UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS
FACULTAD DE TECNOLOGÍAS INTERACTIVAS

MÓDULO DE RESERVACIÓN PARA EL SISTEMA INTEGRAL DE GESTIÓN DE ALIMENTACIÓN XABAL SIGA 2.0

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor: Alejandro Labaut Caro

Tutores: M.Sc. PA Tatiana Leyva Estrada

Ing. Dayron Alejandro Medero Ramos

La Habana, 2024

Todo parece imposible hasta que se hace.
Nelson Mandela

Dedicatoria

Dedico esta tesis con profundo agradecimiento a mi familia, cuyo apoyo incondicional ha sido mi mayor fortaleza. En especial, quiero expresar mi más sincera gratitud a mi madre y a mi padre, quienes con su amor, esfuerzo y ejemplo han sido mi fuente constante de inspiración.

Agradecimientos

En primer lugar, agradezco a Dios por darme la fortaleza, salud y sabiduría necesarias para alcanzar este logro.

Quiero expresar mi gratitud a mis padres, cuyo apoyo incondicional ha sido el pilar fundamental en mi vida. Su amor, sacrificio y enseñanzas han sido una inspiración constante, dándome la motivación para superar cada obstáculo y perseguir mis metas.

A mi familia, quienes con su cariño y comprensión me han brindado el apoyo emocional que tanto he necesitado a lo largo de este camino. Sus palabras de ánimo y confianza en mis capacidades han sido esenciales para este logro.

Finalmente, quiero dedicar un reconocimiento especial a mis amigos y compañeros, quienes han compartido conmigo momentos de esfuerzo, aprendizaje y también de alegría. Su compañía ha hecho que este viaje sea más llevadero y significativo.

A todos los que de una forma u otra formaron parte del camino recorrido, gracias por ser parte de este sueño hecho realidad.

Declaración de autoría

Declaro ser autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales sobre esta, con carácter exclusivo.

Para que así conste firmamos la presente a los 29 días del mes de noviembre del año 2024.

Alejandro Labaut Caro
Autor

M.Sc. PA Tatiana Leyva Estrada
Tutora

Ing. Dayron Alejandro Medero Ramos
Tutor

Resumen

Este trabajo tiene como objetivo desarrollar un Módulo de Reservación para XABAL Sistema Integral de Gestión de Alimentación Versión 2.0 (XABAL SIGA 2.0), utilizado en la Universidad de las Ciencias Informáticas (UCI). El sistema actual presenta varias limitaciones, como la falta de flexibilidad para personalización, interdependencia de componentes y el uso de tecnologías obsoletas, lo que afecta su capacidad para adaptarse a diferentes entornos y restringe su comercialización. Para abordar estas limitaciones, se propone el rediseño del módulo mediante una arquitectura Software como Servicio (SaaS, por sus siglas en inglés), utilizando Django Rest Framework y React. El desarrollo se llevó a cabo bajo la metodología ágil Programación Extrema (XP, por sus siglas en inglés), permitiendo entregas iterativas y retroalimentación continua, garantizando un sistema adaptable y escalable.

Palabras clave: Django Rest Framework, Gestión de alimentación, Módulo de reservación, React, Software como Servicio (SaaS), XP, XABAL SIGA 2.0.

Índice general

Índice de figuras

Índice de tablas

Introducción

Las reservaciones han sido una práctica fundamental en la organización humana desde tiempos antiguos, reflejando la necesidad de planificar, gestionar recursos y anticipar demandas. Originalmente surgieron como mecanismos simples para coordinar espacios, servicios y eventos, evolucionando con el tiempo desde métodos rudimentarios hasta sistemas altamente sofisticados e informatizados.

En el ámbito específico de los alimentos, la práctica de realizar reservaciones tiene raíces profundas en las primeras civilizaciones. Mesopotamia y Egipto ya desarrollaban sistemas elementales para administrar la preparación y distribución de alimentos en grandes eventos religiosos y sociales. Los administradores de templos y palacios utilizaban registros básicos para calcular el consumo, garantizando que los recursos fueran suficientes para toda la comunidad (Mehdawy y Hussein, 2010).

Durante la Edad Media, las ferias, posadas y mercados locales comenzaron a implementar formas más estructuradas de reservación. Las tabernas mantenían inventarios rudimentarios para asegurar el suministro a viajeros y comerciantes, introduciendo de manera informal un sistema de reservas de alimentos y espacios (Braudel, 1982).

Con el auge de la urbanización durante la Revolución Industrial, las prácticas de gestión en los restaurantes evolucionaron, incluyendo por primera vez registros de pedidos anticipados y sistemas rudimentarios de reservas para satisfacer la creciente demanda de los nuevos centros urbanos (Giraldo, 2023).

El avance tecnológico en la gestión de reservas ha revolucionado el sector alimenticio, permitiendo un control más preciso y en tiempo real de las existencias y reservas. Esto ha facilitado una planificación eficiente de la demanda y mejorado la experiencia del cliente gracias a soluciones automatizadas y móviles (Geldres Trujillo, 2015).

En la actualidad, las tecnologías de inteligencia artificial y análisis de datos han llevado las reservaciones de alimentos a un nivel de sofisticación nunca antes imaginado. Los sistemas actuales no solo prevén patrones de consumo, sino que pueden ajustar ofertas, hacer recomendaciones personalizadas y contribuir a una gestión más sostenible y eficiente del sector alimentario. Esta evolución refleja cómo las reservaciones de alimentos han pasado de ser un simple mecanismo de organización a convertirse en una herramienta estra-

tégica que integra tecnología, datos y experiencia del cliente.

En este sentido, el sistema XABAL Sistema Integral de Gestión de Alimentación (XABAL SIGA), desarrollado en la UCI, se destaca como una herramienta integral para la gestión de la alimentación, la cual integra un módulo para la gestión de reservaciones. XABAL SIGA fue diseñado bajo una arquitectura Modelo-Vista-Controlador (MVC) con el objetivo de cubrir las necesidades de gestión alimentaria en la UCI. Sin embargo, pese a sus avances y funcionalidad, este sistema presenta varias limitaciones que han evidenciado la necesidad de una modernización y adaptación para responder a las demandas actuales.

Entre las principales limitaciones de XABAL SIGA se encuentran:

- **Rigidez en la personalización:** El sistema no se adapta fácilmente a las necesidades específicas de diferentes centros de gestión alimentaria, lo que obliga a implementar todos los módulos sin importar su relevancia en cada contexto.
- **Interdependencia de componentes:** Los distintos componentes del sistema están estrechamente integrados, lo que dificulta su modificación individual y aumenta el riesgo de errores durante el mantenimiento.
- **Deuda tecnológica:** XABAL SIGA fue desarrollado con versiones obsoletas de tecnologías como PHP y Symfony, lo que implica una falta de actualizaciones de seguridad y vulnerabilidad frente a amenazas cibernéticas.
- **Limitaciones comerciales:** Las tecnologías desactualizadas y la rigidez del sistema limitan su viabilidad comercial, restringiendo su implementación en otros entornos y reduciendo su potencial para generar ingresos.

Estos desafíos resaltan la necesidad de una renovación integral del sistema, enfocada en adoptar una arquitectura más flexible, tecnologías actualizadas y un diseño que facilite tanto la personalización como el mantenimiento. La modernización de XABAL SIGA busca potenciar su aplicabilidad en distintos contextos y mejorar su valor comercial.

Frente a esta problemática, la investigación se centra en responder: **¿Cómo convertir XABAL SIGA 2.0 en un sistema flexible y adaptable?**

Objetivo General:
Desarrollar el Módulo de Reservación para XABAL SIGA 2.0.

Objeto de Estudio:
Reservaciones en Sistemas Informáticos.

Campo de Acción:
Módulo de Reservación para XABAL SIGA 2.0.

Tareas de Investigación:

1. Elaborar el marco teórico mediante un estudio del estado del arte.
2. Levantar requisitos funcionales y no funcionales del módulo a desarrollar.
3. Planificar y diseñar la solución propuesta.
4. Desarrollar el módulo de reservación.
5. Realizar pruebas al módulo desarrollado.

Métodos Científicos:
Teóricos:

- Modelación: Para el diseño de la solución y el modelo de datos.
- Análisis sintético: Para sintetizar conocimientos sobre gestión de alimentación y sistemas afines.

Empíricos:

- Observación: Para comprender el funcionamiento del sistema actual y los procesos de gestión.

Estructura de la Investigación:
Capítulo 1: FUNDAMENTOS Y REFERENTES TEÓRICO-METODOLÓGICOS SOBRE EL OBJETO DE ESTUDIO

- Estado del arte
- Diagnóstico del módulo actual
- Análisis de soluciones homologas
- Selección del entorno y metodología de desarrollo

Capítulo 2: DISEÑO DE LA SOLUCIÓN PROPUESTA AL PROBLEMA CIENTÍFICO

- Propuesta de solución
- Historias de usuario
- Planificación de entregas
- Diseño del sistema

Capítulo 3: DESARROLLO Y VERIFICACIÓN DEL MÓDULO

- Descripción de iteraciones
- Tareas de ingeniería por historia de usuario
- Pruebas unitarias, de integración y aceptación

CAPÍTULO 1

FUNDAMENTOS Y REFERENTES TEÓRICO-METODOLÓGICOS SOBRE EL OBJETO DE ESTUDIO

En este capítulo se establece el marco conceptual y metodológico que sustenta la investigación sobre el desarrollo del Módulo de Reservación para XABAL SIGA 2.0. El capítulo tiene como objetivo proporcionar una visión de la evolución y la importancia de los sistemas de reservación en el ámbito de la gestión alimentaria, tanto a nivel internacional como en el contexto cubano. Además, se abordan los fundamentos teóricos y metodológicos que guiarán el desarrollo del módulo propuesto, así como un análisis detallado del estado actual del objeto de estudio, identificando las principales limitaciones y áreas de mejora. La estructura del capítulo se organiza en secciones que exploran, respectivamente, la gestión de reservas en sistemas informáticos, los enfoques teórico-metodológicos relevantes, el diagnóstico del sistema actual, soluciones homólogas, las tecnologías seleccionadas para la investigación y la metodología de desarrollo.

1.1. Reservaciones en sistemas informáticos

Existen diversas perspectivas teóricas relacionadas con el término “sistemas informáticos“. Para entender su funcionamiento, es útil descomponerlo en varios conceptos. Según el Diccionario de la Lengua Española, un sistema se define como “Conjunto de cosas que relacionadas entre sí ordenadamente contribuyen a determinado objeto“ (ASALE y RAE, 2024). Aunque esta definición es válida, resulta bastante básica. Desde la perspectiva de las ciencias informáticas, un sistema informático se entiende como el conjunto de componentes que procesan, almacenan y transmiten información (Laudon, 2012).

La gestión de reservaciones en sistemas informáticos ha evolucionado significativamente en las últimas décadas, convirtiéndose en un componente esencial en diversos sectores, permitiendo a las empresas optimizar sus recursos, mejorar la experiencia del cliente y aumentar la eficiencia operativa.

En muchos sectores, la gestión de reservas juega un papel fundamental. Facilita la planificación de recursos,

la gestión de inventarios y la coordinación entre diferentes áreas del negocio. Además, permite a los clientes realizar reservas de manera ágil y personalizada, adaptándose a las tendencias actuales de consumo y preferencia.

A nivel internacional, los sistemas de reservación han sido adoptados por una amplia variedad de industrias, y su evolución ha estado marcada por la incorporación de tecnologías emergentes como la computación en la nube y la inteligencia artificial. La computación en la nube, en particular, ha permitido el desarrollo de sistemas de reservación altamente escalables y accesibles desde cualquier lugar, facilitando su adopción global (Marinescu; Paya y Morrison, 2017). La gestión de reservaciones ahora integra algoritmos de inteligencia artificial para predecir la demanda y optimizar la asignación de recursos, mejorando la eficiencia operativa y la satisfacción del cliente.

En Cuba, el desarrollo de sistemas de reservación ha seguido un camino similar, aunque adaptado a las particularidades del contexto nacional. La UCI ha sido pionera en el desarrollo de sistemas informáticos en el país, incluyendo la gestión de reservaciones como parte de soluciones más amplias, como es el caso de XABAL SIGA. Sin embargo, las limitaciones tecnológicas y de infraestructura han influido en la adopción y evolución de estos sistemas, subrayando la necesidad de innovación y modernización.

1.2. Sistemas de Gestión de Alimentación

El Sistema de Gestión Alimentaria (SGA) en un servicio de alimentación se refiere al conjunto de procedimientos y estrategias utilizados para organizar, dirigir y evaluar todos los procesos involucrados en la prestación de servicios alimentarios. Su objetivo principal es asegurar que las necesidades y expectativas de los comensales sean satisfechas mediante la entrega de alimentos de la más alta calidad, manteniendo estándares rigurosos en la seguridad alimentaria, la eficiencia operativa y la gestión de recursos humanos y materiales (Leal, 2022).

En su estructura, los sistemas de gestión de alimentación suelen integrar módulos especializados que abarcan desde la gestión de inventarios y la planificación de menús, hasta la reserva de espacios y el control de costos. La capacidad de adaptación y personalización de estos sistemas permite a las instituciones ajustarlos a sus necesidades específicas, garantizando una gestión óptima de recursos y una respuesta ágil a las demandas fluctuantes del entorno.

Además, estos sistemas incorporan herramientas avanzadas para la recopilación y análisis de datos, facilitando la generación de informes detallados sobre el rendimiento operativo y la satisfacción del cliente. Esta capacidad analítica no solo apoya la toma de decisiones estratégicas, sino que también promueve la mejora

continua de los procesos y servicios ofrecidos.

Los sistemas de gestión de alimentación son esenciales en áreas donde la eficiencia, la calidad del servicio y la optimización de recursos son prioritarios. Su implementación no solo impulsa la productividad organizacional, sino que también eleva los estándares de atención al cliente, asegurando una experiencia integral y satisfactoria en la gestión de servicios alimenticios. La UCI ha implementado una solución informática avanzada para la gestión de alimentación conocida como XABAL SIGA. Este sistema ofrece una plataforma integral diseñada para optimizar todos los aspectos relacionados con la provisión de alimentos y servicios asociados en la institución.

1.2.1. Sistema de Gestión de Alimentación XABAL SIGA

El sistema de gestión de alimentación XABAL SIGA fue desarrollado por la Dirección de Informatización de la UCI en Cuba. Esta solución informática avanzada está diseñada específicamente para administrar de manera eficiente todos los aspectos relacionados con la provisión de alimentos dentro de la universidad, asegurando un servicio óptimo y organizado para estudiantes, profesores y visitantes.

Descripción de los Módulos:

1. **Abastecimiento:** El módulo de abastecimiento permite una visualización clara de los productos en almacén a través de gráficos detallados. Facilita la gestión de platos, menús y productos, así como la planificación anticipada para fechas y períodos especiales mediante imágenes explicativas que mejoran la comprensión del usuario.
2. **Reservación:** El módulo de reservación ofrece la posibilidad de programar reservas para acceder al servicio de alimentación de la universidad. A través de una interfaz intuitiva, los usuarios pueden consultar el menú del día y realizar reservas para personal de la universidad, familiares, asociados y terceros. Además, proporciona reportes detallados de platos y menús por fechas, asegurando una planificación eficiente.
3. **Distribución:** Este módulo facilita la organización de comensales mediante gráficas que muestran las reservaciones por días y la distribución de tarjetas en los diferentes centros de alimentación definidos por la universidad.
4. **Cajero:** El módulo de cajero automatiza el proceso de acceso a eventos dentro de la universidad. Permite el escaneo de tarjetas para controlar el acceso del personal, mostrando en pantalla información detallada sobre la persona que ingresa al comedor, el evento en curso, la fecha y la hora.
5. **Facturación:** Ofrece una visualización gráfica y detallada del dinero facturado por la universidad en cada evento de desayuno, almuerzo y comida, facilitando el control financiero a través de listados con los importes correspondientes por fecha.
6. **Reportes:** El módulo de reportes proporciona datos estadísticos clave sobre todo el proceso de gestión de alimentación de la universidad. Permite generar informes generales y específicos mediante la

introducción de fechas y diferentes filtros, apoyando así la toma de decisiones informadas.

7. **Administración:** Este módulo permite la gestión de roles, permisos, usuarios y grupos de roles dentro del sistema, garantizando un control adecuado y seguro de los accesos y funciones.
8. **Configuración:** El módulo de configuración establece los principios fundamentales para el funcionamiento del sistema, asegurando una integración eficiente de todos los componentes y adaptándose a las necesidades específicas de la universidad.

El sistema XABAL SIGA no solo optimiza la gestión de alimentación dentro de la UCI, sino que también mejora la experiencia de usuarios y administradores mediante herramientas avanzadas de reservación, control de acceso y análisis estadístico, que ayudan a mantener un servicio alimentario eficaz y de calidad.

1.3. Análisis del Estado Actual del Módulo de Reservación de XABAL SIGA

El Módulo de Reservación de XABAL SIGA es un componente crucial para la gestión eficiente del servicio de alimentación en la UCI, este permite a los usuarios realizar reservas anticipadas para acceder al servicio de alimentación de la universidad, lo cual es esencial para la planificación eficaz de recursos. Los usuarios pueden reservar menús específicos según su categoría, que incluye personal universitario, familiares, asociados y terceros, asegurando que las necesidades de cada grupo sean adecuadamente atendidas. Además, el módulo facilita la generación de reportes detallados sobre los platos y menús disponibles para cada fecha, las reservaciones realizadas, distribuciones del usuario, accesos, facturaciones y tarjetas, lo que contribuye a una gestión precisa y organizada del servicio de alimentación.

Sin embargo, uno de los desafíos más significativos identificados en este módulo es su rigidez estructural, lo que impide la adaptación a contextos variados sin una reconfiguración compleja. La estructura actual del sistema no permite una personalización completa de las opciones de reserva según las necesidades cambiantes del entorno universitario, limitando su capacidad para responder de manera ágil a nuevas demandas y cambios operativos.

El uso de tecnologías obsoletas en el desarrollo de XABAL SIGA ha resultado en una acumulación de deuda tecnológica. Esta deuda incrementa la vulnerabilidad del sistema ante amenazas de seguridad y limita su capacidad para integrar nuevas funcionalidades o adaptarse a los estándares modernos de desarrollo de software.

El diagnóstico realizado indica una necesidad imperiosa de modernización para mantener la relevancia del sistema en un entorno cada vez más exigente. Para mejorar la eficiencia y la adaptabilidad del Módulo de Reservación en XABAL SIGA, se han explorado opciones para incrementar la flexibilidad en la configuración de reservas, permitiendo ajustes dinámicos según las necesidades específicas de cada cliente. Además, es crucial considerar la actualización tecnológica y la reestructuración de su arquitectura para asegurar que

XABAL SIGA pueda continuar siendo una herramienta útil y eficiente en la gestión alimentaria.

1.4. Análisis de los Módulos de Reservación de Soluciones Homólogas Existentes

Con el objetivo de identificar las características fundamentales que deben incluir los módulos de reservación en sistemas de gestión, se llevó a cabo un estudio exhaustivo de estos componentes en diversos sistemas informáticos disponibles en el mercado actual. Este análisis se enfoca en identificar las funcionalidades más relevantes y las mejores prácticas adoptadas por sistemas líderes, con el fin de establecer un estándar óptimo para mejorar la experiencia del usuario en la gestión de servicios. Se realizó un análisis de plataformas que gestionan recursos de manera análoga a XABAL SIGA, abarcando tanto el ámbito internacional como el nacional. Este estudio comparativo permitió identificar las mejores prácticas y oportunidades de mejora.

1.4.1. Oracle Food and Beverage

Oracle Food and Beverage es una solución integral diseñada para optimizar la gestión de restaurantes y servicios de alimentos. Esta plataforma está destinada a empresas de todos los tamaños, desde pequeños restaurantes independientes hasta grandes cadenas de franquicias. Su enfoque se centra en proporcionar una suite completa de herramientas que cubren todas las áreas de la operación diaria de un establecimiento de alimentos y bebidas (Food y Beverage, 2024).

El módulo de reservaciones de Oracle Food and Beverage permite a los restaurantes gestionar eficazmente las reservas de sus clientes. Algunas características clave incluyen:

- **Gestión de Mesas:** Permite asignar mesas de manera eficiente y ajustar el plan de asientos en tiempo real.
- **Reservas en Línea:** Integración con plataformas de reservas en línea, permitiendo a los clientes hacer reservaciones directamente a través del sitio web del restaurante o aplicaciones móviles.
- **Historial de Clientes:** Registro de las preferencias y el historial de visitas de los clientes para personalizar el servicio.
- **Recordatorios y Confirmaciones:** Envío automático de recordatorios y confirmaciones de reserva a los clientes vía SMS o correo electrónico.

Oracle Food and Beverage está construido utilizando una combinación de tecnologías avanzadas y propietarias que aseguran su robustez y escalabilidad. Los principales lenguajes y tecnologías utilizados incluyen:

- **Java:** Utilizado para el desarrollo de la lógica del servidor, garantizando un alto rendimiento y estabilidad.

- **PL/SQL:** Lenguaje de programación específico de Oracle utilizado para desarrollar procedimientos almacenados y desencadenadores en la base de datos.
- **Tecnologías Propietarias de Oracle:** Incluyen diversas herramientas y frameworks desarrollados internamente que optimizan el rendimiento y la integración del sistema.
- **Middleware de Oracle:** Facilita la integración y comunicación entre diferentes módulos y sistemas externos.

Otros aspectos importantes:

- **API Abierta:** Permite la integración con otros sistemas y aplicaciones, como contabilidad, recursos humanos y sistemas de marketing.
- **Personalización:** Altamente configurable para adaptarse a las necesidades específicas de cada negocio, desde la interfaz de usuario hasta los procesos operativos.

Oracle Food and Beverage es una solución poderosa y versátil para la gestión de restaurantes y servicios de alimentos. Su combinación de módulos integrales, tecnologías avanzadas y enfoque en la experiencia del cliente la convierten en una opción ideal para empresas que buscan optimizar sus operaciones y crecer en un mercado competitivo.

1.4.2. CBORD: Solución Integral para la Gestión de Servicios de Alimentación en Instituciones

CBORD es el proveedor líder de software que conecta credenciales, servicio de alimentos y comercio en la educación superior, la atención médica, la vida para personas mayores y los campus comerciales. Con una sólida trayectoria en la industria, CBORD proporciona una gama completa de herramientas y funcionalidades que permiten a las organizaciones administrar sus operaciones de manera eficiente, mejorar la satisfacción del cliente y cumplir con las normativas de seguridad alimentaria (*CBORD* 2024).

El módulo de reservaciones de CBORD está diseñado para manejar eficientemente las reservas de los clientes en entornos institucionales, facilitando la optimización del espacio y los recursos. Las características clave incluyen:

- **Gestión de Mesas y Espacios:** Permite asignar y reorganizar mesas y espacios de comedor de manera eficiente, asegurando un flujo continuo y optimizado de comensales.
- **Reservas en Línea:** Integración con sistemas de reservas en línea, permitiendo a los usuarios hacer reservaciones a través de sitios web o aplicaciones móviles.
- **Confirmaciones y Recordatorios Automatizados:** Envío automático de confirmaciones y recordatorios a los clientes a través de correo electrónico o SMS, reduciendo las tasas de no presentación.
- **Historial y Preferencias del Cliente:** Registro detallado del historial de reservas y preferencias de los clientes, facilitando un servicio personalizado.

CBORD está desarrollado utilizando una combinación de tecnologías avanzadas que aseguran su robustez, escalabilidad y facilidad de integración:

- **Java:** Utilizado principalmente para el desarrollo de la lógica del servidor y aplicaciones backend, garantizando un alto rendimiento y fiabilidad.
- **.NET:** Empleado para desarrollar diversas aplicaciones y servicios dentro del ecosistema CBORD, facilitando la integración con otras plataformas y sistemas empresariales.
- **Tecnologías Web Modernas:** Incluyen HTML5, CSS3 y JavaScript que proporcionan interfaces de usuario ricas y responsivas.

1.4.3. Análisis del Estudio Realizado a los Módulos de Reservación de Soluciones Homólogas Existentes

Para realizar un análisis de los módulos de reservación de los sistemas CBORD y Oracle Food and Beverage según los criterios establecidos, se presentan las características de cada sistema, seguido de una tabla resumen. Los criterios considerados son:

- **Facilidad de Implementación:** Evaluar el tiempo y recursos necesarios para la implementación del sistema en una organización.
- **Experiencia del Usuario (UX):** Analizar la usabilidad y la satisfacción del usuario final con la interfaz del sistema.
- **Escalabilidad:** Considerar la capacidad del sistema para crecer y manejar un volumen creciente de transacciones y usuarios.
- **Seguridad:** Revisar las características de seguridad del sistema, incluyendo la protección de datos y la gestión de acceso.
- **Soporte y Documentación:** Evaluar la calidad del soporte técnico y la documentación disponible para la implementación y el mantenimiento del sistema.
- **Disponibilidad Web:** Es la posibilidad de acceder a través de internet a la plataforma.

Tabla 1.1. Resumen de soluciones homologas

Criterios	**CBORD**	**Oracle Food and Beverage**
Facilidad de Implementación	Compleja	Requiere expertos
Experiencia del Usuario (UX)	Intuitiva	Profesional pero compleja
Escalabilidad	Alta	Alta
Seguridad	Alta	Alta
Soporte y Documentación	Excelente	Excelente
Disponibilidad Web	Sí	Sí

Luego del análisis se puede llegar a las siguientes conclusiones:

- **Tecnologías utilizadas:** CBORD utiliza Java, .NET y tecnologías web modernas, mientras que Oracle usa Java, PL/SQL y tecnologías propietarias.
- **Facilidad de Implementación:** Ambos sistemas pueden ser complejos de implementar, requiriendo tiempo y recursos.
- **Experiencia del Usuario (UX):** Ambos sistemas ofrecen interfaces robustas, aunque Oracle puede ser más compleja para nuevos usuarios.
- **Escalabilidad:** Ambos sistemas son altamente escalables.
- **Seguridad:** Ambos sistemas ofrecen altos estándares de seguridad.
- **Soporte y Documentación:** Ambos sistemas cuentan con excelente soporte y documentación.
- **Disponibilidad Web:** Ambos sistemas son accesibles vía web.

Recomendaciones para XABAL SIGA 2.0:

- **Mantener el Uso de JavaScript:** Continuar usando JavaScript para el desarrollo de la interfaz de usuario.
- **Implementar Seguimiento y Mantenimiento Regular:** Asegurar actualizaciones de seguridad y mejoras funcionales.
- **Uso de Herramientas de Software Libre:** Considerar herramientas de software libre para promover la accesibilidad y reducir costos.

Este análisis de los módulos de reservación de CBORD y Oracle Food and Beverage proporciona una base sólida para la definición de las tecnologías y herramientas necesarias en el desarrollo del módulo de reservación XABAL SIGA 2.0.

1.5. Selección de herramientas y tecnologías

La elección de las herramientas y tecnologías adecuadas es crucial en el desarrollo de software, ya que influye directamente en el éxito del proyecto, la facilidad de desarrollo y la escalabilidad futura. Los factores a considerar incluyen la experiencia del equipo de desarrollo con las tecnologías, las expectativas de rendimiento del proyecto, la compatibilidad con otras aplicaciones, y los riesgos de seguridad. Además, se debe evaluar si las tecnologías elegidas tienen un ecosistema y una comunidad activa que puedan garantizar soporte a largo plazo (Technology, 2021).

Cada proyecto tiene sus propios requisitos, como el tipo de plataforma (web, móvil, etc.), lo cual influye directamente en la selección de la tecnología. También se deben tener en cuenta factores como la escalabilidad y el rendimiento, especialmente para proyectos de mayor tamaño, así como los plazos de entrega y los costos.

Este enfoque estructurado garantiza que la tecnología elegida no solo cumpla con las necesidades actuales del proyecto, sino que también permita un crecimiento y mantenimiento efectivos en el futuro.

Selección de tecnologías para el desarrollo del frontend: El desarrollo del módulo de reservación del sistema XABAL SIGA 2.0 se basará en React, debido a su arquitectura de componentes, flexibilidad y rendimiento. La elección de JavaScript, junto con tecnologías complementarias como Lenguaje de Marcado de Hipertexto, versión 5 (HTML5, por sus siglas en inglés) y Hojas de Estilo en Cascada, versión 3 (CSS3, por sus siglas en inglés), permitirá crear una interfaz de usuario intuitiva y eficiente, adaptada a diversas plataformas y dispositivos.

Marcos de trabajo para el backend: Python y Django Rest Framework (DRF) han sido seleccionados para el desarrollo del backend del módulo de reservación. Esta elección se debe a la robustez del Mapeo Relacional de Objetos (ORM, por sus siglas en inglés) de Django, su sistema de autenticación, y su capacidad para manejar APIs RESTful de manera eficiente, proporcionando una base segura y escalable para la gestión de reservas.

Gestión de la base de datos: PostgreSQL ha sido escogido como el sistema de gestión de bases de datos para el módulo de reservación, debido a su integridad, extensibilidad y capacidad para manejar grandes volúmenes de datos de manera eficiente. La integración con Django facilitará la gestión de estructuras de datos complejas y garantizará la seguridad y consistencia de la información almacenada.

Herramientas complementarias de desarrollo: Visual Studio Code será el Entorno de Desarrollo Integrado (IDE, por sus siglas en inglés) principal, elegido por su extensibilidad y soporte para las tecnologías utilizadas en el proyecto. Además, se utilizarán herramientas como Git, que es un sistema de control de versiones distribuido gratuito y de código abierto diseñado para manejar todo, desde proyectos pequeños hasta proyectos muy grandes con velocidad y eficiencia (Community, 2024).

La selección de tecnologías para esta investigación se realizó basándose en una investigación previa sobre XABAL SIGA 2.0, la cual ofreció un análisis detallado de las mejores prácticas, tendencias emergentes y herramientas más adecuadas para abordar los desafíos específicos (Molina y Ferreiro Ávila, 2023). A partir de dicha investigación, se identificaron las tecnologías más compatibles y eficientes para el desarrollo, garantizando la optimización de recursos y el cumplimiento de los requisitos del proyecto de manera efectiva.

1.6. Metodología de Desarrollo

El concepto de metodología es "conjunto de métodos coherentes y relacionados por unos principios comunes". El concepto de desarrollo, está vinculado a la acción de desarrollar o a las consecuencias de este accionar, por lo tanto es necesario, rastrear el significado del verbo desarrollar: se trata de incrementar, agrandar, extender, ampliar o aumentar alguna característica de algo físico (concreto) o intelectual (abstracto) (Ignacio y Paola, 2015).

La evolución del desarrollo de software ha dado lugar a diversos enfoques metodológicos, cada uno adaptado a las exigencias cambiantes de la industria tecnológica. Estos enfoques proporcionan marcos de trabajo que guían a los equipos en la creación de soluciones efectivas, considerando factores como la complejidad del proyecto, el tamaño del equipo y la interacción con todos los interesados.

En la actualidad, se distinguen dos paradigmas predominantes: los enfoques tradicionales y los ágiles. Los primeros, caracterizados por su énfasis en la planificación exhaustiva y el control riguroso, priorizan la documentación detallada y la predicción de resultados. En contraste, los enfoques ágiles abrazan la flexibilidad, centrándose en ciclos cortos de desarrollo, la colaboración estrecha con el cliente y la adaptabilidad ante los cambios.

Para el desarrollo del módulo de reservación de XABAL SIGA 2.0, se ha optado por XP debido a su enfoque en la calidad del código, la retroalimentación rápida y la satisfacción del cliente. XP promueve un conjunto de valores y prácticas que incluyen (*Extreme Programming Values* s.f.):

- **Simplicidad:** Haremos lo que sea necesario y solicitado, pero no más. Esto maximizará el valor creado por la inversión realizada hasta la fecha. Daremos pequeños pasos simples para alcanzar nuestro objetivo y mitigaremos los errores a medida que ocurran. Crearemos algo de lo que estemos orgullosos y lo mantendremos a largo plazo por costos razonables.
- **Comunicación:** Todos somos parte del equipo y nos comunicamos cara a cara a diario. Trabajaremos juntos en todo, desde los requisitos hasta el código. Crearemos la mejor solución posible para nuestro problema.
- Comentarios: Tomaremos en serio cada compromiso de iteración entregando software funcional. Demostramos nuestro software con anticipación y con frecuencia, luego escuchamos atentamente y hacemos los cambios necesarios. Hablaremos sobre el proyecto y adaptaremos nuestro proceso a él, no al revés.
- **Respeto:** todos dan y sienten el respeto que merecen como miembros valiosos del equipo. Todos aportan valor, incluso si es simplemente entusiasmo. Los desarrolladores respetan la experiencia de los clientes y viceversa. La gerencia respeta nuestro derecho a aceptar la responsabilidad y recibir autoridad sobre nuestro propio trabajo.
- **Valor:** diremos la verdad sobre el progreso y las estimaciones. No documentamos excusas para el fracaso porque planeamos tener éxito. No tememos a nada porque nadie trabaja solo. Nos adaptaremos a los cambios cuando ocurran.

Las prácticas clave de XP en el desarrollo del módulo de reservación incluyen:

- Planificación colaborativa basada en historias de usuario.
- Diseño evolutivo que evita la sobre ingeniería.

- Pruebas como eje central, desarrolladas antes del código (Desarrollo Dirigido por Pruebas (TDD, por sus siglas en inglés)).
- Refactorización continua para mejorar la calidad interna del software.
- Integración continua para detectar problemas tempranamente.
- Estándares de codificación para mantener la coherencia.

La adopción de XP en este proyecto busca acelerar el ciclo de desarrollo, mejorar la calidad del producto y aumentar la satisfacción tanto del equipo como del cliente. Al involucrar activamente al cliente en el proceso, se asegura que el software desarrollado se alinee estrechamente con las necesidades del negocio.

1.7. Conclusiones del capítulo

El capítulo culmina con las siguientes conclusiones:

- Se estableció el punto de partida para el desarrollo, basado en el análisis de fortalezas y debilidades del módulo de reservación vigente.
- Se definió el ecosistema tecnológico:
  ✓Lenguaje y framework backend: Python con Django Rest Framework.
  ✓Biblioteca para frontend: React.js.
  ✓Sistema de gestión de base de datos: PostgreSQL.
  ✓Marco de trabajo: Metodología ágil XP.

- Se fijaron pautas de calidad para el nuevo módulo, derivadas de la evaluación comparativa con sistemas similares en el mercado.

CAPÍTULO 2

DISEÑO DE LA SOLUCIÓN PROPUESTA AL PROBLEMA CIENTÍFICO

En este capítulo, se presenta el diseño detallado de la solución propuesta para resolver el problema científico identificado, abarcando desde la arquitectura del sistema hasta los mecanismos de almacenamiento, procesamiento, y transmisión de datos, así como el manejo de errores. Se estructura en cuatro secciones, que incluyen el modelado de procesos, la ingeniería de requisitos, el diseño arquitectónico, y la implementación de mecanismos específicos para garantizar la eficiencia y robustez del sistema.
Cada sección contribuye a desarrollar una solución integral que responde a las necesidades del sistema de gestión de alimentación de la UCI, asegurando su adaptabilidad, escalabilidad y seguridad, y cumpliendo con los objetivos de la investigación.

2.1. Modelado de la propuesta de solución

En esta sección se realiza el modelado de la propuesta de solución del Módulo de Reservación de XABAL SIGA 2.0. El objetivo de este modelado es proporcionar una representación clara y estructurada de cómo se llevará a cabo la gestión de reservas dentro del sistema, permitiendo una mejor comprensión de su funcionamiento y facilitando su desarrollo e implementación.

El proceso principal de reservación se divide en varias etapas clave: la realización de reservas propias, a familiares o a usuarios asociados, la modificación y cancelación de reservas, y la generación de reportes. Cada una de estas etapas se descompone en subprocesos específicos que describen las acciones necesarias para completar exitosamente cada tarea.

- **Realización de Reservas:** Se elimina la opción de reserva por platos, centrando el módulo en la gestión de reservas por menús. Los usuarios podrán gestionar reservas de menús, con la posibilidad de reservar menús disponibles para sí mismos o para otros, incluyendo familiares, usuarios asociados o terceros. Además el sistema permitirá realizar reservas flexibles, adaptándose a diferentes categorías de las personas.

- **Cancelación de las Reservas:** Los usuarios podrán cancelar las reservaciones realizadas.
- **Generación de Reportes:** El proceso incluye la generación de reportes sobre las reservas realizadas, facturaciones, distribuciones, accesos y tarjetas del usuario. Además de poder vizualizar y filtrar el hitorial de menús y los platos disponibles.
- **Arquitectura Escalable y Distribuida:** Se propone un rediseño de la arquitectura que permita una gran escalabilidad de datos sin necesidad de crear nuevas instancias de la aplicación. Esta nueva arquitectura no solo ayudará a gestionar el crecimiento de los datos, sino que también mejorará la flexibilidad, la modularidad y la mantenibilidad general del sistema.
- **Modernización Tecnológica:** Se propone un cambio tecnológico, que no solo abordará las vulnerabilidades actuales del sistema, sino que también optimizará su rendimiento, escalabilidad y mantenibilidad.

2.2. Fase de Planificación

En esta fase el cliente establece la prioridad de cada historia de usuario, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Una entrega debería obtenerse en no más de tres meses. Esta fase dura unos pocos días (Letelier y Letelier, 2006). La fase de planificación en XP es continua y se divide en dos niveles: planificación de la liberación y planificación de la iteración. En la planificación de la liberación, el equipo de desarrollo y el cliente identifican las "historias de usuario" prioritarias y las estiman en términos de tiempo y complejidad. Luego, en la planificación de la iteración, se seleccionan las historias a trabajar en ciclos cortos de una o dos semanas, desglosándolas en tareas más pequeñas. Este enfoque permite entregar valor funcional al cliente de manera regular, adaptándose continuamente a cambios y necesidades a lo largo del proyecto.

2.2.1. Historias de Usuarios

Una historia de usuario es el resultado de conversaciones entre los interesados del proyecto, los analistas de negocios, los encargados de pruebas y los desarrolladores. Una historia de usuario es una representación de un requisito de software escrito en una o dos frases utilizando el lenguaje común del interesado (Suaza; García y Jaramillo, 2015). Las historias de usuario en XP son descripciones breves y simples de funcionalidades que el sistema debe ofrecer, escritas desde la perspectiva del usuario final. Estas historias se centran en lo que el usuario quiere lograr con el software, evitando detalles técnicos y proporcionando una guía clara para el equipo de desarrollo. Sirven como una herramienta de comunicación entre el cliente y el equipo, permitiendo priorizar funcionalidades, estimar el esfuerzo necesario y guiar la planificación de cada iteración. Cada historia de usuario debe ser pequeña, manejable y representar un valor tangible para el usuario final.

El cliente proporcionó un total de 13 Historia de Usuario (HU) para el desarrollo del módulo. A continuación, se presentan ejemplos de las HU (números 1 (HU1) y 2 (HU2)) que fueron creadas para este proyecto. Las demás historias de usuario pueden consultarse en el Anexo A del documento.

Tabla 2.1. Historia de usuario # 1

Historia de usuario	
Número: 1	**Nombre:** Reservar
Usuario: Todos los usuarios del sistema	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 1.4	**Iteración asignada:** 1
Programador responsable: Alejandro Labaut Caro	
Descripción: • Muestra una lista de menús con sus respectivos eventos y platos, cada menú con la opción de Reservar o Cancelar Reserva. • Permite seleccionar todos los menus y reservar o cancelar los menus según su estado. • Solo se muestran los menús disponibles para reservar. • Se muestran la canatidad de menus configurada segun los elementos a mostrar en la institución.	
Observaciones: • El usuario debe estar autenticado para realizar la acción. • Esta función puede no estar disponible según las configuraciones de la institución.	

Tabla 2.2. Historia de usuario # 2

Historia de usuario	
Número: 2	**Nombre:** Reservar a familiares
Usuario: Todos los usuarios del sistema	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 1.3	**Iteración asignada:** 1
Programador responsable: Alejandro Labaut Caro	

Continúa en la próxima página

Tabla 2.2. Continuación de la página anterior

Descripción:

- Muestra una lista de los familiares del usuario con la opción de Realizar Reserva.
- Realizar Reserva:
  - Muestra una lista de menús con sus respectivos eventos y platos que permite reservar o cancelar la reserva de cada menú.
  - Muestra la información del familiar a reservarle.
  - Permite seleccionar todos los menus y reservar o cancelar los menus según su estado.
  - Solo se muestran los menús disponibles para reservar.
  - Se muestran la canatidad de menus configurada segun los elementos a mostrar en la institución.

Observaciones:

- El usuario debe estar autenticado para realizar la acción.
- Esta función puede no estar disponible según las configuraciones de la institución.
- El usuario debe tener familiares registrados en el sistema

2.2.2. Requisitos no funcionales

Los requisitos no funcionales se refieren a los atributos de calidad de un sistema que definen cómo funciona en lugar de qué hace. A diferencia de los requisitos funcionales, que especifican las acciones y tareas que debe realizar un sistema, los requisitos no funcionales se centran en las características generales y el comportamiento del sistema en diversas condiciones. Abordan aspectos como el rendimiento, la facilidad de uso, la fiabilidad y la escalabilidad, garantizando que el sistema cumpla con los estándares de calidad y proporcione una experiencia de usuario satisfactoria (Bravo; Palacio; Hilari y Tost, 2020). A continuación se especifican los requisitos no funcionales del módulo de reservación:

RnF de Escalabilidad: El sistema debe ser capaz de manejar un incremento en el volumen de datos y usuarios sin afectar su rendimiento.

RnF de Rendimiento: Las operaciones críticas, como consultas, generación de reportes y reservaciones, deben ejecutarse en tiempos óptimos, garantizando una experiencia fluida para el usuario.

RnF de Seguridad: Implementación de autenticación robusta y medidas para prevenir ataques comunes como inyección SQL, XSS y CSRF, asegurando la protección de los datos de los usuarios y las operaciones del sistema.

RnF de Disponibilidad: El sistema debe estar disponible desde cualquier lugar mediante una interfaz web,

asegurando accesibilidad continua y confiable para los usuarios.

RnF de Mantenibilidad: El código debe ser fácil de entender, modificar y extender, facilitando el mantenimiento futuro y la incorporación de nuevas funcionalidades.

RnF de Compatibilidad: El sistema debe garantizar compatibilidad con navegadores modernos y dispositivos variados, adaptándose a diferentes plataformas.

RnF de Usabilidad: La interfaz debe ser intuitiva y sencilla, permitiendo a los usuarios realizar reservaciones y otras operaciones con facilidad, minimizando errores operativos.

RnF de Confiabilidad: El sistema debe garantizar la consistencia de los datos, incluso en caso de fallos, mediante mecanismos de validación y transacciones seguras.

RnF de Eficiencia en el uso de recursos: El sistema debe utilizar de manera óptima los recursos del servidor y la base de datos, reduciendo costos operativos y maximizando la sostenibilidad técnica.

2.2.3. Estimación de Esfuerzo por HU

La estimación de esfuerzo en historias de usuario suele realizarse utilizando puntos de historia, una métrica relativa que mide la complejidad y el esfuerzo necesario para completar una tarea. Los enfoques modernos combinan técnicas como el aprendizaje profundo y procesamiento del lenguaje natural para mejorar la precisión en proyectos ágiles (Fu y Tantithamthavorn, 2022).

Los valores asignados permiten priorizar tareas y establecer la velocidad del equipo, es decir, la cantidad de puntos de historia que pueden completarse en un intervalo de tiempo. Al involucrar a todo el equipo en la estimación, se fomenta un entendimiento común de los requisitos y los desafíos asociados. Este enfoque no solo mejora la precisión de las estimaciones, sino que también facilita la planificación iterativa y la adaptación a cambios frecuentes en los requisitos (Cohn, 2004). Las estimaciones se ajustan con base en la experiencia del equipo y los comentarios recibidos durante el ciclo de desarrollo. A continuación, se presenta una tabla que detalla la estimación de esfuerzos por HU:

Tabla 2.3. Estimación de esfuerzo por historia de usuario

Iteración	Historias de usuario		Puntos estimados (semanas)
1	1	Reservar	1.4
	2	Reservar a Familiares	1.3
	3	Reservar a Asociados	1.3
	4	Reservar a Terceros	1.3

Continúa en la próxima página

Tabla 2.3. Continuación de la página anterior

	5	Reservar por Categorias	1.5
2	6	Reporte de Reservaciones	0.6
	7	Reporte de Distribuciones	0.9
	8	Reporte de Facturaciones	0.8
	9	Reporte de Accesos	0.7
	10	Reporte de Tarjetas	0.6
	11	Reporte de Menús	0.5
	12	Reporte de Platos	0.5
	13	Portada del Módulo	0.6
Total			**12.0**

2.2.4. Plan de Iteraciones

El plan de iteraciones en XP se centra en seleccionar historias de usuario que maximicen el valor del negocio, tomando en cuenta las dependencias y complejidades de implementación. Esta práctica se ajusta iterativamente para garantizar entregas frecuentes que aborden objetivos de negocio prioritarios y promuevan la adaptabilidad del proceso de desarrollo (Valkenhoef; Tervonen; Brock y Postmus, 2010). Durante el plan de iteraciones, el equipo de desarrollo colabora con el cliente para seleccionar las historias de usuario que se abordarán, basándose en su prioridad y el esfuerzo estimado. Este enfoque permite una retroalimentación constante, lo que facilita la adaptación a cambios en los requisitos o nuevas necesidades del cliente. Al final de cada iteración, se presenta una versión funcional del producto, lo que promueve la mejora continua y asegura que el proyecto avance de manera alineada con las expectativas del usuario.

Tabla 2.4. Plan de duración de las iteraciones

Iteración	Historias de usuario		Duración (semanas)
1	1	Reservar	6.8
	2	Reservar a Familiares	
	3	Reservar a Asociados	
	4	Reservar a Terceros	
	5	Reservar por Categorias	
2	6	Reporte de Reservaciones	5.2
	7	Reporte de Distribuciones	
	8	Reporte de Facturaciones	
	9	Reporte de Accesos	
	10	Reporte de Tarjetas	
	11	Reporte de Menús	
	12	Reporte de Platos	
	13	Portada del Módulo	
Total			**12.0**

2.2.5. Plan de Entregas

El Plan de Entregas en XP es una estrategia que define cómo y cuándo se entregarán las diferentes versiones del software a lo largo del ciclo de desarrollo. Este plan establece un cronograma para las entregas incrementales, alineando las funcionalidades desarrolladas con las expectativas del cliente. Cada entrega incluye un conjunto de historias de usuario completadas y validadas, permitiendo al cliente evaluar el progreso del proyecto y proporcionar retroalimentación continua. Al organizar las entregas de manera estructurada, el equipo puede garantizar que cada versión del producto sea funcional y útil, facilitando la adaptación a cambios y la incorporación de nuevas necesidades a medida que surgen durante el desarrollo. En la siguiente tabla se presentan las iteraciones con su fecha de entrega:

Entregable	**Iteración 1**	**Iteración 2**
Fecha Inicio	02 / 07 / 2024	04 / 09 / 2024
Fecha Fin	02 / 09 / 2024	04 / 10 / 2024

Tabla 2.5. Fechas de las Iteraciones

2.3. Fase de Diseño

La fase de diseño en XP se centra en la creación de una arquitectura simple y funcional que permita al equipo responder rápidamente a cambios en los requisitos. A diferencia de enfoques tradicionales donde el diseño detallado precede al desarrollo, XP adopta un diseño evolutivo. Esto significa que el diseño inicial es intencionalmente ligero y se mejora iterativamente en base a la retroalimentación obtenida durante las iteraciones.

Durante esta fase, el equipo utiliza técnicas como la diagramación en papel, tarjetas CRC y reuniones de diseño colectivo para identificar las principales responsabilidades y colaboraciones entre los componentes del sistema. Además, el principio de "diseño simple"guía las decisiones, promoviendo soluciones claras que sean fáciles de entender, implementar y refactorizar.

Este enfoque asegura que el diseño no se convierta en un obstáculo para la entrega rápida de valor, sino en un facilitador que equilibra la simplicidad con la flexibilidad para adaptarse a los cambios (Holtzblatt y Beyer, 2018).

2.3.1. Diseño de la Arquitectura

La arquitectura es un plano para un sistema. Proporciona una abstracción asociada para gestionar la complejidad del sistema y establecer un mecanismo de comunicación y coordinación entre los elementos. Define una resolución estructurada para cumplir con todos los requisitos técnicos y operacionales, mientras optimiza atributos de calidad comunes como el rendimiento y la seguridad. Además, involucra un conjunto de

decisiones críticas relacionadas con la organización del desarrollo del sistema de software, y cada una de esas decisiones tendrá un impacto sustancial en la calidad, mantenibilidad, rendimiento y el éxito general del producto final (Jaiswal, 2019).

Dado los desafíos que enfrenta XABAL SIGA, como la falta de escalabilidad y el uso de tecnologías desactualizadas, se ha decidido rediseñar su arquitectura para adaptarla a un modelo más moderno y flexible. En este contexto, se ha optado por una arquitectura basada en SaaS, con la finalidad de ofrecer una solución más adaptable, eficiente y personalizada. SaaS es un modelo de distribución de software basado en la nube, donde los usuarios acceden a aplicaciones a través de internet en lugar de instalar y mantener el software en dispositivos locales. Esto elimina la necesidad de infraestructura compleja y permite a los usuarios trabajar desde cualquier lugar con acceso a internet (Feder, 2022). Esta nueva propuesta permitirá que las instituciones accedan a la aplicación desde la nube, brindando la capacidad de modificar ciertas características según sus necesidades específicas.

Este enfoque se apoya en una infraestructura centralizada que permitirá a los usuarios interactuar con el sistema desde cualquier dispositivo, utilizando una interfaz ligera, como un navegador web. La capacidad de escalar recursos de manera dinámica y bajo demanda es una de las grandes ventajas de este nuevo modelo, lo que resuelve uno de los problemas más críticos del sistema actual.

El diseño arquitectónico se estructura en torno a un servidor central que actúa como eje del sistema. Este servidor no solo aloja la aplicación, sino también la base de datos, lo que permite centralizar las operaciones y facilitar el manejo de la información. A través de una Interfaz de Programación de Aplicaciones (API, por sus siglas en inglés) desarrollada en DRF, el servidor gestiona las comunicaciones con las interfaces de usuario, las cuales están construidas en React. Estas interfaces son personalizadas para cada institución, permitiendo a los usuarios acceder de manera segura y exclusiva a sus datos.

El proceso de interacción entre la aplicación y los usuarios sigue un flujo constante y seguro. Cuando un cliente accede a la aplicación, se comunica con el servidor de aplicaciones que aloja la interfaz React. Esta, a su vez, realiza solicitudes a la API de Django para recuperar o enviar datos. Toda esta comunicación se maneja a través de protocolos, Protocolo de Transferencia de Hipertexto (HTTP, por sus siglas en inglés) o Protocolo Seguro de Transferencia de Hipertexto (HTTPS, por sus siglas en inglés), asegurando que la transferencia de información sea tanto rápida como protegida.

La seguridad juega un papel esencial en este rediseño. Se ha implementado una capa robusta que garantiza que tanto los datos almacenados como las comunicaciones entre los diferentes componentes del sistema estén protegidos. Esto incluye la utilización de técnicas avanzadas para autenticar y autorizar a los usuarios, así como la encriptación de datos y medidas de prevención frente a ataques comunes como inyecciones Len-

guaje de Consulta Estructurada (SQL, por sus siglas en inglés), Secuencia de Comandos en Sitios Cruzados (XSS, por sus siglas en inglés), y Falsificación de Petición en Sitios Cruzados (CSRF, por sus siglas en inglés).

En cuanto a la base de datos, esta sigue un enfoque centralizado, almacenando toda la información del sistema y permitiendo que los diferentes componentes interactúen con ella de forma eficiente. La API de Django actúa como puente entre la base de datos y el servidor de aplicaciones, asegurando que las operaciones de lectura y escritura se realicen de manera segura y coherente.

La nueva arquitectura no solo ofrece mejoras considerables en cuanto a rendimiento y escalabilidad, sino que también proporciona una plataforma más flexible y segura, alineada con las necesidades actuales y futuras de XABAL SIGA 2.0. Con esta transición a una arquitectura basada en SaaS, el sistema estará mejor equipado para ofrecer servicios personalizados a las instituciones y manejar eficientemente los recursos en la nube.

2.3.2. Patrones de Diseño

Los patrones de diseño son soluciones típicas a problemas que ocurren comúnmente en el diseño de software. Son como planos predefinidos que se pueden personalizar para resolver un problema de diseño recurrente en tu código (*What's a design pattern?* 2024). En XP, una de las técnicas esenciales para optimizar la calidad del código y mejorar la productividad del equipo de desarrollo es el uso de patrones de diseño. Al aplicar estos enfoques, los equipos logran crear sistemas más modulares, flexibles y fáciles de mantener, lo que a su vez fomenta una mejor colaboración y comunicación entre los desarrolladores.

Un grupo fundamental de patrones en el desarrollo orientado a objetos son los Patrones de software de asignación de responsabilidad general (GRASP, por sus siglas en inglés), que facilitan la asignación de responsabilidades de manera organizada y eficiente. Por ejemplo, el patrón Creador permite delegar la creación de objetos a clases específicas, lo que en este proyecto se ha implementado en los serializadores. El patrón Controlador gestiona la lógica del sistema, y en este caso ha sido aplicado en las vistas. El patrón Experto en Información distribuye la responsabilidad de manejar datos a la clase que posee el conocimiento necesario, y en este sistema fue empleado en los modelos.

Otro conjunto importante son los patrones de diseño del grupo Gang of Four (GoF), utilizados para resolver problemas comunes en la arquitectura de software. En este proyecto, el patrón Decorador se utilizó para añadir funcionalidades adicionales a los objetos mediante envoltorios especiales, lo cual resultó particularmente útil en la documentación de una API usando Swagger, permitiendo una descripción más organizada y reutilizable de sus elementos. El uso de estos patrones en XP contribuye a mantener un código limpio, adaptable y de alta calidad, alineado con los principios ágiles.

2.3.3. Tarjetas CRC

Las Tarjetas CRC son una herramienta clave en XP para diseñar y organizar la arquitectura de un sistema orientado a objetos de forma colaborativa. Estas tarjetas permiten a los desarrolladores identificar las clases que forman parte del sistema, definir sus responsabilidades y determinar qué otras clases colaboran para cumplir con esas responsabilidades. El uso de CRC fomenta un enfoque claro y sencillo del diseño, facilitando la discusión entre los miembros del equipo sobre cómo debe estructurarse el sistema.

Un modelo de tarjetas CRC es una colección de tarjetas estándar que se dividen en tres secciones. Una clase representa un conjunto de objetos similares, una responsabilidad es algo que una clase sabe o hace, y un colaborador es otra clase con la que una clase interactúa para cumplir con sus responsabilidades (*Class Responsibility Collaborator (CRC) Cards* 2023).

El proceso de creación de tarjetas CRC suele ser dinámico y colaborativo, a menudo realizado en una pizarra o en papel, permitiendo al equipo simular interacciones entre clases y refinar el diseño del sistema de manera iterativa. Este enfoque refuerza uno de los principios fundamentales de XP: el diseño debe ser lo suficientemente simple y flexible como para evolucionar a medida que el sistema crece. Las tarjetas CRC permiten visualizar la distribución de responsabilidades, asegurando que el diseño sea coherente y bien estructurado, lo que resulta en un código más limpio y fácil de mantener.
Las siguiente tabla representa la tarjeta CRC elaborada en el marco de esta investigación.

Tabla 2.6. Tarjeta CRC # 1

Tarjeta CRC	
Clase: TbDreservacion	
Responsabilidad	**Colaboración**

Continúa en la próxima página

Tabla 2.6. Continuación de la página anterior

• **create():**Este método maneja la solicitud POST para crear una nueva reservación. • **list():**Este método maneja la solicitud GET para obtener una lista de reservaciones. • **retrieve():**Este método maneja la solicitud GET para recuperar una reservación. • **update():**Este método maneja la solicitud PUT para actualizar una reservación. • **partial_update():**Este método maneja la solicitud PATCH para actualizar parcialmente una reservación. • **destroy():**Este método maneja la solicitud DELETE para eliminar una reservación.	Institucion Persona TbDmenu TbNconfiguracionCobro models.Model

2.4. Conclusiones del capítulo

En este capítulo de la investigación se han establecido las bases fundamentales para el desarrollo del sistema, llegando a las siguientes conclusiones:

- **Diseño integral:** Se definió un diseño claro y detallado del Módulo de Reservación de XABAL SIGA 2.0.
- **Arquitectura escalable:** La adopción de una arquitectura SaaS garantiza flexibilidad y escalabilidad.
- **Procesos eficientes:** Se mejoró la gestión de reservaciones para usuarios, familiares y terceros.
- **Metodología ágil:** El uso de historias de usuario y XP asegura un desarrollo enfocado y adaptable.
- **Seguridad mejorada:** Se integraron medidas para proteger el sistema y garantizar su rendimiento.
- **Patrones de diseño:** Se aplicaron patrones para mejorar la modularidad y mantenibilidad del sistema.

CAPÍTULO 3

DESARROLLO Y VERIFICACIÓN DEL MÓDULO

En este capítulo se abordarán el desarrollo y la validación del Módulo de Reservación de XABAL SIGA 2.0, utilizando la metodología XP. Este enfoque permitirá realizar un desarrollo iterativo y enfocado en la retroalimentación constante para asegurar la entrega incremental de valor y la mejora continua del sistema.

El proceso incluirá tres niveles de pruebas para garantizar la calidad del módulo:

- **Pruebas unitarias**, destinadas a verificar el correcto funcionamiento de los componentes individuales.
- **Pruebas de integración**, orientadas a evaluar la interacción entre los diferentes módulos del sistema.
- **Pruebas de aceptación**, que buscarán validar el cumplimiento de los requisitos definidos en las historias de usuario.

3.1. Fase de Desarrollo

La fase de desarrollo en XP se centra en la creación rápida y continua de software mediante ciclos cortos de desarrollo, llamados iteraciones. Durante esta fase, el equipo implementa las funcionalidades definidas en las HU, asegurando que el código sea simple, funcional y de alta calidad. XP promueve prácticas como el TDD, donde las pruebas se crean antes de escribir el código, y la refactorización constante para mejorar el diseño del software sin cambiar su comportamiento externo. La comunicación continua con el cliente y la integración frecuente del código son clave para garantizar que el software se ajuste a los requerimientos y pueda adaptarse a cambios rápidamente.

3.1.1. Tareas de Ingeniería

Las tareas de ingeniería en el desarrollo de software abarcan un conjunto amplio de actividades destinadas a diseñar, construir y mantener sistemas de software que cumplan con las necesidades de los usuarios y las restricciones del proyecto. Estas tareas incluyen la definición de requisitos, el diseño de la arquitectura, la codificación, las pruebas y la implementación. Además, el proceso de ingeniería de software pone un fuerte

énfasis en la calidad, asegurándose de que el producto sea confiable, eficiente y fácil de mantener. A través de un enfoque sistemático y disciplinado, las tareas de ingeniería buscan reducir riesgos y proporcionar soluciones escalables y sostenibles para problemas complejos (Pressman, 2010).

Las tareas de ingeniería son actividades técnicas específicas que el equipo de desarrollo realiza para implementar las funcionalidades definidas en las historias de usuario. Estas tareas desglosan cada historia en acciones concretas que deben completarse durante una iteración, como escribir código, realizar pruebas, o diseñar la arquitectura del sistema. Las tareas de ingeniería están orientadas a mantener el enfoque en la calidad del software, la simplicidad del diseño y la entrega continua de valor al cliente. A continuación, se presentan ejemplos de las tareas de ingeniería de esta investigación, el resto de las tareas podrán encontrarse en el Anexo B de este documento:

Tabla 3.1. Tarea de ingeniería # 1

Tarea	
Número de tarea: 1	**Número de Historia de usuario:** 1
Nombre de la tarea: Implementar funcionalidades de reservación	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.6
Fecha de inicio: 1 de julio de 2024	**Fecha de fin:** 7 de julio de 2024
Programador responsable: Alejandro Labaut Caro	
Descripción: El objetivo de esta tarea es desarrollar e integrar las funcionalidades necesarias para que los usuarios puedan realizar y cancelar reservaciones dentro del sistema, con opciones para Reservar o Cancelar Reserva. Esto incluye: • Crear la lógica backend que gestione las reservas y su estado (reservado/cancelado). • Permitir la selección de todos los menús en lote para su reserva o cancelación, de acuerdo al estado actual. • Implementar validaciones, como la disponibilidad de menús, y restricciones según la configuración de la institución. • Integrar la conexión con la base de datos para registrar y cancelar reservas. • Incluir la funcionalidad de envío de notificaciones al usuario al confirmar sus acciones. Finalmente, se integrará con la interfaz frontend para asegurar una experiencia de usuario coherente y sin interrupciones en el flujo de reservación.	

Tabla 3.2. Tarea de ingeniería # 2

Tarea	
Número de tarea: 2	**Número de Historia de usuario:** 1
Nombre de la tarea: Desarrollar vista para realizar y cancelar reservaciones del usuario	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.8
Fecha de inicio: 8 de julio de 2024	**Fecha de fin:** 17 de julio de 2024
Programador responsable: Alejandro Labaut Caro	

Continúa en la próxima página

Tabla 3.2. Continuación de la página anterior

Descripción: Esta tarea consiste en implementar la interfaz gráfica para que los usuarios puedan realizar y cancelar reservaciones de manera intuitiva. La vista debe:

- Mostrar menús con eventos y platos disponibles para reservación.
- Incluir opciones para Reservar o Cancelar Reserva para cada menú.
- Permitir la selección múltiple para reservar o cancelar menús en lote.
- Mostrar la confirmación de las acciones realizadas.

La vista se conectará con el backend a través de una API, enviando los datos de la reservación o cancelación y recibiendo confirmación en tiempo real. Se debe asegurar que la interfaz sea responsiva, utilizando React para el frontend y ajustada a dispositivos móviles y de escritorio.

3.2. Fase de Pruebas

En la metodología de desarrollo ágil XP, la fase de pruebas es un pilar fundamental para garantizar la calidad del software. En XP, las pruebas son un proceso continuo que acompaña a todo el ciclo de desarrollo, en lugar de ser una etapa separada al final del proyecto. Incluye prácticas clave como el TDD y la integración continua. Estas pruebas se dividen en pruebas unitarias, que aseguran el correcto funcionamiento de pequeñas unidades del código, y pruebas funcionales, que validan que las historias de usuario cumplan con los requisitos definidos por el cliente. La retroalimentación rápida proporcionada por las pruebas en XP permite a los equipos adaptarse a cambios frecuentes en los requisitos y detectar errores de manera temprana, minimizando riesgos y costos (Beck y Andres, 2004).

3.2.1. Pruebas Unitarias

Las pruebas unitarias son el proceso de determinar si las unidades individuales de código fuente están funcionando como se espera (Khorikov, 2020). Cada prueba unitaria evalúa una pequeña parte del sistema de manera aislada para asegurar que produzca los resultados esperados bajo diferentes condiciones. Estas pruebas son fundamentales para detectar errores en las primeras etapas del desarrollo, facilitando la identificación y corrección de fallos de forma rápida y eficiente. En metodologías como XP, las pruebas unitarias suelen escribirse antes del código, siguiendo la práctica de TDD, lo que asegura que el código esté bien diseñado y sea fácilmente mantenible.

A continuación se muestra un ejemplo de algunas de las pruebas realizadas al modelo TbDreservacion del módulo de reservación:

Código fuente 3.1. Ejemplo de código de las pruebas realizadas.

```
1 @pytest.mark.django_db
2 def test_create_reservacion(setup_data):
3     institucion, persona, menu, configuracion_cobro = setup_data
```

```
4       reservacion = TbDreservacion.objects.create(
5           id_institucion=institucion,
6           id_persona=persona,
7           id_menu=menu,
8           id_configuracion_cobro=configuracion_cobro
9       )
10      assert reservacion.id_reservacion is not None
11      assert reservacion.activo is True
12      assert reservacion.fecha_reservacion is not None
13      assert reservacion.fecha_cancelacion is None
14      assert reservacion.costo_reservacion is None
15      assert reservacion.token_reservacion is not None
16      assert reservacion.cobrada is None
17      assert reservacion.id_persona_distribucion is None
18      assert reservacion.guardado is None
19
20  @pytest.mark.django_db
21  def test_generate_unique_token():
22      token = TbDreservacion.generate_unique_token()
23      assert token is not None
24      assert not TbDreservacion.objects.filter(token_reservacion=token).exists()
25
26  @pytest.mark.django_db
27  def test_str_method(setup_data):
28      institucion, persona, menu, configuracion_cobro = setup_data
29      reservacion = TbDreservacion.objects.create(
30          id_institucion=institucion,
31          id_persona=persona,
32          id_menu=menu,
33          id_configuracion_cobro=configuracion_cobro
34      )
35      assert str(reservacion) == f"{reservacion.id_reservacion} - {reservacion.
            id_persona} - MenÃº: {reservacion.id_menu}"
36
37  @pytest.mark.django_db
38  def test_save_method_generates_token(setup_data):
39      institucion, persona, menu, configuracion_cobro = setup_data
40      reservacion = TbDreservacion(
41          id_institucion=institucion,
42          id_persona=persona,
43          id_menu=menu,
44          id_configuracion_cobro=configuracion_cobro
45      )
46      reservacion.save()
47      assert reservacion.token_reservacion is not None
48
```

```
49 @pytest.mark.django_db
50 def test_reservacion_cancelada(setup_data):
51     institucion, persona, menu, configuracion_cobro = setup_data
52     reservacion = TbDreservacion.objects.create(
53         id_institucion=institucion,
54         id_persona=persona,
55         id_menu=menu,
56         id_configuracion_cobro=configuracion_cobro,
57         activo=False
58     )
59     assert reservacion.activo is False
```

Las pruebas unitarias se llevaron a cabo en dos iteraciones, asegurando la calidad y funcionalidad del módulo de reservación.

Primera Iteración:
Durante la primera iteración, las pruebas revelaron algunos errores significativos en el sistema:

- **Errores en las restricciones de unicidad:** Se identificaron fallos en la lógica que asegura que un usuario solo pueda realizar una reservación por menú. Esto permitió múltiples registros duplicados para el mismo menú.
- **Problemas con validaciones de campos obligatorios:** Algunos campos, como la fecha de reservación, no se validaban correctamente, resultando en entradas incompletas.
- **Errores en el manejo de fechas:** Se detectaron inconsistencias al procesar reservas con fechas pasadas o fuera del rango permitido, lo que generaba resultados inesperados.

Estos fallos llevaron a ajustes significativos en las validaciones y la lógica de negocio, además de la refactorización del código para mejorar su robustez.

Segunda Iteración:
En la segunda iteración, tras implementar las correcciones necesarias, se ejecutaron nuevamente las pruebas unitarias. Esta vez, todos los casos pasaron con éxito, cumpliendo con las expectativas y verificando que el módulo funcionara correctamente bajo diferentes escenarios.

El resultado final fue satisfactorio, demostrando la eficacia de las pruebas para identificar y resolver errores críticos. Esto garantiza un sistema más estable y preparado para el uso en producción.

3.2.2. Pruebas de Integración

Las pruebas de integración son una fase en el ciclo de vida del desarrollo de software que se enfoca en verificar la interacción y compatibilidad entre diferentes módulos o componentes de un sistema. A diferencia

de las pruebas unitarias, que evalúan componentes individuales de manera aislada, las pruebas de integración se centran en identificar problemas que surgen cuando dichos componentes trabajan en conjunto. Estas pruebas son esenciales para garantizar que las interfaces entre módulos se comporten como se espera y que el sistema en su conjunto funcione correctamente antes de pasar a fases más avanzadas como las pruebas de sistema o aceptación (Sommerville, 2011).

Las pruebas de integración en esta investigación se llevaron a cabo con éxito, validando la correcta interacción entre el módulo de reservación y el módulo de configuración de XABAL SIGA 2.0 ya desarrollado (Molina y Ferreiro Ávila, 2023). Durante estas pruebas, se verificó que las funcionalidades de cada módulo trabajaran en conjunto sin errores ni conflictos, lo que permitió identificar y resolver incompatibilidades de manera temprana. Gracias a este proceso, se aseguró una cohesión adecuada entre los componentes del sistema, garantizando un desempeño estable y una experiencia de usuario fluida. Esto contribuyó a que el sistema cumpla con los requisitos de negocio y esté preparado para su uso en producción.

3.2.3. Pruebas de Aceptación

Las pruebas de aceptación son una fase crucial en el desarrollo de software, cuyo objetivo principal es verificar que el sistema cumple con los requisitos y expectativas del cliente o usuario final. Estas pruebas suelen realizarse en un entorno lo más cercano posible al real y se centran en validar que el producto es funcional, completo y está listo para su implementación. A menudo, las pruebas de aceptación están alineadas con los criterios de aceptación previamente definidos, sirviendo como la última etapa antes de la entrega oficial del software al cliente (Pressman, 2010).

A continuación se presenta un ejemplo de los resultados obtenidos durante las pruebas realizadas a la HU 1 “Reservar“ Tabla 2.1. Los detalles adicionales sobre las pruebas de aceptación para el módulo de reservaciones del sistema XABAL SIGA 2.0 están disponibles en el Anexo C.

Tabla 3.3. Prueba de aceptación # 1

Caso de prueba de aceptación	
Código: HU1_P1	**Historia de usuario:** 1
Nombre: Reservar o cancelar reserva de menus	
Descripción: Prueba para validar la funcionalidad de reservar o cancelar la reserva de un menú disponible.	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • La institución debe tener habilitado el proceso de reservación. • Deben existir menús disponibles para reservar en el sistema.	

Continúa en la próxima página

Tabla 3.3. Continuación de la página anterior

Pasos de ejecución: 1. El sistema muestra una interfaz con los menús disponibles para reservar, y cada menú tiene su botón de Reservar o Cancelar Reserva según su estado de reservado o no. 2. El usuario elige un menú sin reservar y presiona el botón "Reservar" para confirmar la reserva del menú elegido. 3. El sistema muestra una notificación informando que la reserva fue realizada. 4. Opcionalmente, el usuario selecciona un menú ya reservado y presiona el botón "Cancelar Reserva" para anular su reserva. 5. El sistema muestra una notificación informando que la reserva fue cancelada.
Resultados esperados: • La reservación o cancelación se realiza con éxito, mostrando los cambios en los menús afectados y notificando la acción realizada. • En caso de reservar un menú, aparece el botón de "Cancelar Reserva" en el menú correspondiente. • En caso de cancelar la reserva, el estado del menú vuelve a estar disponible para reservar nuevamente.

Se llevaron a cabo un total de 26 pruebas de aceptación distribuidas en dos iteraciones. Durante la primera iteración, se identificaron algunos errores relacionados con la implementación de ciertas funcionalidades, lo que permitió realizar los ajustes necesarios para cumplir con los requisitos establecidos. En la segunda iteración, gracias a las correcciones aplicadas, todas las pruebas fueron superadas exitosamente, validando tanto los requisitos funcionales como no funcionales del sistema y garantizando que el producto final cumpliera con las expectativas del cliente.

3.3. Conclusiones del capítulo

- Se implementaron las funcionalidades y vistas necesarias para la realización, modificación y cancelación de reservas, cumpliendo con los requisitos funcionales definidos en las historias de usuario.
- Las pruebas unitarias fueron realizadas de manera continua, lo que permitió detectar y corregir errores tempranamente, asegurando un código más robusto y mantenible.
- Las pruebas de integración aseguraron la correcta interacción entre los componentes del sistema.
- Las pruebas de aceptación validaron que el sistema desarrollado cumpliera con las expectativas del cliente, garantizando que las funcionalidades respondieran a sus necesidades operativas.
- La metodología XP, aplicada en las fases de desarrollo y pruebas, facilitó una retroalimentación constante y una mejora continua, lo que resultó en un sistema flexible, escalable y adaptable a futuros cambios.

Conclusiones

A través del desarrollo de esta investigación, se alcanzaron los siguientes resultados:

- Mediante la actualización tecnológica, con herramientas como DRF y React, se superaron las deficiencias de rendimiento y seguridad, logrando un sistema robusto, eficiente y alineado con estándares modernos.
- La implementación de una arquitectura basada en SaaS garantizó una mayor eficiencia operativa, modularidad y capacidad de personalización, resolviendo las limitaciones estructurales del sistema anterior.
- Se validó la funcionalidad y calidad del Módulo de Reservación de XABAL SIGA 2.0 mediante pruebas unitarias, de integración y de aceptación, confirmando su viabilidad técnica y operativa para su despliegue en entornos reales.
- Se modernizó exitosamente el Módulo de Reservación de XABAL SIGA, implementando una solución flexible, escalable y adaptable que supera las restricciones de personalización y escalabilidad del sistema original, respondiendo efectivamente a los desafíos planteados en está investigación.

Recomendaciones

- Continuar con el desarrollo de los módulos restantes del sistema XABAL SIGA 2.0
- Integrar las pasarelas de pago en línea existentes en Cuba al módulo de reservación de XABAL SIGA 2.0 para facilitar el proceso de pago a los usuarios.

Acrónimos

API Interfaz de Programación de Aplicaciones. 23, 24

CRC Clase-Responsabilidad-Colaborador. 22, 25
CSRF Falsificación de Petición en Sitios Cruzados. 24
CSS3 Hojas de Estilo en Cascada, versión 3. 13

DRF Django Rest Framework. 13, 23, 34

GoF Gang of Four. 24
GRASP Patrones de software de asignación de responsabilidad general. 24

HTML5 Lenguaje de Marcado de Hipertexto, versión 5. 13
HTTP Protocolo de Transferencia de Hipertexto. 23
HTTPS Protocolo Seguro de Transferencia de Hipertexto. 23
HU Historia de Usuario. 18, 27, 32

IDE Entorno de Desarrollo Integrado. 13

ORM Mapeo Relacional de Objetos. 13

SaaS Software como Servicio. 23, 24, 34
SGA Sistema de Gestión Alimentaria. 6
SQL Lenguaje de Consulta Estructurada. 23

TDD Desarrollo Dirigido por Pruebas. 15, 27, 29

UCI Universidad de las Ciencias Informáticas. 2, 6–8, 16

XABAL SIGA XABAL Sistema Integral de Gestión de Alimentación. 2, 6–9, 23, 34
XABAL SIGA 2.0 XABAL Sistema Integral de Gestión de Alimentación Versión 2.0. 2, 3, 5, 12, 13, 16, 24, 26, 27, 32, 34, 35
XP Programación Extrema. 14, 15, 17, 21, 22, 24–27, 29, 33
XSS Secuencia de Comandos en Sitios Cruzados. 24

Referencias bibliográficas

ASALE, RAE- y RAE, 2024. *sistema | Diccionario de la lengua española* [online] [visitado 2024-07-23]. Disponible desde: https://dle.rae.es/sistema (vid. pág. 5).

BECK, Kent y ANDRES, Cynthia, 2004. *Extreme Programming Explained: Embrace Change*. 2nd. Boston: Addison-Wesley Professional (vid. pág. 29).

BRAUDEL, Fernand, 1982. *The Structures of Everyday Life: Civilization and Capitalism 15th-18th Century*. Harper & Row (vid. pág. 1).

BRAVO, María José Salamea; PALACIO, Liliana González; HILARI, Marc Oriol y TOST, Carles Farré, 2020. Estimación y priorización de requisitos no-funcionales para desarrollo de software: Estado del arte. En: *Estimación y priorización de requisitos no-funcionales para desarrollo de software: Estado del arte*. *Conferencia Internacional de Ingeniería: Desarrollo e innovación en ingeniería*. Instituto Antioqueño de Investigación, págs. 150-157. ISBN 978-958-52333-4-8. Url: http://hdl.handle.net/2117/336760 (vid. pág. 19).

CBORD, 2024 [online] [visitado 2024-07-23]. Disponible desde: https://www.cbord.com/ (vid. pág. 10).

Class Responsibility Collaborator (CRC) Cards, 2023. *Class Responsibility Collaborator (CRC) Cards: An Agile Intro* [online] [visitado 2024-11-22]. Disponible desde: https://agilemodeling.com/artifacts/crcmodel.htm (vid. pág. 25).

COHN, Mike, 2004. *User Stories Applied: For Agile Software Development*. Boston: Addison-Wesley Professional (vid. pág. 20).

COMMUNITY, The Git, 2024. *Git - Distributed Version Control System*. Url: https://git-scm.com (vid. pág. 13).

Extreme Programming Values. Url: http://www.extremeprogramming.org/values.html (vid. pág. 14).

FEDER, Michael, 2022. Cloud computing: Definition, uses and benefits. *University of Phoenix*. Url: https://www.phoenix.edu/blog/cloud-computing-and-cloud-technology.html (vid. pág. 23).

FOOD, Oracle y BEVERAGE. *Restaurant Technology Solutions* [online] [visitado 2024-07-23]. Disponible desde: https://www.oracle.com/food-beverage/ (vid. pág. 9).

FU, Michael y TANTITHAMTHAVORN, Chakkrit, 2022. GPT2SP: A Transformer-Based Agile Story Point Estimation Approach. *IEEE Transactions on Software Engineering*. Vol. PP. Disponible desde DOI: 10.1109/TSE.2022.3158252 (vid. pág. 20).

GELDRES TRUJILLO, Betto Martin, 2015. *Influencia de la informatización del proceso de reservaciones de un restaurante en las ventas y la satisfacción del cliente*. Url: https://repositorio.upao.edu.pe/handle/20.500.12759/2810. Universidad Privada Antenor Orrego (vid. pág. 1).

GIRALDO, Marcos, 2023. *El origen de los restaurantes: Historia y evolución de un lugar para disfrutar la gastronomía.* [online] [visitado 2024-08-24]. Disponible desde: https://comosurgen.com/como-surgen-los-restaurantes/ (vid. pág. 1).

HOLTZBLATT, Karen y BEYER, Hugh, 2018. *Agile Experience Design: A Digital Designer's Guide to Agile, Lean, and Continuous*. Elsevier (vid. pág. 22).

IGNACIO, Carlos y PAOLA, Verónica, 2015. Metodologías actuales de desarrollo de software. N.º 2015 (vid. pág. 13).

JAISWAL, Manishaben, 2019. *Software Architecture and Software Design* [online]. Rochester, NY [visitado 2024-11-22]. Disponible desde DOI: 10.2139/ssrn.3772387. Informe técnico (vid. pág. 23).

KHORIKOV, Vladimir, 2020. *Unit Testing Principles, Practices, and Patterns*. Simon y Schuster (vid. pág. 29).

LAUDON, Kenneth C., 2012. *Management Information Systems: Managing the Digital Firm*. 12th. Upper Saddle River, NJ: Pearson Education (vid. pág. 5).

LEAL, Mgt. Marcela, 2022. *Gestión en un servicio de alimentación* [online] [visitado 2024-09-23]. Disponible desde: https://redalimentariafoodtech.com/nota/501082-gestion-en-un-servicio-de-alimentacion-2 (vid. pág. 6).

LETELIER, Patricio y LETELIER, Patricio, 2006. *www.cyta.com.ar/ta0502/v5n2a1.htm*. Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP). Disponible desde DOI: Artículo (vid. pág. 17).

MARINESCU, Dan C.; PAYA, Ashkan y MORRISON, John P., 2017. A Cloud Reservation System for Big Data Applications. *IEEE Transactions on Parallel and Distributed Systems*. Vol. 28, n.º 3, págs. 606-618. Disponible desde DOI: 10.1109/TPDS.2016.2594783 (vid. pág. 6).

MEHDAWY, Magda y HUSSEIN, Amr, 2010. *The Pharaohs Kitchen: Recipes from Ancient Egypts Enduring Food Traditions*. American University in Cairo Press (vid. pág. 1).

MOLINA, Alejandro Javier y FERREIRO ÁVILA, Enrique, 2023. *Módulo de Configuración para Sistema de Gestión de Alimentación XABAL SIGA versión 2.0.* B.S. thesis. Universidad de las Ciencias Informáticas. Facultad 4. (vid. págs. 13, 32).

PRESSMAN, Roger S., 2010. *Software Engineering: A Practitioner's Approach*. 7th. New York: McGraw-Hill Education (vid. págs. 28, 32).

SOMMERVILLE, Ian, 2011. *Software Engineering*. 9th. Boston: Pearson Education (vid. pág. 32).

SUAZA, Katerine Villamizar; GARCÍA, John Jairo Tabares y JARAMILLO, Carlos Mario Zapata, 2015. Mejora de historias de usuario y casos de prueba de metodologías ágiles con base en TDD. *Cuaderno activa*. Vol. 7, págs. 41-53. Url: `https://ojs.tdea.edu.co/index.php/cuadernoactiva/article/view/246` (vid. pág. 17).

TECHNOLOGY, TPP, 2021. Software Development: Choosing the Right Technology for Your Project. *TPP Technology Blog*. Url: `https://www.tpptechnology.com/blog/software-development-choosing-the-right-technology-for-your-project` (vid. pág. 12).

VALKENHOEF, Gert; TERVONEN, Tommi; BROCK, Bert y POSTMUS, Douwe, 2010. Product and Release Planning Practices for Extreme Programming. En: *Product and Release Planning Practices for Extreme Programming*. Vol. 48, págs. 238-243. ISBN 9783642130533. Disponible desde DOI: `10.1007/978-3-642-13054-0_25` (vid. pág. 21).

What's a design pattern? [online] [visitado 2024-09-22]. Disponible desde: `https://refactoring.guru/design-patterns/what-is-pattern` (vid. pág. 24).

Generado con LaTeX: 26 de noviembre de 2024: 11:08pm

Apéndices

APÉNDICE A

Historias de Usuario

Tabla A.1. Historia de usuario # 3

Historia de usuario	
Número: 3	**Nombre:** Reservar a Asociados
Usuario: Todos los usuarios del sistema	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 1.3	**Iteración asignada:** 1
Programador responsable: Alejandro Labaut Caro	
Descripción: • Muestra una lista de las personas asociadas al usuario con la opción de Realizar Reserva. • Realizar Reserva: ◦ Muestra una lista de menús con sus respectivos eventos y platos que permite reservar o cancelar la reserva de cada menú. ◦ Muestra la información de la persona asociada a reservarle. ◦ Permite seleccionar todos los menus y reservar o cancelar los menus según su estado. ◦ Solo se muestran los menús disponibles para reservar. ◦ Se muestran la canatidad de menus configurada segun los elementos a mostrar en la institución.	
Observaciones: • El usuario debe estar autenticado para realizar la acción. • Esta función puede no estar disponible según las configuraciones de la institución. • El usuario debe tener usuarios asociados en el sistema	

Tabla A.2. Historia de usuario # 4

Historia de usuario	
Número: 4	**Nombre:** Reservar a Terceros
Usuario: Todos los usuarios del sistema	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 1.3	**Iteración asignada:** 1
Programador responsable: Alejandro Labaut Caro	
Descripción: • Muestra una lista de todas las personas que pertenecen a la institución del usuario con la opción de Realizar Reserva. • Realizar Reserva: ◦ Muestra una lista de menús con sus respectivos eventos y platos que permite reservar o cancelar la reserva de cada menú. ◦ Muestra la información de la persona a reservarle. ◦ Permite seleccionar todos los menus y reservar o cancelar los menus según su estado. ◦ Solo se muestran los menús disponibles para reservar. ◦ Se muestran la canatidad de menus configurada segun los elementos a mostrar en la institución.	
Observaciones: • El usuario debe estar autenticado para realizar la acción. • Esta función puede no estar disponible según las configuraciones de la institución.	

Tabla A.3. Historia de usuario # 5

Historia de usuario	
Número: 5	**Nombre:** Reservar por Categorias
Usuario: Todos los usuarios del sistema	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 1.5	**Iteración asignada:** 1
Programador responsable: Alejandro Labaut Caro	
Descripción: • Muestra un formulario con los campos obligatorios Categoría, Categoría de Residente y Fecha del Menú a reservar, acompañados de un botón de Reservar. • Al seleccionar la categoría, la categoría de residente y la fecha del menú se muestra una lista de las personas que cumplen con estas categorias y no tienen reservación para la fecha del menú seleccionado y una lista de las personas que ya tienen un reservación.	

Continúa en la próxima página

Tabla A.3. Continuación de la página anterior

Observaciones:
• El usuario debe estar autenticado para realizar la acción. • Esta función puede no estar disponible según las configuraciones de la institución. • Solo se podra reservar si hay un menú disponible para la fecha del menú seleccionada y hay usuarios que cumplen con las categorias seleccionadas.

Tabla A.4. Historia de usuario # 6

Historia de usuario	
Número: 6	**Nombre:** Reporte de Reservaciones
Usuario: Todos los usuarios del sistema	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Bajo
Puntos estimados: 0.6	**Iteración asignada:** 1
Programador responsable: Alejandro Labaut Caro	
Descripción: • Muestra una lista con las reservaciones realizadas por el usuario, pudiendo ver también las reservaciones realizadas a usuarios asociados y a familiares. • Permite ver los detalles de las reservaciones. • Muestra información del usuario. • Permite exportar las reservaciones realizadas a un pdf y descargarlas. • Permite seleccionar la cantidad de reservaciones a mostrar en la lista y realizar la paginación de todas las reservaciones.	
Observaciones: • El usuario debe estar autenticado para realizar la acción.	

Tabla A.5. Historia de usuario # 7

Historia de usuario	
Número: 7	**Nombre:** Reporte de Distribuciones
Usuario: Todos los usuarios del sistema	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Bajo
Puntos estimados: 0.9	**Iteración asignada:** 1
Programador responsable: Alejandro Labaut Caro	

Continúa en la próxima página

Tabla A.5. Continuación de la página anterior

Descripción: • Muestra una lista con las distribuciones del usuario, pudiendo ver también las distribuciones de los usuarios asociados y familiares. • Muestra información del usuario. • Permite exportar las distribuciones a un pdf y descargarlas. • Permite seleccionar la cantidad de distribuciones a mostrar en la lista y realizar la paginación de todas las distribuciones.
Observaciones: • El usuario debe estar autenticado para realizar la acción.

Tabla A.6. Historia de usuario # 8

Historia de usuario	
Número: 8	**Nombre:** Reporte de Facturaciones
Usuario: Todos los usuarios del sistema	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Bajo
Puntos estimados: 0.8	**Iteración asignada:** 1
Programador responsable: Alejandro Labaut Caro	
Descripción: • Muestra una lista con las facturaciones del usuario, pudiendo ver también las facturaciones de los usuarios asociados y familiares. • Muestra información del usuario. • Permite exportar las facturaciones a un pdf y descargarlas. • Permite seleccionar la cantidad de facturaciones a mostrar en la lista y realizar la paginación de todas las facturaciones.	
Observaciones: • El usuario debe estar autenticado para realizar la acción.	

Tabla A.7. Historia de usuario # 9

Historia de usuario	
Número: 9	**Nombre:** Reporte de Accesos
Usuario: Todos los usuarios del sistema	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Bajo

Continúa en la próxima página

Tabla A.7. Continuación de la página anterior

Puntos estimados: 0.7	**Iteración asignada:** 1
Programador responsable: Alejandro Labaut Caro	
Descripción: • Muestra una lista con los accesos del usuario, pudiendo ver accesos de los usuarios asociados y familiares. • Muestra información del usuario. • Permite exportar los accesos a un pdf y descargarlos. • Permite seleccionar la cantidad de accesos a mostrar en la lista y realizar la paginación de todos los accesos.	
Observaciones: • El usuario debe estar autenticado para realizar la acción.	

Tabla A.8. Historia de usuario # 10

Historia de usuario	
Número: 10	**Nombre:** Reporte de Tarjetas
Usuario: Todos los usuarios del sistema	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Bajo
Puntos estimados: 0.6	**Iteración asignada:** 1
Programador responsable: Alejandro Labaut Caro	
Descripción: • Muestra una lista con las tarjetas del usuario. • Muestra información del usuario. • Permite seleccionar la cantidad de tarjetas a mostrar en la lista y realizar la paginación de todas las tarjetas.	
Observaciones: • El usuario debe estar autenticado para realizar la acción.	

Tabla A.9. Historia de usuario # 11

Historia de usuario	
Número: 11	**Nombre:** Reporte de Platos
Usuario: Todos los usuarios del sistema	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Bajo
Puntos estimados: 0.5	**Iteración asignada:** 1
Programador responsable: Alejandro Labaut Caro	

Continúa en la próxima página

Tabla A.9. Continuación de la página anterior

Descripción: • Muestra una lista con los platos registrados por la institución en el sistema. • Permite buscar platos y filtrar por unidad de medida, clasificación de plato y la composición. • Permite seleccionar la cantidad de platos a mostrar en la lista.
Observaciones: • El usuario debe estar autenticado para realizar la acción. • Solo se muestran los platos activos.

Tabla A.10. Historia de usuario # 12

Historia de usuario	
Número: 12	**Nombre:** Reporte de Menús
Usuario: Todos los usuarios del sistema	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Bajo
Puntos estimados: 0.5	**Iteración asignada:** 1
Programador responsable: Alejandro Labaut Caro	
Descripción: • Muestra una lista con los menús registrados por la institución en el sistema. • Permite filtrar por un intervalo de fechas.	
Observaciones: • El usuario debe estar autenticado para realizar la acción. • Solo se muestran los menus activos.	

Tabla A.11. Historia de usuario # 13

Historia de usuario	
Número: 13	**Nombre:** Portada del módulo
Usuario: Todos los usuarios del sistema	
Prioridad en negocio: Baja	**Riesgo en desarrollo:** Bajo
Puntos estimados: 0.6	**Iteración asignada:** 1
Programador responsable: Alejandro Labaut Caro	

Continúa en la próxima página

Tabla A.11. Continuación de la página anterior

Descripción: • Muestra una lista con los menús próximos, a partir de la fecha actual.
Observaciones: • El usuario debe estar autenticado para realizar la acción.

APÉNDICE B

Tareas de Ingeniería

Tabla B.1. Tarea de ingeniería # 3

Tarea	
Número de tarea: 3	**Número de Historia de usuario:** 2
Nombre de la tarea: Implementar funcionalidad de reserva a familiares	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.6
Fecha de inicio: 18 de julio de 2024	**Fecha de fin:** 31 de julio de 2024
Programador responsable: Alejandro Labaut Caro	
Descripción: El objetivo de esta tarea es desarrollar la funcionalidad que permita a los usuarios realizar reservaciones a familiares dentro del sistema. Esto incluye: • Creación de la lógica backend para gestionar la selección de menús, incluyendo la opción de reservar o cancelar reservas. • Conexión con la lista de familiares asociados al usuario, mostrando información del familiar al realizar la reserva. • Implementación de validaciones para autenticación del usuario, existencia de familiares registrados y disponibilidad de menús. Finalmente, se integrará con la interfaz frontend para una experiencia de usuario continua y sin interrupciones.	

Tabla B.2. Tarea de ingeniería # 4

Tarea	
Número de tarea: 4	**Número de Historia de usuario:** 2
Nombre de la tarea: Desarrollar vista para reservas a familiares	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.7
Fecha de inicio: 1 de agosto de 2024	**Fecha de fin:** 8 de agosto de 2024
Programador responsable: Alejandro Labaut Caro	

Continúa en la próxima página

Tabla B.2. Continuación de la página anterior

Descripción: Esta tarea consiste en implementar la interfaz gráfica para realizar reservaciones a familiares. La vista debe: • Mostrar una lista de familiares del usuario con opción de "Realizar Reserva". • Presentar una lista de menús con eventos y platos, permitiendo la reserva o cancelación de cada menú, y mostrar la información del familiar seleccionado. • Permitir la selección de todos los menús en lote, de acuerdo con el estado de cada uno. • Asegurar la navegación entre los menús y el correcto uso de paginación, limitando la visualización a la cantidad de menús configurada en la institución. Además, debe ser intuitiva, responsiva y conectarse al backend mediante API para enviar y recibir información de reservas, incluyendo la posibilidad de realizar o cancelar reservas existentes.

Tabla B.3. Tarea de ingeniería # 5

Tarea	
Número de tarea: 5	**Número de Historia de usuario:** 3
Nombre de la tarea: Implementar funcionalidad de reserva a asociados	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.6
Fecha de inicio: 9 de agosto de 2024	**Fecha de fin:** 22 de agosto de 2024
Programador responsable: Alejandro Labaut Caro	
Descripción: El objetivo de esta tarea es desarrollar la funcionalidad que permita a los usuarios realizar reservaciones a usuarios asociados dentro del sistema. Esto incluye la creación de la lógica backend para gestionar la selección de menús, así como la conexión con la lista de usuarios asociados al usuario. Se debe implementar la capacidad de realizar y cancelar reservas, asegurando que las validaciones necesarias se realicen (como la autenticación del usuario y la existencia de usuarios asociados registrados). También se debe contemplar la paginación de los menús disponibles, limitando la visualización a tres menús por vez y permitiendo la selección de múltiples menús para la reserva. Finalmente, se integrará con la interfaz frontend para proporcionar una experiencia de usuario fluida.	

Tabla B.4. Tarea de ingeniería # 6

Tarea	
Número de tarea: 6	**Número de Historia de usuario:** 3
Nombre de la tarea: Desarrollar vista para reservas a asociados	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 23 de agosto de 2024	**Fecha de fin:** 30 de agosto de 2024
Programador responsable: Alejandro Labaut Caro	

Continúa en la próxima página

Tabla B.4. Continuación de la página anterior

Descripción: Esta tarea consiste en implementar la interfaz gráfica que permitirá a los usuarios seleccionar asociados para realizar las reservaciones y los menús disponibles para reservar. La vista debe mostrar una lista de los menús disponibles, cada uno con sus respectivos platos, y una lista de usuarios asociados al usuario. Se debe garantizar que la interfaz sea intuitiva y responsiva, permitiendo a los usuarios navegar entre los menús y realizar selecciones fácilmente. Además, se incluirá la opción para cancelar reservas, y se asegurará la conexión con el backend a través de la API para enviar y recibir la información necesaria.

Tabla B.5. Tarea de ingeniería # 7

Tarea	
Número de tarea: 7	**Número de Historia de usuario:** 4
Nombre de la tarea: Implementar funcionalidad de reserva a terceros	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.6
Fecha de inicio: 1 de septiembre de 2024	**Fecha de fin:** 14 de septiembre de 2024
Programador responsable: Alejandro Labaut Caro	
Descripción: El objetivo de esta tarea es desarrollar la funcionalidad que permita a los usuarios realizar reservaciones a terceros dentro del sistema. Esto incluye la creación de la lógica backend para gestionar la selección de menús, así como la conexión con la lista de usuarios de la institución. Se debe implementar la capacidad de realizar, modificar y cancelar reservas, asegurando que las validaciones necesarias se realicen (como la autenticación del usuario y las configuraciones de la institución). También se debe contemplar la paginación de los menús disponibles, limitando la visualización a tres menús por vez y permitiendo la selección de múltiples menús para la reserva. Finalmente, se integrará con la interfaz frontend para proporcionar una experiencia de usuario fluida.	

Tabla B.6. Tarea de ingeniería # 8

Tarea	
Número de tarea: 8	**Número de Historia de usuario:** 4
Nombre de la tarea: Desarrollar vista para reservas a terceros	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 15 de septiembre de 2024	**Fecha de fin:** 22 de septiembre de 2024
Programador responsable: Alejandro Labaut Caro	

Continúa en la próxima página

Tabla B.6. Continuación de la página anterior

Descripción: Esta tarea consiste en implementar la interfaz gráfica que permitirá a los usuarios seleccionar terceros para realizar las reservaciones. La vista debe mostrar una lista de los menús disponibles, cada uno con sus respectivos platos, y una lista de todos los usuarios de la institución. Se debe garantizar que la interfaz sea intuitiva y responsiva, permitiendo a los usuarios navegar entre los menús y realizar selecciones fácilmente. Además, se incluirá opción para cancelar reservas, y se asegurará la conexión con el backend a través de la API para enviar y recibir la información necesaria.

Tabla B.7. Tarea de ingeniería # 9

Tarea	
Número de tarea: 9	**Número de Historia de usuario:** 5
Nombre de la tarea: Implementar funcionalidad de reporte de reservaciones	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.6
Fecha de inicio: 23 de septiembre de 2024	**Fecha de fin:** 6 de octubre de 2024
Programador responsable: Alejandro Labaut Caro	
Descripción: El objetivo de esta tarea es desarrollar la funcionalidad que permite a los usuarios ver un reporte de sus reservaciones realizadas, así como las reservaciones a familiares y asociados. Esto incluye la creación de la lógica backend para recuperar la información necesaria de la base de datos y la implementación de la API para enviar esta información al frontend. Además, se deberá permitir que los usuarios seleccionen la cantidad de reservaciones a mostrar en la lista, asegurando que la visualización sea clara y accesible. También se incluirá la opción de ver detalles específicos de cada reservación.	

Tabla B.8. Tarea de ingeniería # 10

Tarea	
Número de tarea: 10	**Número de Historia de usuario:** 5
Nombre de la tarea: Desarrollar vista de reporte de reservaciones	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 7 de octubre de 2024	**Fecha de fin:** 14 de octubre de 2024
Programador responsable: Alejandro Labaut Caro	
Descripción: Esta tarea consiste en implementar la interfaz gráfica que mostrará al usuario un reporte de sus reservaciones. La vista debe incluir una lista de las reservaciones realizadas, permitiendo también la visualización de reservaciones a familiares y asociados. Se debe asegurar que la interfaz sea intuitiva, permitiendo a los usuarios ver los detalles de cada reservación y seleccionar la cantidad de reservaciones a mostrar. Además, la vista se conectará con el backend a través de una API para recuperar la información necesaria, asegurando que el usuario esté autenticado para realizar esta acción.	

Tabla B.9. Tarea de ingeniería # 11

Tarea	
Número de tarea: 11	**Número de Historia de usuario:** 6
Nombre de la tarea: Implementar funcionalidad de reporte de distribuciones	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.6
Fecha de inicio: 15 de octubre de 2024	**Fecha de fin:** 29 de octubre de 2024
Programador responsable: Alejandro Labaut Caro	
Descripción: El objetivo de esta tarea es desarrollar la funcionalidad que permite a los usuarios ver un reporte de las distribuciones, así como las distribuciones de familiares y asociados. Esto incluye la creación de la lógica backend para recuperar la información necesaria de la base de datos y la implementación de la API para enviar esta información al frontend. Se deberá permitir que los usuarios seleccionen la cantidad de distribuciones a mostrar en la lista y ver los detalles específicos de cada distribución.	

Tabla B.10. Tarea de ingeniería # 12

Tarea	
Número de tarea: 12	**Número de Historia de usuario:** 6
Nombre de la tarea: Desarrollar vista de reporte de distribuciones	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 30 de octubre de 2024	**Fecha de fin:** 6 de noviembre de 2024
Programador responsable: Alejandro Labaut Caro	
Descripción: Esta tarea consiste en implementar la interfaz gráfica que mostrará al usuario un reporte de sus distribuciones. La vista debe incluir una lista de las distribuciones realizadas, permitiendo también la visualización de distribuciones a familiares y asociados. Se debe asegurar que la interfaz sea intuitiva, permitiendo a los usuarios ver los detalles de cada distribución y seleccionar la cantidad de distribuciones a mostrar. La vista se conectará con el backend a través de una API para recuperar la información necesaria, asegurando que el usuario esté autenticado para realizar esta acción.	

Tabla B.11. Tarea de ingeniería # 13

Tarea	
Número de tarea: 13	**Número de Historia de usuario:** 7
Nombre de la tarea: Implementar funcionalidad de reporte de facturaciones	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.6
Fecha de inicio: 7 de noviembre de 2024	**Fecha de fin:** 21 de noviembre de 2024
Programador responsable: Alejandro Labaut Caro	

Continúa en la próxima página

Tabla B.11. Continuación de la página anterior

Descripción: El objetivo de esta tarea es desarrollar la funcionalidad que permite a los usuarios ver un reporte de las facturaciones realizadas, así como las facturaciones de familiares y asociados. Esto incluye la creación de la lógica backend para recuperar la información necesaria de la base de datos y la implementación de la API para enviar esta información al frontend. Se deberá permitir que los usuarios seleccionen la cantidad de facturaciones a mostrar en la lista y ver los detalles específicos de cada facturación.

Tabla B.12. Tarea de ingeniería # 14

Tarea	
Número de tarea: 14	**Número de Historia de usuario:** 7
Nombre de la tarea: Desarrollar vista de reporte de facturaciones	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 22 de noviembre de 2024	**Fecha de fin:** 29 de noviembre de 2024
Programador responsable: Alejandro Labaut Caro	
Descripción: Esta tarea consiste en implementar la interfaz gráfica que mostrará al usuario un reporte de sus facturaciones. La vista debe incluir una lista de las facturaciones realizadas, permitiendo también la visualización de facturaciones a familiares y asociados. Se debe asegurar que la interfaz sea intuitiva, permitiendo a los usuarios ver los detalles de cada facturación y seleccionar la cantidad de facturaciones a mostrar. La vista se conectará con el backend a través de una API para recuperar la información necesaria, asegurando que el usuario esté autenticado para realizar esta acción.	

Tabla B.13. Tarea de ingeniería # 15

Tarea	
Número de tarea: 15	**Número de Historia de usuario:** 8
Nombre de la tarea: Implementar funcionalidad de reporte de accesos	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.6
Fecha de inicio: 30 de noviembre de 2024	**Fecha de fin:** 14 de diciembre de 2024
Programador responsable: Alejandro Labaut Caro	
Descripción: El objetivo de esta tarea es desarrollar la funcionalidad que permite a los usuarios ver un reporte de los accesos realizados, así como los accesos de familiares y asociados. Esto incluye la creación de la lógica backend para recuperar la información necesaria de la base de datos y la implementación de la API para enviar esta información al frontend. Se deberá permitir que los usuarios seleccionen la cantidad de accesos a mostrar en la lista.	

Tabla B.14. Tarea de ingeniería # 16

Tarea	
Número de tarea: 16	**Número de Historia de usuario:** 8
Nombre de la tarea: Desarrollar vista de reporte de accesos	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 15 de diciembre de 2024	**Fecha de fin:** 22 de diciembre de 2024
Programador responsable: Alejandro Labaut Caro	
Descripción: Esta tarea consiste en implementar la interfaz gráfica que mostrará al usuario un reporte de sus accesos. La vista debe incluir una lista de los accesos realizados, permitiendo también la visualización de accesos a familiares y asociados. Se debe asegurar que la interfaz sea intuitiva, permitiendo a los usuarios seleccionar la cantidad de accesos a mostrar. La vista se conectará con el backend a través de una API para recuperar la información necesaria, asegurando que el usuario esté autenticado para realizar esta acción.	

Tabla B.15. Tarea de ingeniería # 17

Tarea	
Número de tarea: 17	**Número de Historia de usuario:** 9
Nombre de la tarea: Implementar funcionalidad de reporte de tarjetas	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.6
Fecha de inicio: 23 de diciembre de 2024	**Fecha de fin:** 6 de enero de 2025
Programador responsable: Alejandro Labaut Caro	
Descripción: El objetivo de esta tarea es desarrollar la funcionalidad que permite a los usuarios ver un reporte de las tarjetas asociadas a su cuenta. Esto incluye la creación de la lógica backend para recuperar la información necesaria de la base de datos y la implementación de la API para enviar esta información al frontend. Se deberá asegurar que el usuario esté autenticado para realizar esta acción.	

Tabla B.16. Tarea de ingeniería # 18

Tarea	
Número de tarea: 18	**Número de Historia de usuario:** 9
Nombre de la tarea: Desarrollar vista de reporte de tarjetas	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 7 de enero de 2025	**Fecha de fin:** 14 de enero de 2025
Programador responsable: Alejandro Labaut Caro	

Continúa en la próxima página

Tabla B.16. Continuación de la página anterior

Descripción: Esta tarea consiste en implementar la interfaz gráfica que mostrará al usuario un reporte de sus tarjetas. La vista debe incluir una lista de las tarjetas asociadas, mostrando la información relevante del usuario. Se debe asegurar que la interfaz sea intuitiva y que la vista se conecte con el backend a través de una API para recuperar la información necesaria, garantizando que el usuario esté autenticado para realizar esta acción.

Tabla B.17. Tarea de ingeniería # 19

Tarea	
Número de tarea: 19	**Número de Historia de usuario:** 10
Nombre de la tarea: Implementar funcionalidad de reporte de platos	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.6
Fecha de inicio: 15 de enero de 2025	**Fecha de fin:** 29 de enero de 2025
Programador responsable: Alejandro Labaut Caro	
Descripción: El objetivo de esta tarea es desarrollar la funcionalidad que permite a los usuarios ver un reporte de los platos registrados en el sistema. Esto incluye la creación de la lógica backend para recuperar la información necesaria de la base de datos, filtrando solo los platos activos. Se debe implementar la API correspondiente para enviar esta información al frontend, asegurando que el usuario esté autenticado para realizar esta acción.	

Tabla B.18. Tarea de ingeniería # 20

Tarea	
Número de tarea: 20	**Número de Historia de usuario:** 10
Nombre de la tarea: Desarrollar vista de reporte de platos	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 30 de enero de 2025	**Fecha de fin:** 6 de febrero de 2025
Programador responsable: Alejandro Labaut Caro	
Descripción: Esta tarea consiste en implementar la interfaz gráfica que mostrará al usuario un reporte de los platos registrados. La vista debe incluir una lista de los platos, permitiendo la búsqueda y el filtrado por unidad de medida, clasificación de plato y composición. Además, se debe permitir seleccionar la cantidad de platos a mostrar en la lista. La vista se conectará con el backend a través de una API para recuperar la información necesaria, garantizando que el usuario esté autenticado para realizar esta acción.	

Tabla B.19. Tarea de ingeniería # 21

Tarea	
Número de tarea: 21	**Número de Historia de usuario:** 11

Continúa en la próxima página

Tabla B.19. Continuación de la página anterior

Nombre de la tarea: Implementar funcionalidad de reporte de menús	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.6
Fecha de inicio: 7 de febrero de 2025	**Fecha de fin:** 21 de febrero de 2025
Programador responsable: Alejandro Labaut Caro	
Descripción: El objetivo de esta tarea es desarrollar la funcionalidad que permite a los usuarios ver un reporte de los menús registrados en el sistema. Esto incluye la creación de la lógica backend para recuperar la información necesaria de la base de datos, filtrando solo los menús activos y permitiendo el filtrado por un intervalo de fechas. Se debe implementar la API correspondiente para enviar esta información al frontend, asegurando que el usuario esté autenticado para realizar esta acción.	

Tabla B.20. Tarea de ingeniería # 22

Tarea	
Número de tarea: 22	**Número de Historia de usuario:** 11
Nombre de la tarea: Desarrollar vista de reporte de menús	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.4
Fecha de inicio: 22 de febrero de 2025	**Fecha de fin:** 1 de marzo de 2025
Programador responsable: Alejandro Labaut Caro	
Descripción: Esta tarea consiste en implementar la interfaz gráfica que mostrará al usuario un reporte de los menús registrados. La vista debe incluir una lista de los menús, permitiendo el filtrado por un intervalo de fechas. La vista se conectará con el backend a través de una API para recuperar la información necesaria, garantizando que el usuario esté autenticado para realizar esta acción.	

Tabla B.21. Tarea de ingeniería # 23

Tarea	
Número de tarea: 23	**Número de Historia de usuario:** 12
Nombre de la tarea: Desarrollar vista de la portada del módulo	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.2
Fecha de inicio: 11 de marzo de 2025	**Fecha de fin:** 15 de marzo de 2025
Programador responsable: Alejandro Labaut Caro	
Descripción: Esta tarea consiste en implementar la interfaz gráfica que mostrará al usuario la lista de los menús próximos. La vista debe conectarse al backend a través de una API para recuperar la información y debe garantizar que el usuario esté autenticado para acceder a esta sección.	

Tabla B.22. Tarea de ingeniería # 24

Tarea	
Número de tarea: 24	**Número de Historia de usuario:** 13
Nombre de la tarea: Implementar funcionalidad de reserva por categorías	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.6
Fecha de inicio: 1 de septiembre de 2024	**Fecha de fin:** 14 de septiembre de 2024
Programador responsable: Alejandro Labaut Caro	
Descripción: El objetivo de esta tarea es desarrollar la lógica backend necesaria para realizar reservas por categorías. Esto incluye: • Creación de endpoints que permitan filtrar y consultar personas según las categorías de "Categoría", "Categoría de Residente" y la "Fecha del Menú". • Implementación de validaciones para asegurar que solo las personas que cumplen con los criterios seleccionados y que no tienen reservas previas en la fecha especificada sean retornadas en la lista de resultados. • Verificación de que haya un menú disponible para la fecha seleccionada antes de completar el proceso de reserva. La funcionalidad debe conectarse a la base de datos, garantizando la integridad y disponibilidad de la información necesaria para las reservas por categoría.	

Tabla B.23. Tarea de ingeniería # 25

Tarea	
Número de tarea: 25	**Número de Historia de usuario:** 13
Nombre de la tarea: Desarrollar formulario de reserva por categorías y la lista de usuarios en el frontend	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.6
Fecha de inicio: 15 de septiembre de 2024	**Fecha de fin:** 25 de septiembre de 2024
Programador responsable: Alejandro Labaut Caro	
Descripción: Esta tarea consiste en implementar el formulario de reserva por categorías en la interfaz de usuario. El formulario debe: • Incluir los campos obligatorios de "Categoría", "Categoría de Residente" y "Fecha del Menú" junto con un botón de "Reservar". • Mostrar dinámicamente la lista de personas que cumplen con las categorías seleccionadas y que no tienen reservas previas en la fecha especificada. • Validar los campos y manejar posibles errores, como la ausencia de un menú disponible para la fecha o la falta de personas que coincidan con las categorías seleccionadas. La interfaz debe ser intuitiva y responsiva, y conectarse al backend mediante API para enviar y recibir los datos de las reservas.	

Tabla B.24. Tarea de ingeniería # 26

Tarea	
Número de tarea: 26	**Número de Historia de usuario:** 13
Nombre de la tarea: Validar y gestionar disponibilidad de menús para reservas por categorías	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.3
Fecha de inicio: 26 de septiembre de 2024	**Fecha de fin:** 30 de septiembre de 2024
Programador responsable: Alejandro Labaut Caro	
Descripción: El objetivo de esta tarea es asegurar que el sistema gestione correctamente la disponibilidad de los menús para reservas por categorías. Esto incluye: • Implementar validaciones que confirmen la existencia de un menú disponible para la fecha seleccionada antes de procesar cualquier reserva. • Actualizar la lógica para bloquear reservas en fechas sin disponibilidad de menús y notificar al usuario en caso de que no haya menús habilitados en la fecha elegida. Con esta tarea se garantiza que las reservas se realicen solo cuando existen menús válidos en el sistema para la fecha especificada.	

APÉNDICE C

Pruebas de Aceptación

Tabla C.1. Prueba de aceptación # 2

Caso de prueba de aceptación	
Código: HU2_P1	**Historia de usuario:** 2
Nombre: Reservar para familiares	
Descripción: Prueba para validar la funcionalidad de reservar y cancelar menús para familiares del usuario.	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • La institución debe tener habilitado el proceso de reservación. • Deben existir familiares registrados y menús disponibles.	
Pasos de ejecución: 1. El sistema muestra una interfaz con la lista de familiares del usuario y la opción de Realizar Reserva. 2. El usuario selecciona la opción de Realizar Reserva del familiar al que desea reservar. 3. El sistema muestra una interfaz con los menús disponibles para reservar, y cada menú tiene su botón de Reservar o Cancelar Reserva según su estado de reservado o no. 4. El usuario elige un menú sin reservar y presiona el botón “Reservar” para confirmar la reserva del menú elegido. 5. El sistema muestra una notificación informando que la reserva fue realizada. 6. Opcionalmente, el usuario selecciona un menú ya reservado y presiona el botón “Cancelar Reserva” para anular su reserva. 7. El sistema muestra una notificación informando que la reserva fue cancelada.	

Continúa en la próxima página

Tabla C.1. Continuación de la página anterior

Resultados esperados: • La reservación o cancelación se realiza con éxito, mostrando los cambios en los menús afectados y notificando la acción realizada. • En caso de reservar un menú, aparece el botón de “Cancelar Reserva” en el menú correspondiente. • En caso de cancelar la reserva, el estado del menú vuelve a estar disponible para reservar nuevamente.

Tabla C.2. Prueba de aceptación # 3

Caso de prueba de aceptación	
Código: HU3_P1	**Historia de usuario:** 3
Nombre: Reservar para asociados	
Descripción: Prueba para validar la funcionalidad de reservar y cancelar menús para asociados del usuario.	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • La institución debe tener habilitado el proceso de reservación. • Deben existir asociados asignados al usuario y menús disponibles.	
Pasos de ejecución: 1. El sistema muestra una interfaz con la lista de asociados del usuario y la opción de Realizar Reserva. 2. El usuario selecciona la opción de Realizar Reserva del asociado al que desea reservar. 3. El sistema muestra una interfaz con los menús disponibles para reservar, y cada menú tiene su botón de Reservar o Cancelar Reserva según su estado de reservado o no. 4. El usuario elige un menú sin reservar y presiona el botón “Reservar” para confirmar la reserva del menú elegido. 5. El sistema muestra una notificación informando que la reserva fue realizada. 6. Opcionalmente, el usuario selecciona un menú ya reservado y presiona el botón “Cancelar Reserva” para anular su reserva. 7. El sistema muestra una notificación informando que la reserva fue cancelada.	

Continúa en la próxima página

Tabla C.2. Continuación de la página anterior

Resultados esperados: • La reservación o cancelación se realiza con éxito, mostrando los cambios en los menús afectados y notificando la acción realizada. • En caso de reservar un menú, aparece el botón de "Cancelar Reserva" en el menú correspondiente. • En caso de cancelar la reserva, el estado del menú vuelve a estar disponible para reservar nuevamente.

Tabla C.3. Prueba de aceptación # 4

Caso de prueba de aceptación	
Código: HU4_P1	**Historia de usuario:** 4
Nombre: Reservar para terceros	
Descripción: Prueba para validar la funcionalidad de reservar o cancelar menús para cualquier usuario (terceros).	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • La institución debe tener habilitado el proceso de reservación. • Deben existir usuarios válidos y menús disponibles.	
Pasos de ejecución: 1. El sistema muestra una interfaz con la lista de usuarios del sistema y la opción de Realizar Reserva. 2. El usuario selecciona la opción de Realizar Reserva del usuario al que desea reservar. 3. El sistema muestra una interfaz con los menús disponibles para reservar, y cada menú tiene su botón de Reservar o Cancelar Reserva según su estado de reservado o no. 4. El usuario elige un menú sin reservar y presiona el botón "Reservar" para confirmar la reserva del menú elegido. 5. El sistema muestra una notificación informando que la reserva fue realizada. 6. Opcionalmente, el usuario selecciona un menú ya reservado y presiona el botón "Cancelar Reserva" para anular su reserva. 7. El sistema muestra una notificación informando que la reserva fue cancelada.	

Continúa en la próxima página

Tabla C.3. Continuación de la página anterior

Resultados esperados: • La reservación o cancelación se realiza con éxito, mostrando los cambios en los menús afectados y notificando la acción realizada. • En caso de reservar un menú, aparece el botón de "Cancelar Reserva" en el menú correspondiente. • En caso de cancelar la reserva, el estado del menú vuelve a estar disponible para reservar nuevamente.

Tabla C.4. Prueba de aceptación # 5

Caso de prueba de aceptación	
Código: HU5_P1	**Historia de usuario:** 5
Nombre: Visualizar mis reservaciones	
Descripción: Prueba para validar la funcionalidad de ver la lista de reservaciones realizadas por el usuario.	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • Deben existir reservaciones realizadas por el usuario.	
Pasos de ejecución: 1. El usuario selecciona la pestaña "Mis Reservaciones" en la interfaz de reportes. 2. El sistema muestra una tabla con la lista de reservaciones realizadas por el usuario. 3. El usuario selecciona una reservación y presiona el botón "Ver Detalles".	
Resultados esperados: • El sistema muestra correctamente la lista de reservaciones realizadas por el usuario. • Al presionar "Ver Detalles", se muestran los detalles completos de la reservación seleccionada.	

Tabla C.5. Prueba de aceptación # 6

Caso de prueba de aceptación	
Código: HU5_P2	**Historia de usuario:** 5
Nombre: Visualizar reservaciones de familiares	
Descripción: Prueba para validar la funcionalidad de ver la lista de reservaciones realizadas para los familiares del usuario.	

Continúa en la próxima página

Tabla C.5. Continuación de la página anterior

Condiciones de ejecución: • El usuario debe estar autenticado previamente. • Deben existir reservaciones realizadas para los familiares del usuario.
Pasos de ejecución: 1. El usuario selecciona la pestaña “Reservaciones de Familiares” en la interfaz de reportes. 2. El sistema muestra una tabla con la lista de reservaciones realizadas para los familiares del usuario. 3. El usuario selecciona una reservación y presiona el botón “Ver Detalles”.
Resultados esperados: • El sistema muestra correctamente la lista de reservaciones realizadas para los familiares. • Al presionar “Ver Detalles”, se muestran los detalles completos de la reservación seleccionada.

Tabla C.6. Prueba de aceptación # 7

Caso de prueba de aceptación	
Código: HU5_P3	**Historia de usuario:** 5
Nombre: Visualizar reservaciones de asociados	
Descripción: Prueba para validar la funcionalidad de ver la lista de reservaciones realizadas para los asociados del usuario.	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • Deben existir reservaciones realizadas para los asociados del usuario.	
Pasos de ejecución: 1. El usuario selecciona la pestaña “Reservaciones de Asociados” en la interfaz de reportes. 2. El sistema muestra una tabla con la lista de reservaciones realizadas para los asociados del usuario. 3. El usuario selecciona una reservación y presiona el botón “Ver Detalles”.	
Resultados esperados: • El sistema muestra correctamente la lista de reservaciones realizadas para los asociados. • Al presionar “Ver Detalles”, se muestran los detalles completos de la reservación seleccionada.	

Tabla C.7. Prueba de aceptación # 8

Caso de prueba de aceptación	
Código: HU6_P1	**Historia de usuario:** 6
Nombre: Visualizar mis distribuciones	
Descripción: Prueba para validar la funcionalidad de ver la lista de distribuciones del usuario.	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • Deben existir distribuciones para el usuario.	
Pasos de ejecución: 1. El usuario selecciona la pestaña “Mis Distribuciones” en la interfaz de reportes de Distribuciones. 2. El sistema muestra una tabla con la lista de distribuciones del usuario.	
Resultados esperados: • El sistema muestra correctamente la lista de distribuciones del usuario.	

Tabla C.8. Prueba de aceptación # 9

Caso de prueba de aceptación	
Código: HU6_P2	**Historia de usuario:** 6
Nombre: Visualizar distribuciones de familiares	
Descripción: Prueba para validar la funcionalidad de ver la lista de distribuciones de los familiares del usuario.	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • Deben existir distribuciones para los familiares del usuario.	
Pasos de ejecución: 1. El usuario selecciona la pestaña “Distribuciones de Familiares” en la interfaz de reportes de Distribuciones. 2. El sistema muestra una tabla con la lista de distribuciones para los familiares del usuario.	
Resultados esperados: • El sistema muestra correctamente la lista de distribuciones realizadas para los familiares.	

Tabla C.9. Prueba de aceptación # 10

Caso de prueba de aceptación	
Código: HU6_P3	**Historia de usuario:** 6
Nombre: Visualizar distribuciones de asociados	
Descripción: Prueba para validar la funcionalidad de ver la lista de distribuciones para los asociados del usuario.	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • Deben existir distribuciones para los asociados del usuario.	
Pasos de ejecución: 1. El usuario selecciona la pestaña "Distribuciones de Asociados" en la interfaz de reportes de Distribuciones. 2. El sistema muestra una tabla con la lista de distribuciones para los asociados del usuario.	
Resultados esperados: • El sistema muestra correctamente la lista de distribuciones para los asociados.	

Tabla C.10. Prueba de aceptación # 11

Caso de prueba de aceptación	
Código: HU7_P1	**Historia de usuario:** 7
Nombre: Visualizar mis facturaciones	
Descripción: Prueba para validar la funcionalidad de ver la lista de facturaciones del usuario.	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • Deben existir facturaciones realizadas para el usuario.	
Pasos de ejecución: 1. El usuario selecciona la pestaña "Mis Facturaciones" en la interfaz de reportes de facturaciones. 2. El sistema muestra una tabla con la lista de facturaciones del usuario.	
Resultados esperados: • El sistema muestra correctamente la lista de facturaciones realizadas por el usuario.	

Tabla C.11. Prueba de aceptación # 12

Caso de prueba de aceptación	
Código: HU7_P2	**Historia de usuario:** 7
Nombre: Visualizar facturaciones de familiares	
Descripción: Prueba para validar la funcionalidad de ver la lista de facturaciones para los familiares del usuario.	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • Deben existir facturaciones realizadas para los familiares del usuario.	
Pasos de ejecución: 1. El usuario selecciona la pestaña "Facturaciones de Familiares" en la interfaz de reportes de facturaciones. 2. El sistema muestra una tabla con la lista de facturaciones realizadas para los familiares del usuario.	
Resultados esperados: • El sistema muestra correctamente la lista de facturaciones realizadas para los familiares.	

Tabla C.12. Prueba de aceptación # 13

Caso de prueba de aceptación	
Código: HU7_P3	**Historia de usuario:** 7
Nombre: Visualizar facturaciones de asociados	
Descripción: Prueba para validar la funcionalidad de ver la lista de facturaciones realizadas para los asociados del usuario.	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • Deben existir facturaciones realizadas para los asociados del usuario.	
Pasos de ejecución: 1. El usuario selecciona la pestaña "Facturaciones de Asociados" en la interfaz de reportes de facturaciones. 2. El sistema muestra una tabla con la lista de facturaciones realizadas para los asociados del usuario.	
Resultados esperados: • El sistema muestra correctamente la lista de facturaciones realizadas para los asociados.	

Tabla C.13. Prueba de aceptación # 14

Caso de prueba de aceptación	
Código: HU8_P1	**Historia de usuario:** 8
Nombre: Visualizar mis accesos	
Descripción: Prueba para validar la funcionalidad de ver la lista de accesos realizados por el usuario.	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • Deben existir accesos registrados para el usuario.	
Pasos de ejecución: 1. El usuario selecciona la pestaña “Mis Accesos” en la interfaz de reportes de accesos. 2. El sistema muestra una tabla con la lista de accesos realizados por el usuario.	
Resultados esperados: • El sistema muestra correctamente la lista de accesos realizados por el usuario.	

Tabla C.14. Prueba de aceptación # 15

Caso de prueba de aceptación	
Código: HU8_P2	**Historia de usuario:** 8
Nombre: Visualizar accesos de familiares	
Descripción: Prueba para validar la funcionalidad de ver la lista de accesos realizados por los familiares del usuario.	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • Deben existir accesos registrados para los familiares del usuario.	
Pasos de ejecución: 1. El usuario selecciona la pestaña “Accesos de Familiares” en la interfaz de reportes de accesos. 2. El sistema muestra una tabla con la lista de accesos realizados por los familiares del usuario.	
Resultados esperados: • El sistema muestra correctamente la lista de accesos realizados por los familiares.	

Tabla C.15. Prueba de aceptación # 16

Caso de prueba de aceptación	
Código: HU8_P3	**Historia de usuario:** 8
Nombre: Visualizar accesos de asociados	
Descripción: Prueba para validar la funcionalidad de ver la lista de accesos realizados por los asociados del usuario.	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • Deben existir accesos registrados para los asociados del usuario.	
Pasos de ejecución: 1. El usuario selecciona la pestaña “Accesos de Asociados” en la interfaz de reportes accesos. 2. El sistema muestra una tabla con la lista de accesos realizados por los asociados del usuario.	
Resultados esperados: • El sistema muestra correctamente la lista de accesos realizados por los asociados.	

Tabla C.16. Prueba de aceptación # 17

Caso de prueba de aceptación	
Código: HU9_P1	**Historia de usuario:** 9
Nombre: Visualizar tarjetas del usuario	
Descripción: Prueba para validar la funcionalidad de ver la lista de tarjetas asociadas al usuario.	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • Deben existir tarjetas asociadas al usuario.	
Pasos de ejecución: 1. El usuario accede a la sección de “Tarjetas” en la interfaz del sistema. 2. El sistema muestra una tabla con la lista de tarjetas asociadas al usuario.	
Resultados esperados: • El sistema muestra correctamente la lista de tarjetas asociadas al usuario.	

Tabla C.17. Prueba de aceptación # 18

Caso de prueba de aceptación	
Código: HU10_P1	**Historia de usuario:** 10
Nombre: Visualizar lista de platos	
Descripción: Prueba para validar la funcionalidad de ver la lista de platos activos registrados por la institución.	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • Deben existir platos registrados en el sistema.	
Pasos de ejecución: 1. El usuario accede a la sección de "Reporte de Platos" en la interfaz del sistema. 2. El sistema muestra una tabla con la lista de platos activos registrados por la institución.	
Resultados esperados: • El sistema muestra correctamente la lista de platos activos registrados por la institución.	

Tabla C.18. Prueba de aceptación # 19

Caso de prueba de aceptación	
Código: HU10_P2	**Historia de usuario:** 10
Nombre: Buscar y filtrar platos	
Descripción: Prueba para validar la funcionalidad de buscar y filtrar los platos en la lista por unidad de medida, clasificación de plato y composición.	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • Deben existir platos registrados en el sistema.	
Pasos de ejecución: 1. El usuario accede a la sección de "Platos". 2. El sistema muestra una lista de platos activos. 3. El usuario utiliza los campos de búsqueda y filtros para filtrar los platos por unidad de medida, clasificación o composición.	
Resultados esperados: • El sistema filtra y muestra correctamente los platos que coinciden con los criterios de búsqueda y filtrado seleccionados por el usuario.	

Tabla C.19. Prueba de aceptación # 20

Caso de prueba de aceptación	
Código: HU11_P1	**Historia de usuario:** 11
Nombre: Visualizar lista de menús	
Descripción: Prueba para validar la funcionalidad de ver la lista de menús activos registrados por la institución.	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • Deben existir menús registrados en el sistema.	
Pasos de ejecución: 1. El usuario accede a la sección de "Reporte de Menús" en la interfaz del sistema. 2. El sistema muestra una tabla con la lista de menús activos registrados por la institución para el día actual.	
Resultados esperados: • El sistema muestra correctamente la lista de menús activos registrados por la institución.	

Tabla C.20. Prueba de aceptación # 21

Caso de prueba de aceptación	
Código: HU11_P2	**Historia de usuario:** 11
Nombre: Filtrar menús por intervalo de fechas	
Descripción: Prueba para validar la funcionalidad de filtrar los menús activos en la lista por un intervalo de fechas.	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • Deben existir menús registrados en el sistema.	
Pasos de ejecución: 1. El usuario accede a la sección de "Reporte de Menús". 2. El sistema muestra una lista de menús activos. 3. El usuario utiliza el filtro de intervalo de fechas para seleccionar un rango específico.	
Resultados esperados: • El sistema filtra y muestra correctamente los menús que coinciden con el intervalo de fechas seleccionado por el usuario.	

Tabla C.21. Prueba de aceptación # 22

Caso de prueba de aceptación	
Código: HU12_P1	**Historia de usuario:** 12
Nombre: Visualizar menús próximos en la portada del módulo.	
Descripción: Prueba para validar la funcionalidad de mostrar los menús próximos a partir de la fecha actual.	
Condiciones de ejecución: • El usuario debe estar autenticado previamente. • Deben existir menús próximos en el sistema.	
Pasos de ejecución: 1. El usuario accede al módulo de reservación. 2. El sistema muestra una lista con los menús próximos a partir de la fecha actual.	
Resultados esperados: • El sistema muestra correctamente los menús próximos que inician después de la fecha actual.	

Tabla C.22. Prueba de aceptación # 23

Caso de prueba de aceptación	
Código: HU13_P1	**Historia de usuario:** 13
Nombre: Visualización de formulario de reserva por categorías	
Descripción: Prueba para validar que el formulario de reserva por categorías se muestra correctamente.	
Condiciones de ejecución: • El usuario debe estar autenticado en el sistema. • La institución debe tener habilitado el proceso de reservación.	
Pasos de ejecución: 1. El usuario accede a la sección de “Reserva por Categorías” en la aplicación. 2. El sistema muestra el formulario con los campos de “Categoría”, “Categoría de Residente”, y “Fecha del Menú”.	
Resultados esperados: • El formulario de reserva por categorías se muestra correctamente con los campos requeridos y el botón de “Reservar”.	

Tabla C.23. Prueba de aceptación # 24

Caso de prueba de aceptación	
Código: HU13_P2	**Historia de usuario:** 13
Nombre: Filtrado de personas según categoría y fecha	
Descripción: Prueba para validar que el sistema filtra correctamente las personas según la categoría, categoría de residente y fecha seleccionada.	
Condiciones de ejecución: • El usuario debe estar autenticado en el sistema. • La institución debe tener habilitado el proceso de reservación. • Debe haber un menú disponible en la fecha seleccionada.	
Pasos de ejecución: 1. El usuario selecciona una “Categoría”, “Categoría de Residente” y “Fecha del Menú” en el formulario. 2. El sistema muestra una lista de personas que cumplen con los criterios seleccionados y que no tienen reservas para esa fecha.	
Resultados esperados: • La lista de personas que cumplen con las categorías y no tienen reservas para la fecha seleccionada se muestra correctamente.	

Tabla C.24. Prueba de aceptación # 25

Caso de prueba de aceptación	
Código: HU13_P3	**Historia de usuario:** 13
Nombre: Realizar reserva para personas seleccionadas por categoría	
Descripción: Prueba para validar que el sistema permite realizar una reserva para las personas seleccionadas de la lista filtrada.	
Condiciones de ejecución: • El usuario debe estar autenticado en el sistema. • Debe haber personas en la lista que cumplen con las categorías seleccionadas y que no tienen reservas previas. • Debe haber un menú disponible en la fecha seleccionada.	
Pasos de ejecución: 1. El usuario presiona el botón “Reservar”.	
Resultados esperados: • La reserva se realiza con éxito para las personas seleccionadas. • El sistema muestra un mensaje de confirmación indicando que la reserva fue exitosa.	

Tabla C.25. Prueba de aceptación # 26

Caso de prueba de aceptación	
Código: HU13_P4	**Historia de usuario:** 13
Nombre: Validación de disponibilidad de menú para la fecha seleccionada	
Descripción: Prueba para validar que el sistema no permite reservas si no hay un menú disponible en la fecha seleccionada.	
Condiciones de ejecución: • El usuario debe estar autenticado en el sistema. • La institución debe tener habilitado el proceso de reservación. • No debe haber un menú disponible en la fecha seleccionada.	
Pasos de ejecución: 1. El usuario selecciona una “Categoría”, “Categoría de Residente” y “Fecha del Menú” en el formulario. 2. El sistema intenta mostrar la lista de personas que cumplen con los criterios seleccionados.	
Resultados esperados: • El sistema muestra un mensaje de error indicando que no hay un menú disponible en la fecha seleccionada. • No se permite realizar la reserva en esa fecha.	