valoración y comentarios después de cada cita. Esto permite a los clientes calificar a los barberos y servicios, brindando a la barbería una fuente valiosa de retroalimentación. Con esta información, se pueden identificar áreas de mejora en el servicio y destacar a los barberos mejor valorados, lo que ayuda a mantener un alto nivel de calidad y satisfacción del cliente.

GLOSARIO

Agendación: Proceso de reservar una cita para un servicio en la barbería.

Barbero: Profesional encargado de realizar cortes de cabello y otros servicios en la barbería.

Catálogo de cortes: Lista de estilos de corte de cabello disponibles para los clientes.

Servicio: Actividad ofrecida por la barbería, como cortes de cabello, afeitados o tratamientos.

Cliente: Persona que utiliza la aplicación para agendar una cita y recibir servicios en la barbería.

Cita: Reservación de un servicio en una fecha y hora específicas.

Disponibilidad: Horarios en los que un barbero o servicio está disponible para agendarse.

Recordatorio: Notificación enviada al cliente para recordar una cita próxima.

Cancelación: Acción de anular una cita previamente agendada.

Reprogramación: Cambio de la fecha o la hora de una cita ya establecida.

Notificación push: Mensaje que se envía directamente a los dispositivos móviles para recordar o actualizar información sobre la cita.

Pasarela de pago: Sistema que permite realizar pagos en línea dentro de la aplicación.

Feedback: Comentarios o valoraciones que los clientes dan sobre el servicio recibido.

Optimización móvil: Adaptación de la aplicación para que funcione correctamente en dispositivos móviles.

Perfil de usuario: Información personal y preferencias del cliente en la aplicación.

Historial de citas: Registro de todas las citas anteriores de un cliente en la barbería.

Evento: Actividad especial organizada por la barbería, como promociones o lanzamientos de productos.

Producto: Artículos vendidos en la barbería, como geles, champús o cremas.

Promoción: Descuento o incentivo ofrecido a los clientes para fomentar el uso de la aplicación o la compra de servicios.

Personalización: Ajustes en la experiencia del cliente basados en sus preferencias o historial de uso.

Seguridad: Medidas implementadas para proteger los datos personales y transacciones del cliente.

Interfaz de usuario (UI): Diseño y estructura visual de la aplicación, que facilita la interacción del cliente.

Experiencia de usuario (UX): Calidad de la interacción del cliente con la aplicación.

Login: Proceso de acceso a la aplicación mediante credenciales, como usuario y contraseña.

Agenda en tiempo real: Sistema que actualiza automáticamente la disponibilidad de citas y barberos.

REFERENCIAS

- Academy, D. (s.f.). *Diadema Academy*. Diadema Academy.
<https://diadema.academy/es/convertirse-en-barbero-el-arte-de-cuidar-la-barba-y-el-cabello-para-hombres/#:~:text=Una%20barber%C3%A9s%20un%20sal%C3%B3n,dedicado%20exclusivamente%20a%20los%20hombres.>
- Agendapro. (s.f.). *Agendapro*. <https://blog.agendapro.com/>
- Aura, M. (s.f.). *Calameo*. Calameo. <https://www.calameo.com/books/00493014106d039702883>
- Booksy. (2,024). *Booksy*. <https://booksy.com>
- Christudas, B. (26 de Junio de 2,019). *Springer Link*. Springer Link.
https://link.springer.com/chapter/10.1007/978-1-4842-4501-9_27
- Concepto, E. (2.024). *Enciclopedia Concepto*.
- Consulting, S. (27 de Septiembre de 2,022). *Linkedin*. <https://es.linkedin.com/pulse/riesgos-y-problemas-en-los-proyectos-cu%C3%A1l-es-la-diferencia-#:~:text=Un%20problema%20es%20un%20evento,en%20los%20objetivos%20del%20proyecto.>
- CUAED, UNAM. (2,017). *Cuaed Unam*. Cuaed Unam. https://repositorio-ropa.cuaeed.unam.mx/repositorio/moodle/pluginfile.php/2655/mod_resource/content/1/UAPA-Lenguajes-Programacion/index.html
- Dimes, T. (2,015). En D. Troy, *Programacion C#* (pág. 27). Babelcube.
<https://books.google.com.gt/books?id=LID2CQAAQBAJ&pg=PT9&lpg=PT9&dq=c%23&source=bl&ots=OWXWfm-mF1&sig=ACfU3U11KOL61llqBqEk0kEoTY7lXFyTRQ&hl=es-419&sa=X&ved=2ahUKEwiXpdyk3uKFAXeRzABHX-yDbY4FBDoAXoECAQQAw#v=onepage&q=c%23&f=false>
- Dubois, P. (2,003). *MySQL*. MySQL. <http://dev.mysql.com/doc/>
- Engagebay. (s.f.). *engagebay*. engagebay. <https://www.engagebay.com/es/glossary/appointment-scheduling>
- Euroinnova. (2,024). *Euroinnova*. <https://www.euroinnova.edu.es>
- GoodFellas. (2,015). *GoodFellas*. GoodFellas. <https://goodfellaspomade.com.mx/que-servicios-debe-tener-una-barberia>

Hade. (27 de Febrero de 2,023). *Hadepeluqueria*. Hadepeluqueria.

<https://www.hadepeluqueria.com/tipos-de-peluquerias/>

IBM Corporation. (09 de 03 de 2,021). *IBM*.

<https://www.ibm.com/docs/es/psfoa/1.0.0?topic=information-user-roles-hardware-administrators>

Infoguía. (21 de Octubre de 2,019). *Infoguia*. Infoguia. <https://infoguia.com/infotip.asp?t=que-es-barberia&a=1843>

IONOS Inc. (24 de 07 de 2,020). *IONOS Digital Guide*. <https://www.ionos.com/es-us/digitalguide>

Javier. (17 de Agosto de 2,023). *Formadoresit*. Formadoresit. <https://formadoresit.es/net-maui-que-es-y-que-ventajas-ofrece/>

Microsoft. (2,024). *Microsoft Learn*. Microsoft Learn. https://learn.microsoft.com/es-mx/dotnet/csharp/?WT.mc_id=dotnet-35129-website

Oracle. (s.f.). *Oracle*. Oracle. <https://www.oracle.com/mx/database/what-is-database/>

Paul, D. (2,003). *MySQL*. MySQL. <http://dev.mysql.com/doc/>