UNIVERSIDAD DE ARTES, CIENCIAS Y COMUNICACIÓN
Facultad de Administración
Carrera de Ingeniería Informática Multimedia

Título del trabajo

Proyecto para optar al Grado Académico de Licenciado/a en Ingeniería Informática Multimedia y al Título Profesional de Ingeniero Informático Multimedia

Profesor Guía:
Felipe Montenegro

Estudiante:
Fernando Enrique Sepúlveda Campos

Santiago de Chile, abril de 2022

e Figuras

ɜumen.

La tecnología ha evolucionado permitiendo de esta manera que se logre un mejor desempeño a través de nuevas herramientas, las mismas que permiten la optimización de recursos humanos, además, de materia prima y el factor tiempo, siendo este la principal característica para el desarrollo de una aplicación, cabe señalar que mediante el uso de estas tareas se logrará una mejor eficacia y seguridad, evitando la redundancia de información.

Internet cambio la forma de hacer negocios, modifico la dinámica del comercio y abrió un gran número de nuevas posibilidades de crecimiento. Es indudable que Internet influye cada vez más en las actividades de las personas. En este contexto, surge este proyecto de investigación con el objetivo de analizar la factibilidad de comercializar platos de comida por internet.

En primera instancia se realiza un proceso de contextualización en el que se describe el desarrollo del comercio electrónico.

Se evalúa y analiza cada uno de los aspectos del modelo de negocio de comercialización online con el objeto de cumplir con los estándares de calidad del comercio electrónico. Finalmente se realiza un estudio financiero en el que se evalúa la viabilidad del negocio en el ámbito financiero.

ɔducción.

El objetivo va a ser el desarrollo de una tienda virtual enfocada a la venta de platos de comida. La aplicación para desarrollar ofrecerá la funcionalidad que disponen las tienda en la red, dispondrá de un catálogo de platos donde se podrá visualizar su ingrediente y por supuesto tendremos la opción de poder comprarlos. Dependiendo del tipo de usuario se dispondrá de unas opciones o de otras.

El usuario anónimo será el que menos funcionalidades tenga, podrá navegar por nuestra web, ver los platos y añadirlos al carro, pero no podrá efectuar la finalización de la compra hasta que no se registre o se identifique, una vez realizada cualquiera de estas operaciones se convertirá en usuario registrado el cual ya podrá finalizar la compra.

El usuario con más privilegios será el usuario administrador, que además de disponer de las funcionalidades anteriores podrá gestionar el catálogo del restaurante, administrar clientes, revisar stock disponible.

En resumen, se trata de desarrollar una aplicación para facilitar la venta de platos de alimentos, así como ayudar a su gestión, de una forma sencilla y clara para los usuarios y el administrador de la aplicación.

Se pretende responder con una solución al problema planteado sobre la utilización centralizada de la información de la bodega del restaurante. El trabajo será desarrollado en Java donde la aplicación que consuma servicios web de una que se integran con una misma fuente de datos (base de datos). Esto será mediante un ambiente web. La programación de la aplicación será con Java, el servidor será Glassfish y la base de datos será MySQL.

La necesidad de garantizar el software de calidad ha motivado el diseño de diversas metodologías para el desarrollo y ejecución de pruebas de software. Lo anterior debido a que, en la práctica, definir los pasos para mejorar y controlar las fases de un proceso de pruebas de software y el orden que estas se implementan es, en general, una tarea difícil.

Por lo anterior, todo profesional que se encuentra en el área del desarrollo de software debe establecer una estrategia adecuada para probar sus productos. Dichos productos deben pasar por diferentes tipos de pruebas durante todo el ciclo de vida del software. A pesar de la diversidad de pruebas a las que un software es sometido es muy probable que al ser entregado contenga diferentes anomalías, unas más graves que otras, que pueden pasar desapercibidas durante el ciclo de desarrollo del software. Por tal motivo es importante tratar de encontrar el mayor número de errores antes de que el producto sea implementado. Existe una gran variedad de herramientas y técnicas que facilitan a los probadores de software la ardua tarea de encontrar anomalías.

3 Planteamiento del Problema.

Nuestro cliente es dueño de un Restaurant, local con gran trayectoria y tradición en la zona, todos sus procesos se realizan con boletas en papel, a tal punto que jamás imagino en cambiar o renovar su actual sistema, persistiendo de esta forma hasta la actualidad, pero con el caso covid-19 pandemia mundial, tuvo que cerrar el local durante un tiempo y ya cuando las cosas en la zona se comienzan a tranquilizar contando con más flexibilidad para al retorno de los locales comerciales de comida, debido al miedo reinante en realizar pagos en efectivo, se ve en la necesidad de adquirir los dispositivos para permitir el pago con red compra y quiere aprovechar esta instancia para actualizar su actual sistema por uno más tecnológico, que le permita, entre otras cosas, un fácil crecimiento futuro en cuanto a sucursales y que le facilite en gran medida la incorporación de trabajadores más jóvenes. Además de querer mostrar a su clientela una imagen moderna y actual acorde a los tiempos.

Es en base a esta realidad que nuestro cliente se acerca a nosotros solicitando en primera medida un sistema base que permita realizar todo el control de stock y selección platos en su local, que visualice los platos que se ofrecen en el menú mostrando la información de los ingredientes para cada uno de ellos, de modo que el cocinero al ir seleccionándolos vaya automáticamente realizando el descuento al stock.

Por todo lo expuesto es que inicialmente se piensa construir una aplicación web responsiva que cuente con un acceso a la data mediante web service, de modo que a futuro pueda ser accedida desde otras aplicaciones fácilmente además de permitirle el acceso desde diferentes dispositivos, lo cual para nuestro cliente es un aspecto clave.

4 Propósitos

La especificación de requisitos nos servirá de base para desarrollar nuestra aplicación Web en forma de una tienda virtual de venta de platos de comida, detallar los requisitos de diseño de interfaz, contenidos y funcionalidad de esta. Así como la especificación de los objetivos que queremos conseguir.

5 Objetivo Principal.

Desarrollar una aplicación, la cual desplegará un menú con platos diferentes. Al seleccionar cada plato deberá tener una lista de cada ingrediente que se utilizará.

Al momento de seleccionar un ingrediente, este será descontado en forma automática de la base de datos. En caso de tener stock “0”, se deberá desplegar un mensaje que indique la no existencia de este ingrediente, la cual entregará el stock real de cada ingrediente para la preparación de los diferentes platos. Deberá contener un Menú mantenedor del sistema (Agregar – Modificar – Eliminar).

Desarrolla una base de datos en SQL que contenga los campos necesarios para que la aplicación funcione. Esta base de datos deberá estar modelada en 3ra forma normal.

6 Justificación.

La finalidad de realizar la Tienda Virtual es para incrementar ventas, publicidad y lograr extenderse fuera de la ciudad todo esto de una manera más económica que ayude al posicionamiento en el mercado del restaurante de comida ya que se han detectado varias falencias que conllevan a causas como:

Falta de direccionamiento estratégico, desorganización, la carencia de un departamento de ventas, el deficiente presupuesto de publicidad, las promociones, ausencia de un plan estratégico de marketing y la falta de orientación de atención al cliente.

Indudablemente este proyecto servirá de aporte e incentivo al dueño de restaurante, ya que se pretende establecer las condiciones básicas necesarias que favorezcan al desarrollo de esta actividad y con ella el progreso económico.

Desde el punto de vista económico es posible realizar la investigación del problema planteado, porque los valores que se invertirán en el diseño y mantenimiento de la Tienda Virtual no son muy elevados en comparación a los beneficios y comodidades que recibirá el restaurante y los clientes de este.

7 Marco Teórico.

7.1 Bases Teóricas

Antecedentes: En base a la tecnología de alto nivel es posible crear un sistema base que permita realizar todo el control de stock del local, que visualice los platos que se ofrecen en el menú mostrando la información de los ingredientes para cada uno de ellos, de modo que el cocinero al ir seleccionándolos vaya automáticamente realizando el descuento al stock.

Marco Conceptual

En este capítulo se describirán las tecnologías y conceptos que fueron relevantes para el desarrollo de la solución. De manera breve, estos conceptos y tecnologías son:

- JavaEE: Java Enterprise Edition, Java EE en adelante, es un conjunto de estándares de tecnologías dedicadas al desarrollo de Java del lado del servidor. La plataforma Java EE consta de un conjunto de servicios, API y protocolos que proporcionan la funcionalidad necesaria para desarrollar aplicaciones basadas en web de varios niveles. Es decir, desarrollaremos aplicaciones empresariales distribuidas, con arquitecturas multicapa, escritas en Java y que se ejecutan en un servidor de aplicaciones.
- SQL: La programación SQL permite interactuar con una base de datos. El lenguaje de consulta estructurado (SQL) es el lenguaje de base de datos más implementado y valioso para cualquier persona involucrada en la programación informática o que usa bases de datos para recopilar y organizar información
- Glassfish: es un servidor de aplicaciones de software libre desarrollado por Sun Microsystems, compañía adquirida por Oracle Corporación, que implementa las tecnologías definidas en la plataforma Java EE y permite ejecutar aplicaciones que siguen esta especificación

- Base de datos: Una base de datos es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso. En este sentido; una biblioteca puede considerarse una base de datos compuesta en su mayoría por documentos y textos impresos en papel e indexados para su consulta.
- Ambiente web: En primer lugar, el entorno web es una presentación de información que utiliza combinaciones de; texto, imagen, animación, sonido, video. En el entorno web podemos distinguir diferentes tipos de multimedia.

Bases Teóricas

Definición

Según Luján-Mora, Sergio (2002) afirman lo siguiente: Aunque los inicios de Internet se remontan a los años sesenta, no ha sido hasta los años noventa cuando, gracias a la Web, se ha extendido su uso por todo el mundo. En pocos años la Web ha evolucionado enormemente: se ha pasado de páginas sencillas, con pocas imágenes y contenidos estáticos a páginas complejas con contenidos dinámicos que provienen de bases de datos, lo que permite la creación de "aplicaciones web". De forma breve, una aplicación web se puede definir como una aplicación en la cual un usuario por medio de un navegador realiza peticiones a una aplicación remota accesible a través de Internet (o a través de una intranet) y que recibe una respuesta que se muestra en el propio navegador. El contenido de este libro se estructura en dos partes. En la primera parte del libro se tratan temas introductorios a la programación de aplicaciones web: un breve repaso de la historia de Internet y de la Web, características de las arquitecturas cliente/servidor, el concepto de aplicación web y la estructura de un sitio web tanto a nivel físico como lógico. La segunda parte del libro se centra en la programación de la parte cliente de las aplicaciones web. En el "mundo Internet" existen muchas tecnologías que se pueden emplear para programar los clientes web, como ActiveX, applet, Flash, VRML, etc., pero sólo dos son las tecnologías más

extendidas y se pueden considerar "el estándar": HTML y JavaScript. Este libro se centra en esas dos tecnologías y presta una especial atención a la creación de formularios, la base para cualquier aplicación web.

Antecedentes de Investigación

Según Menich (2017) datos de [4], 20% de los chilenos visita un restaurant al menos una vez al mes, porcentaje que se pronostica aumentará, por lo que cada vez los problemas mencionados se harán más recurrentes. Cabe mencionar que solo en Santiago, existe un restaurante por cada 825 habitantes [5], razón que se ve multiplicada en varias otras partes de Latinoamérica y el mundo, por lo que es un problema de alcance global. Por otro lado, la alta cantidad de smartphones en el mundo, los que se pronostican llegarán a 1.84 billones para el año 2020 [6], junto a el crecimiento de la tecnología con respecto a los pagos, hacen que hoy el pago móvil sea prácticamente una realidad. Tecnologías como Apple Pay, Samsung Pay, PayPal, y Square han impulsado esta tendencia, brindando como ventajas seguridad, rapidez y simplicidad. Según [7], se pronostican 450 millones de usuarios de pagos móviles a nivel mundial para 2017, lo que manifiesta una confianza en esta tecnología por parte del consumidor que antes no existía, por lo que esta tecnología es cada vez menos ajena al público general, por lo que existe una costumbre ya instaurada entorno a pagar mediante un smartphone para los usuarios. (pág. 2)

Mientras que Silva (2015) comenta que: El proyecto se centra en la industria gastronómica de Chile, la cual ha experimentado un sostenido crecimiento en la última década, sin embargo, la calidad de servicio no forma parte de los factores que explican este desarrollo, sino que se encuentra en niveles muy bajos, no existen procesos de evaluación estandarizados y las empresas no invierten en este sector. En adición, la principal base teórica de este trabajo se basa en las metodologías de mejoramiento y medición continua de calidad de servicio, MMEDCAL, del Programa de Innovación y Sociotecnología del Departamento de

Ingeniería Industrial de la Universidad de Chile, cuya base conceptual es la teoría del constructivismo radical, donde se postula que “conocer una realidad es construirla”. Estos sistemas permiten la expansión de conciencia de quienes proveen el servicio y de los clientes a quienes atienden, utilizando como principal herramienta la generación de conversaciones clave que faciliten un proceso transformacional de la convivencia. Lo anterior, con el foco en mejorar continuamente la calidad de servicio a través de la colaboración entre servidor y cliente. (pág. 1)

7.2 Bases Legales.

La seguridad de la información es un concepto que abarca un conjunto de normas, que, en muchas ocasiones de forma voluntaria, se adhiere una organización con el fin de mejorar los procesos y garantizar un mayor nivel de protección, minimizando y conociendo los riesgos con los que se puede encontrar.

Nacionales

Según Biblioteca Nacional afirma que Los delitos informáticos se encuentran regulados en Chile en tres normas legales: Ley N° 19.2231, que tipifica figuras penales relativas a la informática; Ley N° 17.3362, sobre Propiedad Intelectual; y la Ley N° 19.9273, que modifica el Código Penal, el Código de Procedimiento Penal y el Código Procesal Penal en materia de delitos de pornografía infantil. (pag.2).

Artículo 2°. - La presente ley ampara los derechos de todos los autores, artistas intérpretes o ejecutantes, productores de fonogramas y organismos de radiodifusión chilenos y de los extranjeros domiciliados en Chile. Los derechos de los autores, artistas intérpretes o ejecutantes, productores de fonogramas y organismos de radiodifusión extranjeros no domiciliados en el país gozarán de la

protección que les sea reconocida por las convenciones internacionales que Chile suscriba y ratifique.

16) Los programas computacionales, cualquiera sea el modo o forma de expresión, como programa fuente o programa objeto, e incluso la documentación preparatoria, su descripción técnica y manuales de uso.

Internacional

La Unión Europea ha adoptado dos Directivas para la aproximación de las normas nacionales sobre protección de la intimidad en lo que se refiere al tratamiento de datos personales. El artículo 24 de la Directiva 95/46/CE obliga claramente a los Estados miembros a adoptar las medidas adecuadas para garantizar la plena aplicación de las disposiciones de esta y a determinar, en particular, las sanciones que deben aplicarse en caso de incumplimiento de las disposiciones de las leyes nacionales. Los derechos fundamentales a la intimidad y la protección de datos se incluyen, además, en la Carta de los Derechos Fundamentales de la Unión Europea.

En síntesis, los delitos informáticos, de acuerdo con el Consejo de Europa, pueden ser clasificados en los siguientes cuatro tipos:

a. Delitos contra la confidencialidad, la integridad y la disponibilidad de los datos y sistemas informáticos: Sanciona el acceso y la interceptación ilegal, interferencia de datos y sistemas y el mal uso de dispositivos.

b. Delitos de fraude informático: Falsificación y fraude computacional.

c. Delitos por su contenido: Producción, diseminación y posesión de pornografía infantil.

d. Delitos relacionados con infracciones de la propiedad intelectual y derechos afines: La amplia gama de reproducciones ilícitas, por medios informáticos, de obras protegidas por el derecho de autor.

8 Marco Metodológico.

La modalidad de la investigación será cuali-cuantitativa. Las características cualitativas de la problemática existente, se determina de acuerdo con las observaciones realizadas, tomando en cuenta los criterios y opiniones de las personas involucradas. Los aspectos cuantitativos se ratifican porque se acudirá a modelos estadísticos para la interpretación de los datos obtenidos.

8.1 Metodología Utilizada.

En este apartado vamos a abordar la fase de análisis, con ayuda de la metodología UML se desarrollará un modelo de la aplicación donde diferenciaremos a los actores que interactúan con los objetos del sistema mediante distintas relaciones.

Análisis por definición es el proceso de construcción de un modelo, o especificación detallada del problema del mundo real al que nos enfrentamos. Está desprovisto de consideraciones de diseño e implementación. Todo esto nos ayuda a crear un sistema robusto y mantenible.

La notación que se va a utilizar es la proporcionada por el estándar UML.

8.1.1 Diagramas del Proyecto.

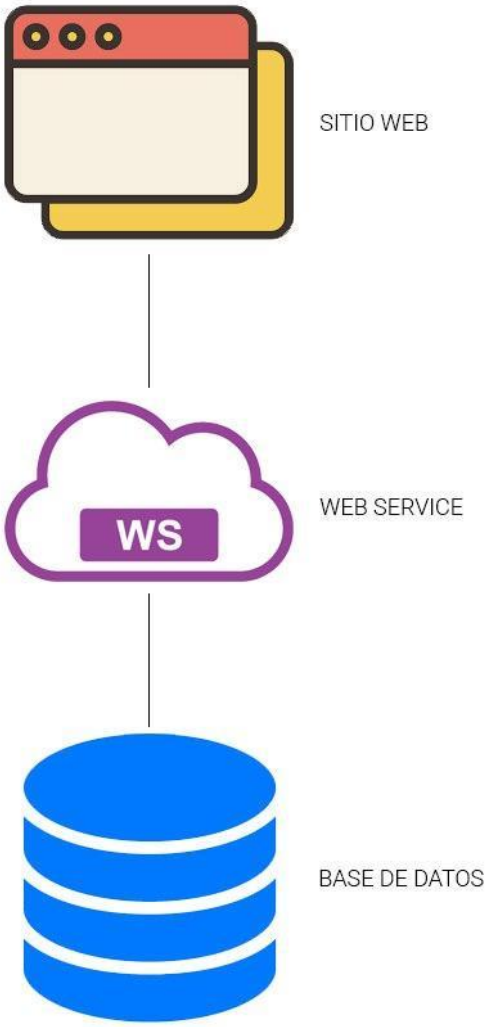


Ilustración 1, Diagramas de Despliegue

Desarrollando una aplicación del tipo webservice

API REST de la Tienda Virtual.

Para dar solución al problema planteado de calcular el valor final de un plato seleccionado en la moneda del cliente, se desarrolló un API REST que al enviar ciertos parámetros retorna una respuesta en formato JSON, la cual puede ser procesada y mostrada por la aplicación.

Para lo cual se creó la tabla “Platos” en la base de datos y se agregó un atributo llamado “Valor”; que corresponde al valor del plato en pesos chilenos. Se desarrolló un API REST con Slim en PHP, con los siguientes end-points para ser consumidos:

Tipo	End Points	Descripción	Parámetros
GET	https://cocina.dhenriquez.dev/platos	Se retornan todos los platos que existen.	
GET	https://cocina.dhenriquez.dev/platos/{ID}	Se retorna plato con sus ingredientes según ID enviado.	int ID
POST	https://cocina.dhenriquez.dev/platos/add	Se crea Plato, si no existe uno con el mismo nombre.	string nombre string imagen string receta int valor
POST	https://cocina.dhenriquez.dev/platos/update	Se actualiza plato según ID enviado.	int id string nombre string imagen string receta int valor
GET	https://cocina.dhenriquez.dev/platos/delete/{ID}	Se elimina plato según ID enviado.	int ID
GET	https://cocina.dhenriquez.dev/ingredientes	Se retornan todos los ingredientes que existan.	

GET	https://cocina.dhenriquez.dev/ingredientes/{ID}	Se retorna ingrediente según ID enviado.	int ID
POST	https://cocina.dhenriquez.dev/ingredientes/add	Se crea un ingrediente nuevo en el caso de no existir.	string nombre int stock string imagen
POST	https://cocina.dhenriquez.dev/ingredientes/update	Se actualiza ingrediente según ID enviado.	int id string nombre int stock string imagen
GET	https://cocina.dhenriquez.dev/ingredientes/delete/{ID}	Se elimina ingrediente según ID enviado.	int ID
POST	https://cocina.dhenriquez.dev/plato-ingrediente/add	Se crea la asociación entre un plato y un ingrediente con la cantidad requerida, en caso de existir se actualiza la cantidad.	int id_plato int id_ingrediente int cantidad
GET	https://cocina.dhenriquez.dev/plato-ingrediente/delete/{ID}	Se elimina relación de platos e ingredientes según ID de plato.	int ID
GET	https://cocina.dhenriquez.dev/platos/pagar/{ID}	Se retorna nuevo valor de plato calculado según moneda enviada e ID de plato enviado. Monedas permitidas USD (Dólar) y EUR (Euro).	int ID string moneda

Ilustración 2, Tabla de Servicios Web.

Por lo tanto, toda la aplicación puede ser desarrollada a través de esta API REST, en el caso particular para calcular el valor a pagar de un plato, se realizó un consumo de servicio desde la API de SBIF.cl con el valor del Dólar y Euro al día.

Para comprobar el funcionamiento de esta API se utilizó la aplicación Postman. Por ejemplo, si necesitamos saber el valor del plato con id 1 en dólares debemos realizar la siguiente consulta:

https://cocina.dhenriquez.dev/platos/pagar/1?moneda=usd

En la imagen podemos ver que el plato con id 1 tiene un valor de $8009, el cambio será a USD que tiene un valor de $805,75 y la conversión del plato a USD sería de 9,9US.

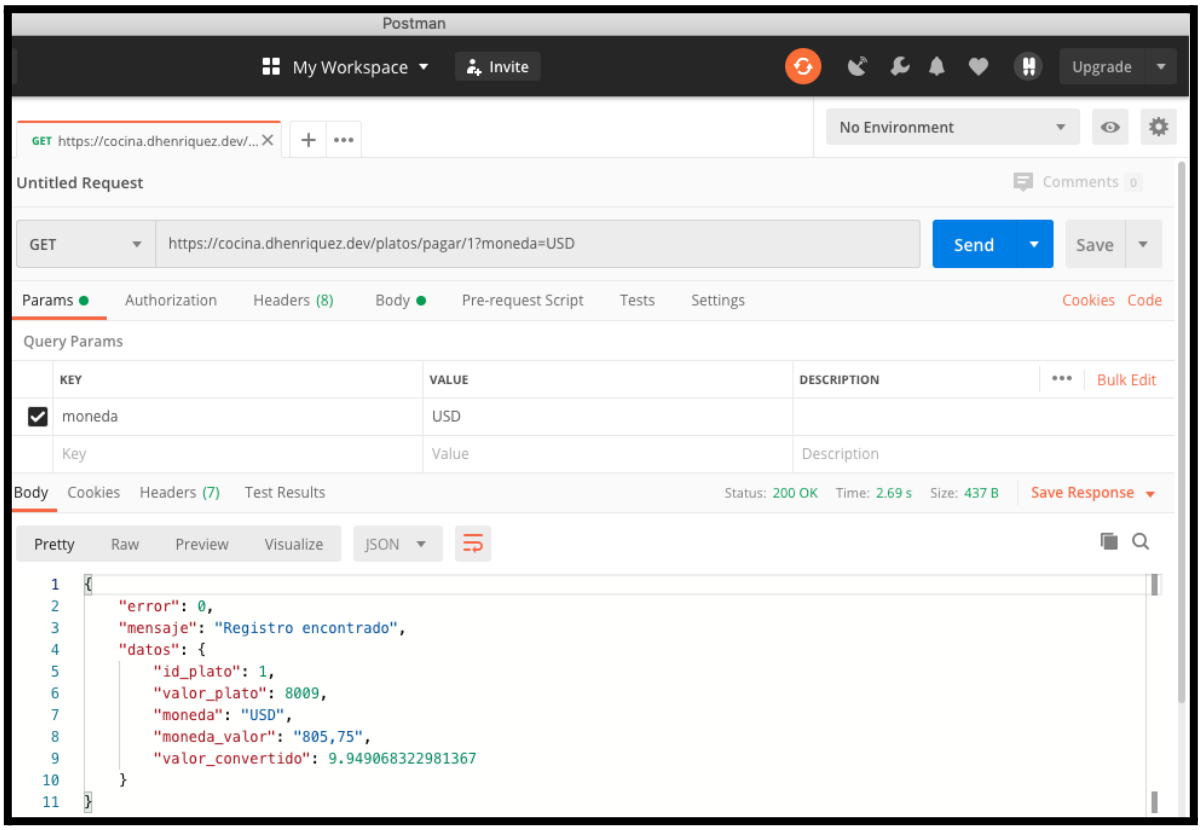


Ilustración 3, Consumo de Servicio Web.

8.1.2 Diagramas de Caso de Uso.

8.1.3 Diagrama de Secuencias.

Un diagrama de secuencia muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada método de la clase. El diagrama de secuencia contiene detalles de implementación del escenario, incluyendo los objetos y clases que se usan para implementar el escenario, y mensajes intercambiados entre los objetos.

Típicamente se examina la descripción de un caso de uso para determinar qué objetos son necesarios para la implementación del escenario. Si se dispone de la descripción de cada caso de uso como una secuencia de varios pasos, entonces se puede "caminar sobre" esos pasos para descubrir qué objetos son necesarios para que se puedan seguir los pasos.

Un diagrama de secuencia muestra los objetos que intervienen, y los mensajes pasados entre los objetos.

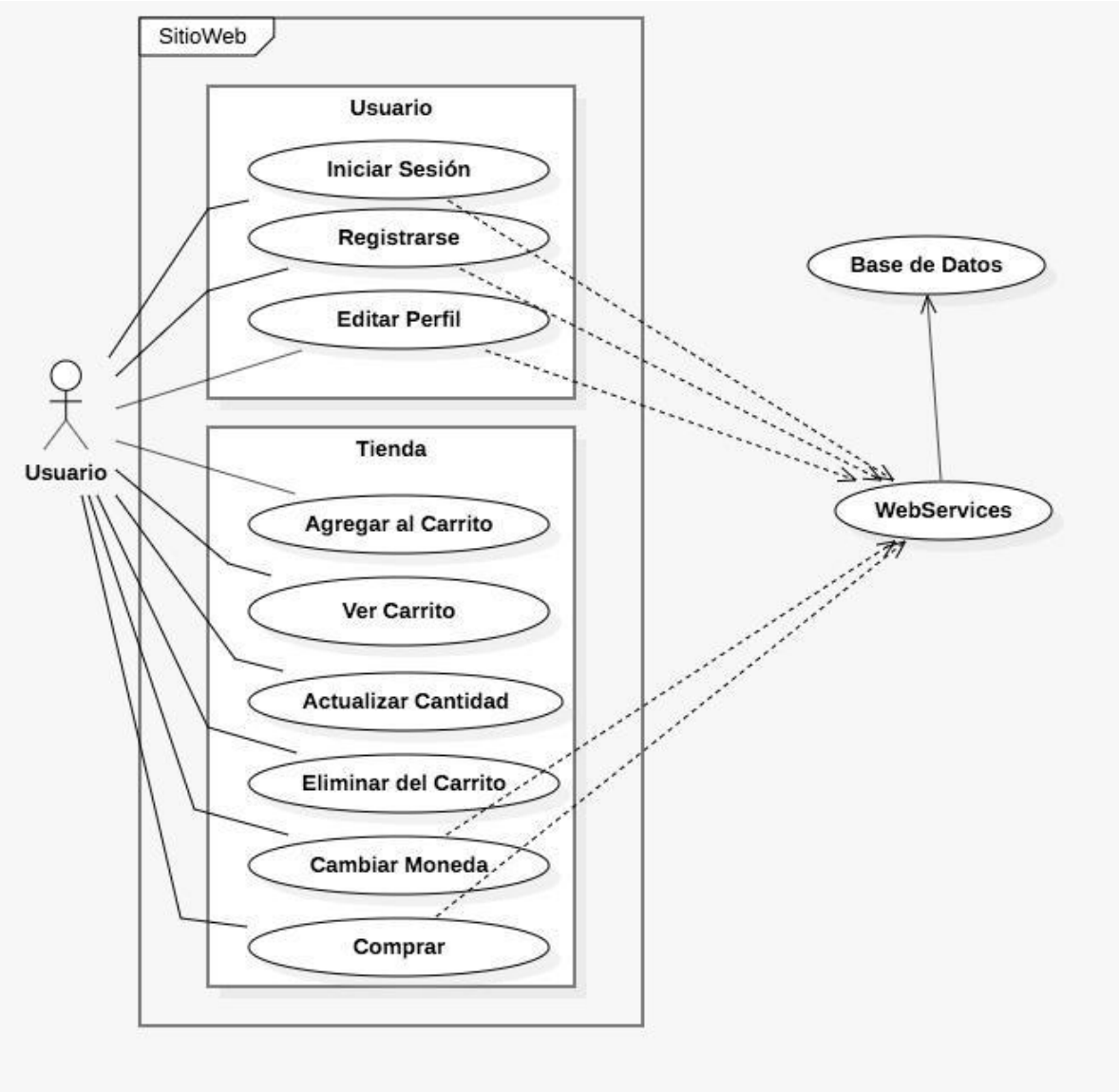


Ilustración 4 Diagramas de Caso de Uso

Escenario: Agregar Plato al Carro.

Una vez seleccionados los platos a comprar, se confirma que se quiere realizar la compra, se guardan los datos del pedido en nuestra base de datos se envía un mensaje de confirmación o en caso contrario un mensaje de advertencia.

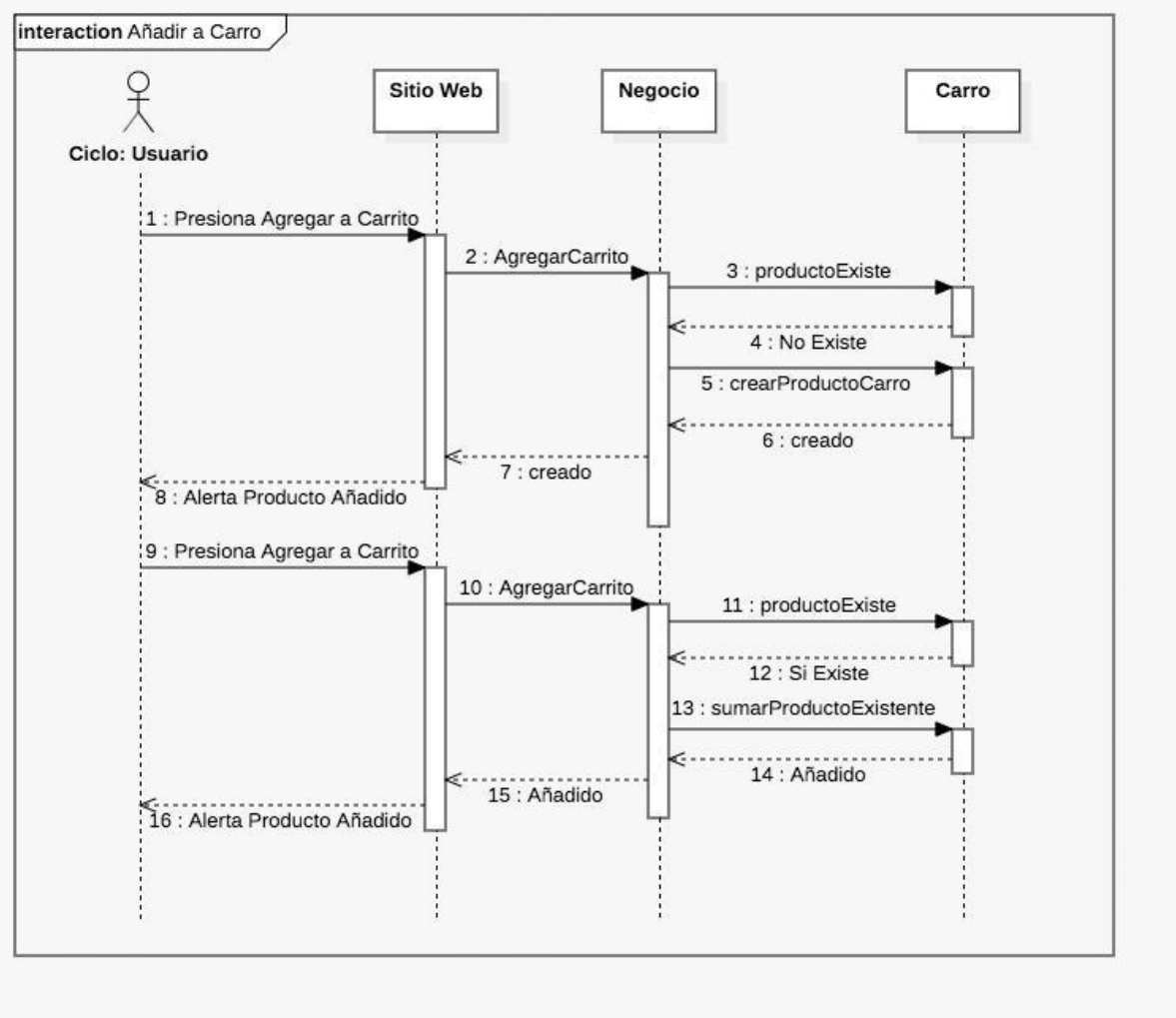


Ilustración 5 Diagrama de secuencia de Añadir al Carro

Escenario: Registrar.

Desde esta opción, el usuario no registrado podrá darse de alta en la base de datos de nuestra aplicación.

Inicialmente se nos pedirá que rellenemos un formulario con los datos personales, así como nuestro identificador y contraseña. Se comprobará que el identificador elegido no esté dado de alta por otro usuario en nuestra base de datos, si ya existe mostraremos un mensaje advirtiéndolo, en caso contrario se efectuará el alta, con su respectiva pantalla de confirmación.

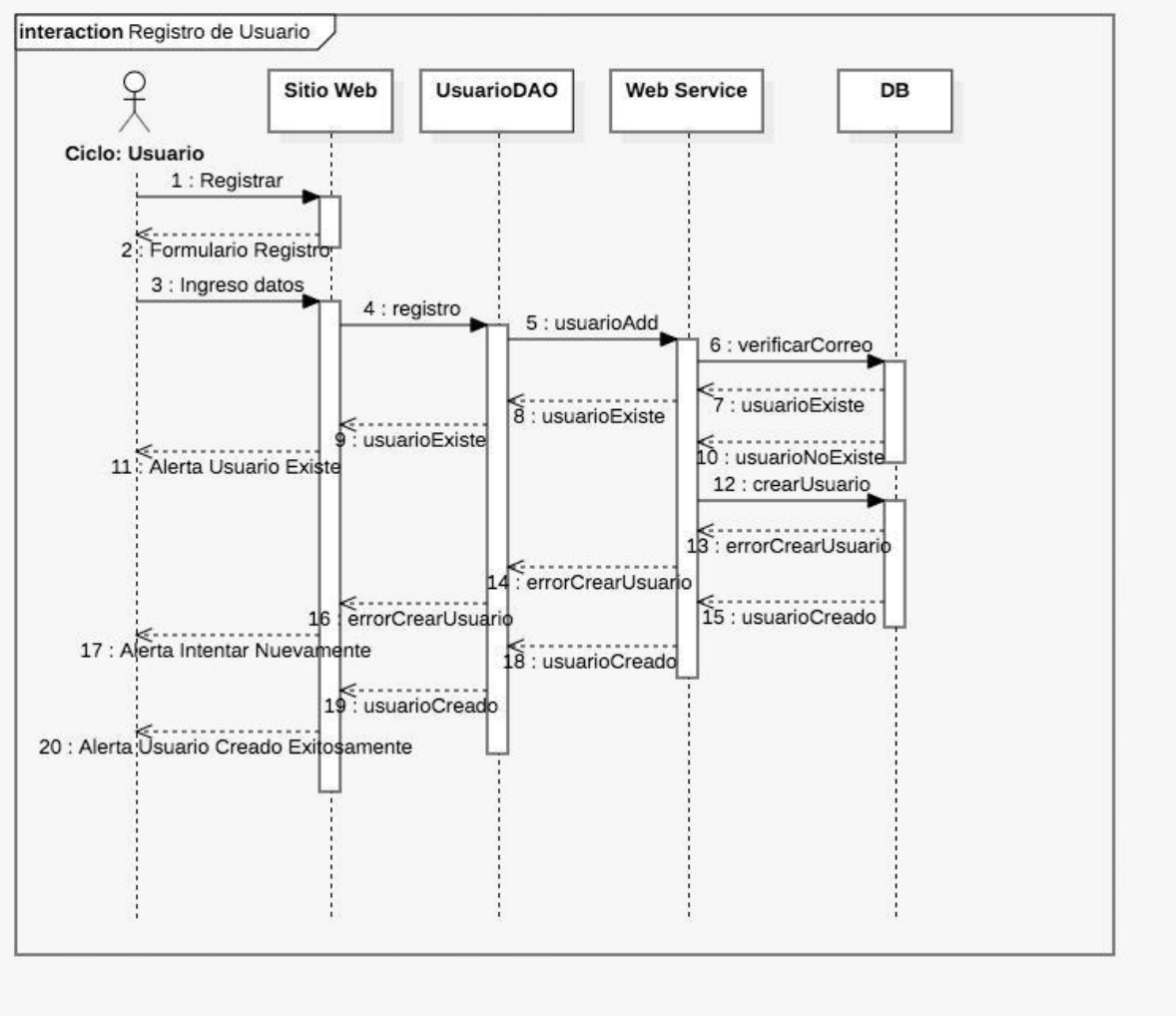


Ilustración 6 Diagrama de secuencia de Registrar

8.2 Metodología de Desarrollo de Software.

8.2.1 Desarrollo de Pantallas.

Vistas

Las vistas de nuestro sistema fueron creadas teniendo como foco principal al usuario final, pues es este quien finalmente debe estar satisfecho con la interacción, aplicamos conceptos de usabilidad y la diseñamos para que fuese simple y accesible desde cualquier dispositivo, pues es desde dispositivos móviles donde se espera recibir la mayor parte del tráfico. un usuario sin conocer de ninguna manera el sistema, es capaz de poder moverse por cada sección e interactuar con todos los botones que han sido colocados estratégicamente para que la usabilidad sea óptima. Si nos referimos a facilidad de aprendizaje, se comprueba que luego de diferentes pruebas exitosas, un usuario promedio puede interactuar cómodamente en cosa de media hora de entrar por primera vez; Dado que la aplicación es intuitiva y fácil de comprender.

A continuación, se muestran y describen las funcionalidades de algunas de las páginas más importantes:

Página principal:

En esta página se listan los platos, lo que incluye una imagen referencial, el nombre del plato, su precio en la moneda local y el botón para agregar al carrito. Una vez que se escogen todos los platillos podrás acceder a la vista del carro donde podrá aumentar o disminuir las cantidades, eliminar algún plato, cambiar, si se desea, la moneda en que se está realizando el pago y finalmente pagar.

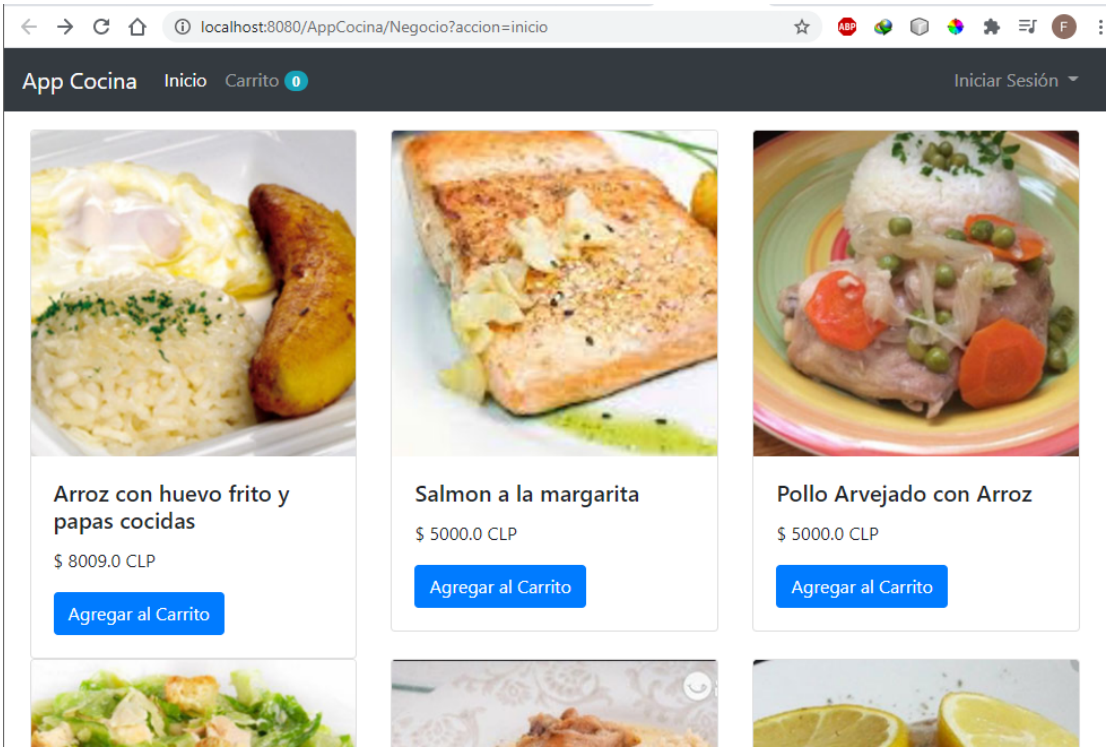


Ilustración 7 Página principal

Página Carrito:

Acá se despliegan los platos seleccionados y se dan opciones para cambiar la cantidad de cada uno o eliminarlos. También se dan otras opciones generales como cambiar el tipo de moneda entre pesos chilenos, dólares o euros, la opción para pagar solo se mostrará al momento de iniciar sesión.

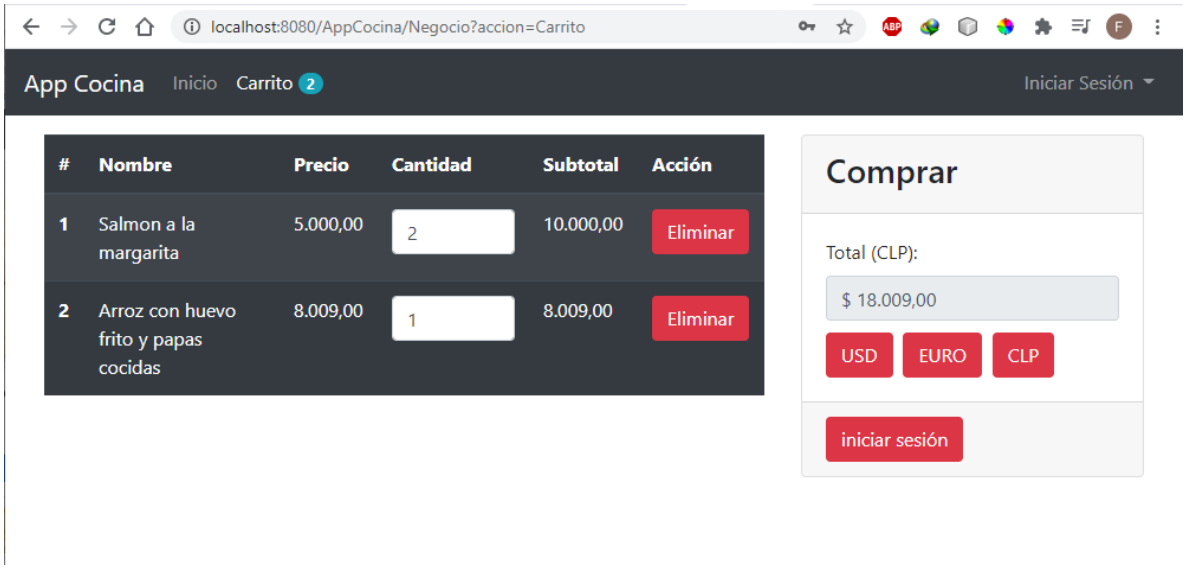


Ilustración 8 Página Carrito.

Página Stock:

Desde esta página el encargado puede gestionar el stock de los ingredientes usados para preparar los platos del restaurant, se muestra el listado, junto a

opciones de filtrado de la información para facilitar las labores asociadas, además permite editar o eliminar los ingredientes de ser necesario.

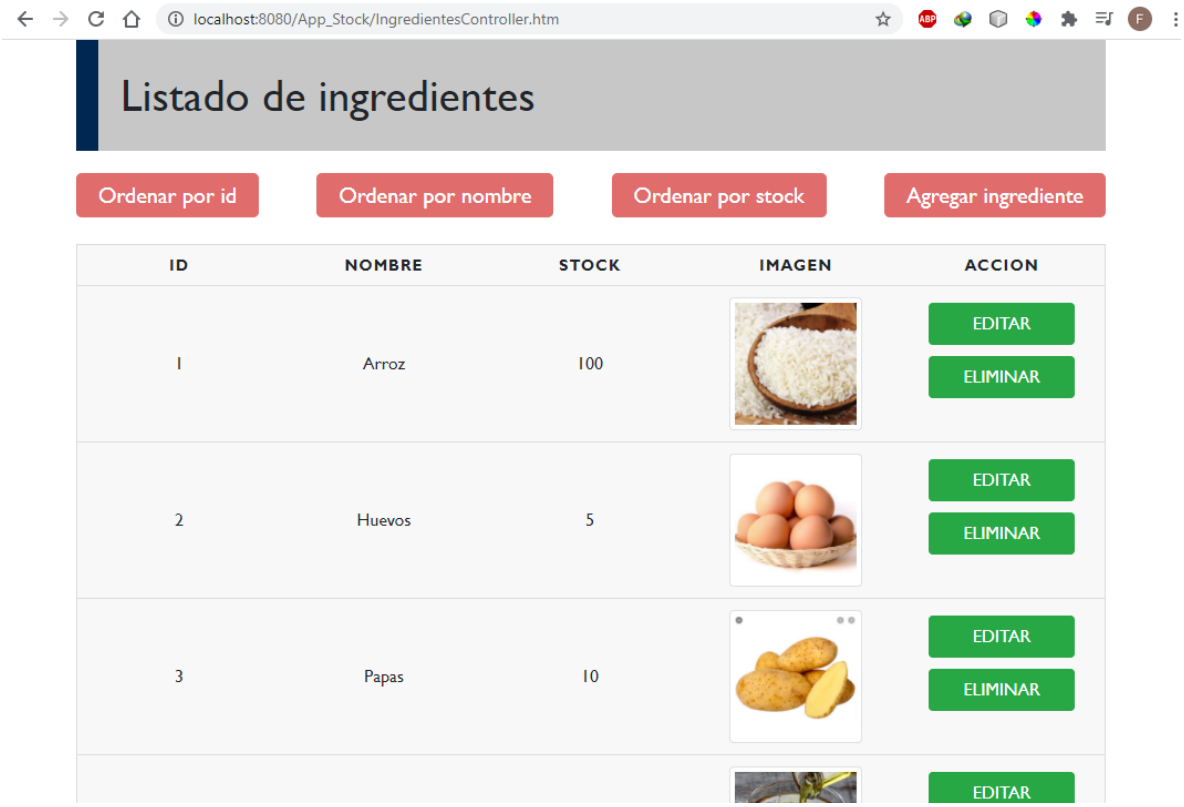


Ilustración 9 Página Stock

Página editar ingrediente:

Página utilizada por el administrador de bodega, desde esta página se pueden editar los datos de un ingrediente, todos a excepción del código son editables y obligatorios.

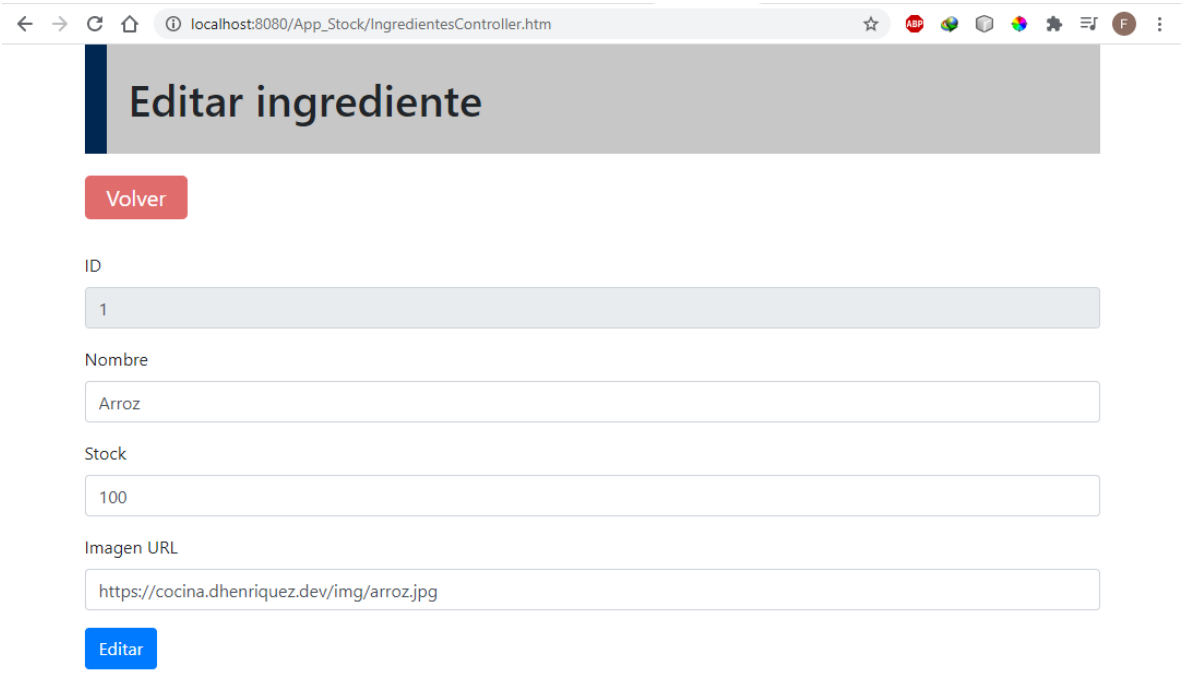


Ilustración 10 Página editar ingrediente.

Página agregar ingrediente:

Página utilizada por el administrador de bodega, permite añadir un ingrediente para ser gestionado mediante la aplicación. Todos los datos son obligatorios y el id del producto se genera automáticamente, siendo este correlativo a los demás en la base de datos.

Ilustración 11 Página agregar ingrediente

8.2.2 Modelo de Datos.

A partir de la solicitud planteada, se desarrolló el siguiente modelo de datos en 3FN como dice Irigoyen, L (2018): “los atributos que no forman parte de la clave primaria facilitan información solo acerca de la clave y no acerca de otros atributos”, por lo cual tenemos 2 tablas principales en la base de datos “Cocina”: Plato e Ingrediente. En la siguiente imagen se puede ver el modelo de Entidad Relación aplicado a nuestra base de datos.

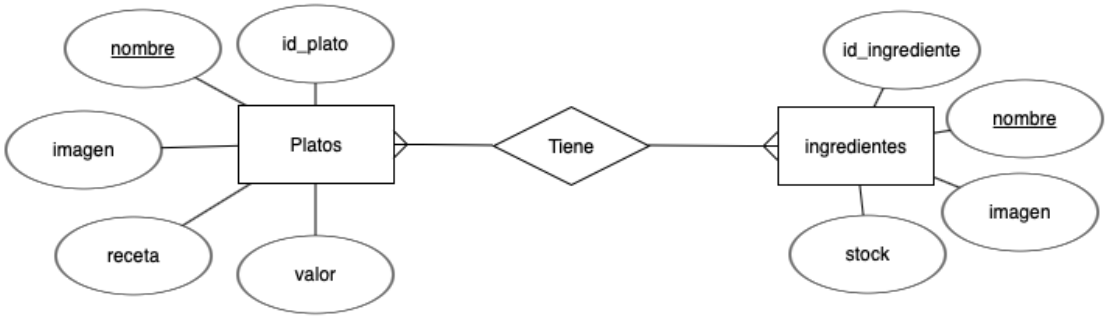


Ilustración 12, Modelo de Entidad de Relación 3FN.

La tabla Plato se compone por los siguientes atributos:

- ID_PLATO
- NOMBRE
- IMAGEN
- RECETA
- VALOR

La tabla Ingrediente se compone por los siguientes atributos:

- ID_INGREDIENTE
- NOMBRE
- STOCK
- IMAGEN

Y además contamos con una tabla en la cual existe la relación entre la tabla Plato y la tabla Ingrediente, la cual se llama Plato_Ingrediente y está compuesta con los siguientes atributos:

- ID_INGREDIENTE
- ID_PLATO
- CANTIDAD

La manera en que podemos realizar la relación entre las tablas es a partir de las llaves primarias en las tablas principales, en donde la tabla Plato tiene la llave primaria “ID_PLATO” y en la tabla Ingrediente tenemos la llave primaria “ID_INGREDIENTE”, así conformamos la relación con las llaves foráneas en la tabla Plato_Ingrediente.

La estructura final de base de datos propuesta para este trabajo se puede ver en la Figura 1.

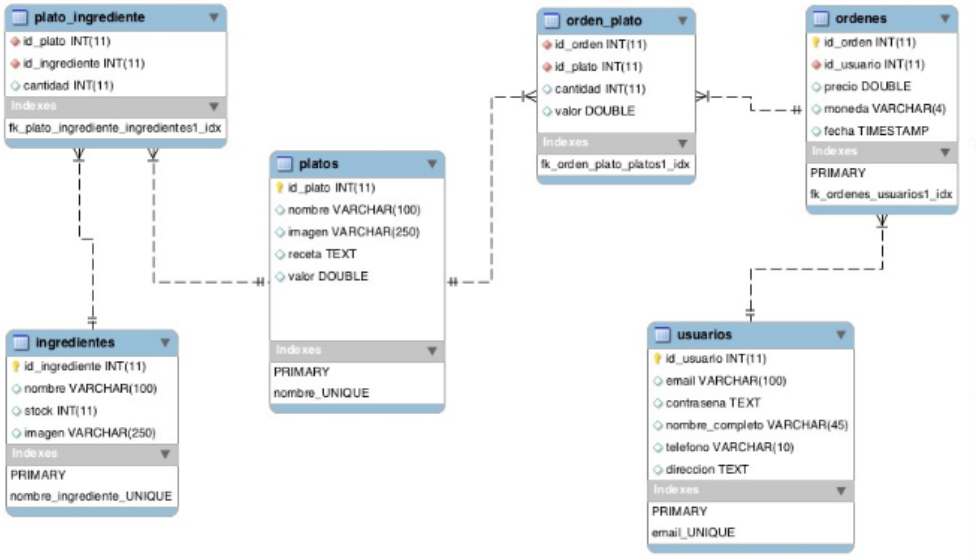


Ilustración 13, Modelo de Datos en 3FN.

8.2.3 Diccionario De Datos.

Tabla: ingredientes						
Nombre	Tipo	Nullable	PK	FK	Predeterminado	Comentarios
id_ingrediente	INT(11)	Yes	Yes	No		
nombre	VARCHAR(100)	No	No	No	NULL	
stock	INT(11)	No	No	No	NULL	
imagen	VARCHAR(250)	No	No	No	NULL	

Tabla: orden_plato						
Nombre	Tipo	Nullable	PK	FK	Predeterminado	Comentarios
id_orden_plato	INT(11)	Yes	Yes	No		
id_orden	INT(11)	Yes	No	No		
id_plato	INT(11)	Yes	No	No		
cantidad	INT(11)	No	No	No	NULL	
valor	DOUBLE	No	No	No	NULL	

Tabla: ordenes						
Nombre	Tipo	Nullable	PK	FK	Predeterminado	Comentarios
id_orden	INT(11)	Yes	Yes	No		
id_usuario	INT(11)	Yes	No	No		
precio	DOUBLE	No	No	No	NULL	
moneda	VARCHAR(4)	No	No	No	NULL	
fecha	TIMESTAMP	No	No	No	CURRENT_TIMESTAMP	

Tabla: plato_ingrediente						
Nombre	Tipo	Nullable	PK	FK	Predeterminado	Comentarios
id_plato	INT(11)	Yes	Yes	No		
id_ingrediente	INT(11)	Yes	Yes	No		
cantidad	INT(11)	No	No	No	NULL	

Tabla: platos						
Nombre	Tipo	Nullable	PK	FK	Predeterminado	Comentarios
id_plato	INT(11)	Yes	Yes	No		
nombre	VARCHAR(100)	No	No	No	NULL	
imagen	VARCHAR(250)	No	No	No	NULL	
receta	TEXT	No	No	No	NULL	
valor	DOUBLE	No	No	No	'0'	

Tabla: usuarios						
Nombre	Tipo	Nullable	PK	FK	Predeterminado	Comentarios
id_usuario	INT(11)	Yes	Yes	No		
email	VARCHAR(100)	No	No	No	NULL	
contrasena	TEXT	No	No	No	NULL	
nombre_completo	VARCHAR(45)	No	No	No	NULL	
telefono	VARCHAR(10)	No	No	No	NULL	
direccion	TEXT	No	No	No	NULL	

Ilustración 14 Diccionario De Datos.

8.2.4 Arquitectura propuesta.

Para el desarrollo de esta aplicación se utilizó una arquitectura en 3 capas, en este modelo, una aplicación se convierte en un conjunto de servicios de usuario, negocios y datos que satisface las necesidades de los procesos de negocios o procesa su soporte. Como los servicios están diseñados para el uso general y siguen lineamientos de interfaz publicados, pueden ser reutilizados y compartidos entre múltiples aplicaciones.

Es necesario puntualizar las siguientes características que traen consigo esta forma de arquitectura:

Utilización de esquemas más complejos.

Los datos y los servicios web aparecen separados.
Facilidad para separar datos de la “lógica de negocio”.
Mayor seguridad en los “datos corporativos”.
El cliente recibe los datos y la información de forma indirecta a través del servidor.

Funcionalidad de las Capas

Capa de Presentación o Interfaz de Usuario: Esta capa, está formada por los formularios y los controles que se encuentran en los formularios. Es la capa con la que interactúa el usuario.

Capa de Negocio: Está formada por las entidades, que representan objetos que van a ser manejados o utilizados por toda la aplicación. En este caso, están representados por clases y “DataTables” que se crean.

Capa de Acceso a Datos: Contiene clases que interactúan con la base de datos, estas clases altamente especializadas se encuentran en la arquitectura del sistema y permiten, utilizando los procedimientos almacenados generados, realizar todas las operaciones con la base de datos de forma transparente para la capa de negocio.

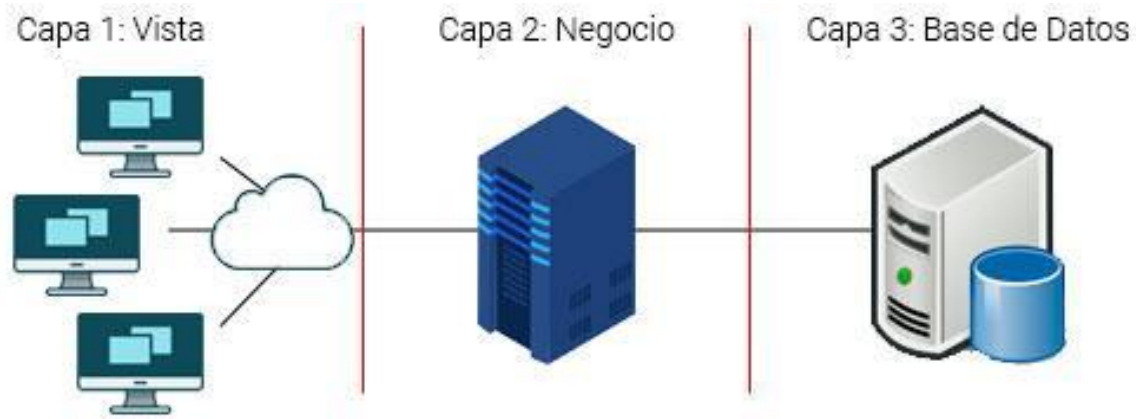


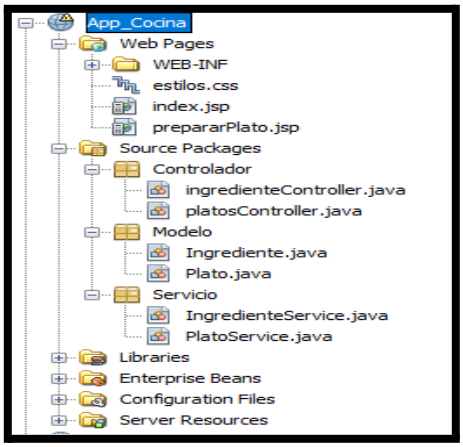
Ilustración 15, Modelo de tres capas de la aplicación.

8.2.5 Desarrollo de la Solución.

Para desarrollar las aplicaciones solicitadas, se utilizó java EE (Enterprise Edition), debido al requerimiento web dado por la empresa. Se utilizaron diferentes tecnologías en la solución, para gestionar la base de datos MySQL, se utilizó el servidor de aplicaciones GlassFish en su versión 4.1.

Para desplegar la información en la capa de presentación, aparte de HTML y CSS se utilizó JSTL + EL (Java Standard Tag Library + Expression Language) a modo de no escribir código Java en las vistas más allá del necesario para desplegar la información y esté en forma de etiquetas similares al HTML. Para la capa de negocio (Controllers) se utilizó servlets, y para gestionar la interacción con la capa de datos se utilizaron clases java. A continuación, se presenta una vista general del proyecto y su composición:

Ilustración 16, Estructura del proyecto.



Como se aprecia en la imagen, las capas MVC están bien definidas, lo que permite la correcta separación de las responsabilidades en la aplicación web.

A continuación, se muestran imágenes de cada una de las capas:

- Capa de presentación:

```
<div class="row">
    <div class="col">
        <table class="table" border="1">
            <tr>
                <th>Nombre</th>
                <th>Imagen</th>
                <th>Info</th>
                <th>Preparar</th>
            </tr>
            <c:forEach items="${requestScope.platos}" var="row" varStatus="contador">
                <tr>
                    <td>
                        <c:out value="${row.nombre}"/>
                    </td>
                    <td>
                        <div class="cont-image">
                            <img alt="imagen ${row.nombre}" class="img-thumbnail" src="${row.imagen}"/>
                        </div>
                    </td>
                    <td>
                        <c:out value="${row.receta}"/>
                    </td>
                    <td>
                        <form method="POST" action="platosController.htm">
                            <input type="hidden" name="seleccionado" value="${row.idPlato}"/>
                            <button type="submit" class="btn btn-success">Preparar</button>
                        </form>
                    </td>
                </tr>

            </c:forEach>
```

Ilustración 17, Capa de presentación.

Se aprecia la utilización de JSTL en vez de scriptlets java tal como recomienda el estándar para estos casos.

- Capa de negocio:

```
@WebServlet(name = "ingredienteController", urlPatterns = {"/ingredienteController.htm"})
public class ingredienteController extends HttpServlet {

    @EJB
    IngredienteService service;
    @EJB
    PlatoService platoService;

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        List<Modelo.Ingrediente> ingredientes = service.getIngredientes();
        request.setAttribute("ingredientes", ingredientes);
        request.getRequestDispatcher("prepararPlato.jsp").forward(request, response);
    }

    @Override
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        if (request.getParameter("descontar") != null) {
            String mensaje = service.descontarStock(Integer.parseInt(request.getParameter("descontar")));
            request.setAttribute("plato", platoService.getPlato(Integer.parseInt(request.getParameter("seleccionado"))));
            request.setAttribute("mensaje", mensaje);
            doGet(request, response);
        }
    }

}
```

Ilustración 18, Capa de negocio.

Se aprecia la utilización de Servlets, recomendado por el estándar. Estos realizan la conexión entre la capa de datos y la vista.

- Capa de datos:

```
@Entity
@Table(name = "INGREDIENTE")
@XmlRootElement
@NamedQueries({
    @NamedQuery(name = "Ingrediente.findAll", query = "SELECT i FROM Ingrediente i"),
    @NamedQuery(name = "Ingrediente.findByIdIngrediente", query = "SELECT i FROM Ingre
    @NamedQuery(name = "Ingrediente.findByNombre", query = "SELECT i FROM Ingrediente
    @NamedQuery(name = "Ingrediente.findByStock", query = "SELECT i FROM Ingrediente i
    @NamedQuery(name = "Ingrediente.findByImagen", query = "SELECT i FROM Ingrediente
public class Ingrediente implements Serializable {
    private static final long serialVersionUID = 1L;
    // @Max(value=?)  @Min(value=?)//if you know range of your decimal fields consider
    @Id
    @Basic(optional = false)
    @NotNull
    @Column(name = "ID_INGREDIENTE")
    private BigDecimal idIngrediente;
    @Basic(optional = false)
    @NotNull
    @Size(min = 1, max = 60)
    @Column(name = "NOMBRE")
```

```
@Stateless
public class IngredienteService {

    @PersistenceContext
    EntityManager em;

    public List<Ingrediente> getIngredientes() {
        return em.createNamedQuery("Ingrediente.findAll", Ingrediente.class).getResultList();
    }

    public String descontarStock(int id) {
        try {
            Ingrediente i = em.find(Ingrediente.class, new BigDecimal(id));
            if (i.getStock().equals(BigInteger.ZERO)) {
                return "Stock de "+i.getNombre()+" insuficiente";
            }
            i.setStock(i.getStock().subtract(BigInteger.ONE));
            em.merge(i);
            return "Stock de: "+i.getNombre()+" descontado!.";
        } catch (Exception e) {
            return "Error producido: " + e;
        }
    }

}
```

Ilustración 19, Capa de accesos a datos.

Conformada por la entidad, las cuales son gestionadas mediante clases java, etiquetadas para lograr la interacción con el EntityManager (ORM).

8.2.6 Funcionalidades Propuestas.

Una funcionalidad propuesta para este trabajo es la creación del mantenedor de platos dentro de la aplicación, ya que solo se solicita la implementación de un mantenedor de Ingredientes sin la posibilidad de administrar los platos y poder asociar los ingredientes correspondientes a cada uno de ellos.

8.2.7 Diagrama de Componentes.

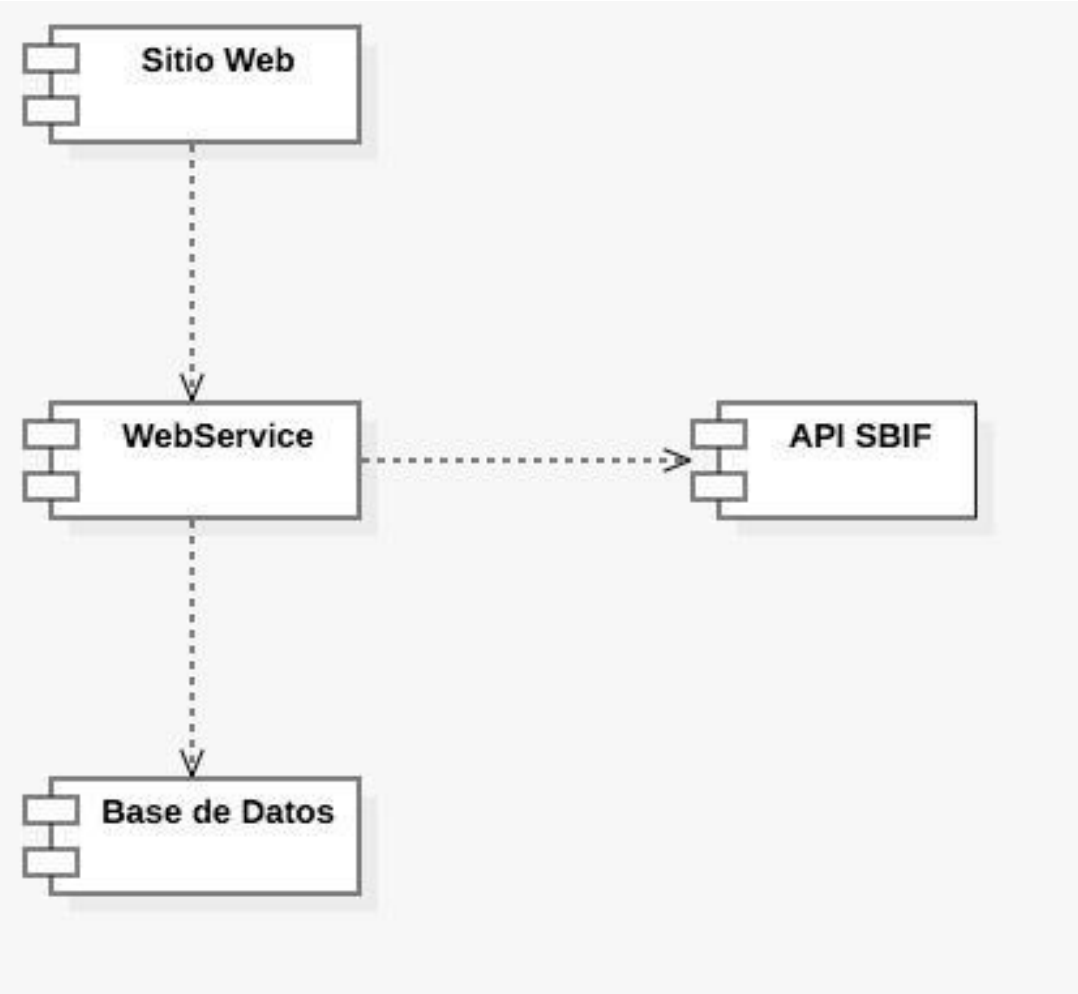


Ilustración 20, Diagrama de Componentes.

8.2.8 Diagrama de Métodos y Clases.

El primer paso va a ser la realización del diagrama de clases, es el diagrama principal para el modelado orientado a objetos. Representa las clases del sistema junto con sus relaciones estructurales agregación y asociación. También se definirán los atributos y operaciones. Con el diagrama de clases entenderemos que funciones pueden ser llevadas a cabo y quien las puede realizar. Como se pueden desarrollar y que tipo de objetos interactúan entre sí.

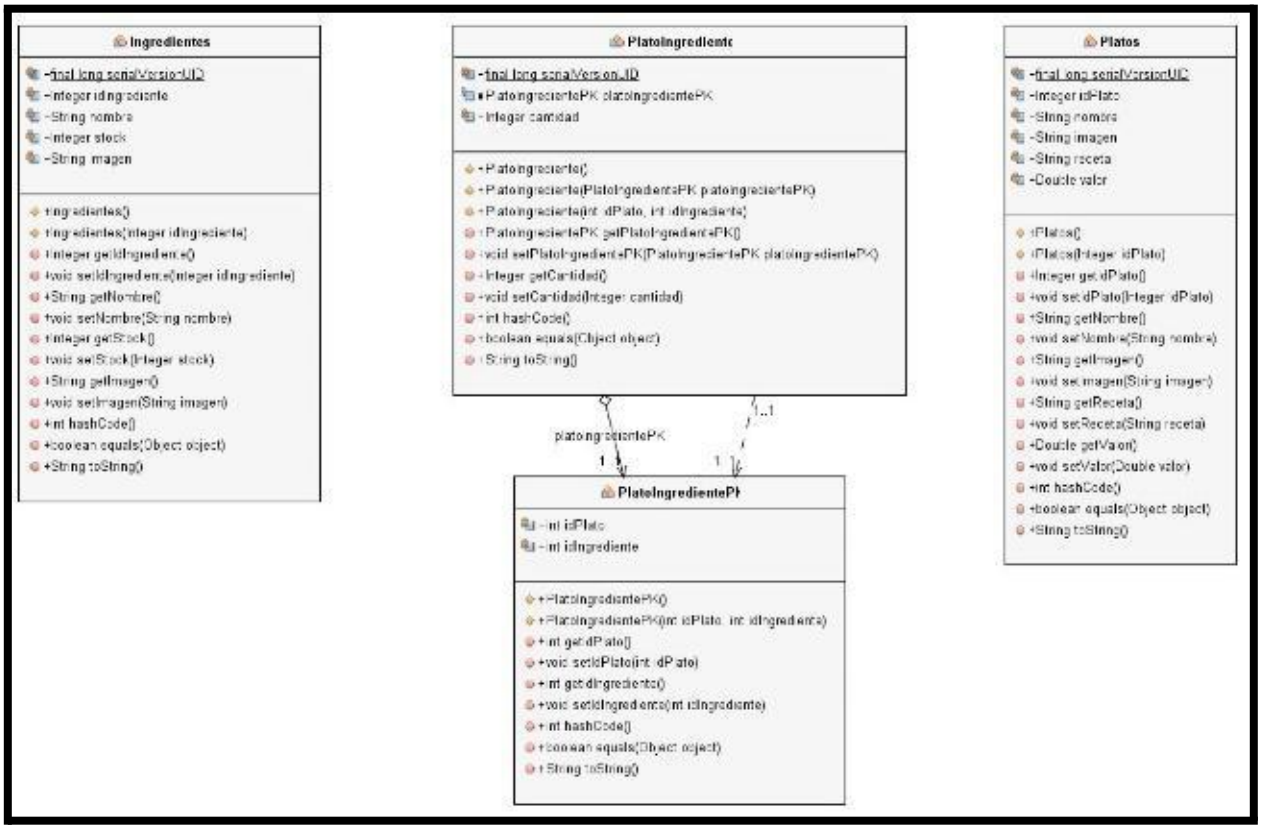


Ilustración 21, Diagrama de Métodos y Clases.

9 Planificación del Proyecto.

9.1 Cronograma de Actividades.

	Nombre	Duración	Inicio	Terminado
1	PROYECTO DE INTEGRACION IV	35 days	11-08-20 8:00	28-09-20 17:00
2	Fase de planificación	2 days	11-08-20 8:00	12-08-20 17:00
3	Calidad, Testing y Tipo de Pruebas	8 days	13-08-20 8:00	24-08-20 17:00
4	Herramientas para Pruebas	10 days	25-08-20 8:00	07-09-20 17:00
5	Diseño de Casos de Prueba	10 days	08-09-20 8:00	21-09-20 17:00
6	Ciclo de prueba	5 days	22-09-20 8:00	28-09-20 17:00

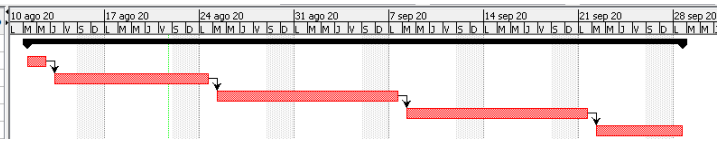


Ilustración 22, Cronograma de Actividades.

9.2 Factibilidad Técnica.

Con este estudio se analizan los recursos técnicos y humanos para el correcto desarrollo del proyecto, como lo es el software, el hardware, el recurso humano y la experiencia o conocimiento para poder lograr este objetivo y establecer su factibilidad o no.

En el desarrollo del portal web de comercio electrónico se debe considerar los siguientes aspectos:

- El uso de aplicaciones web, que implemente la lógica del negocio y responda a las peticiones de los usuarios, mediante el uso de páginas dinámicas.

- Uso de servicios web que representan un modelo de computación distribuida para internet, que manejen la comunicación usuario-aplicación y la interacción de aplicación-aplicación.

- Utilización de la arquitectura de aplicación web, que generalmente incluye un navegador o browser (por ejemplo, Mozilla Firefox o Internet Explorer), la red (Internet), un servidor web y un repositorio de datos.

9.2.1 Sistema operativo.

Este elemento es uno de los más importante ya que logra la comunicación entre el cliente y el producto que necesita, debe ser seguro, cómodo escalable, tener una comunicación rápida con la base de datos, seguridad e integridad de los datos y la información del usuario.

9.2.2 Navegadores Estadísticas 2019.

Se ve un amplio dominio de Chrome, seguido por safari, edge, Firefox y opera, lo que nos indica que debemos tener full compatibilidad con chrome por ser el más utilizados por los usuarios este año.

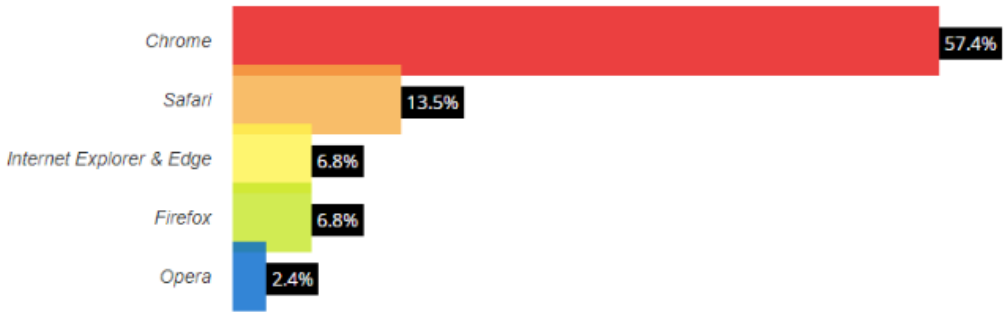


Ilustración 23 Navegadores Estadísticas 2019.

Ejemplo: Alba Mora (2019). Los mejores navegadores web de 2019 [Fotografía]. Recuperado de https://www.pcworld.es/mejores-productos/internet/mejores-navegadores-web-3672988/

9.2.3 Funcionalidades.

Se logra apreciar la compatibilidad con los sistemas operativos de los navegadores.

COMPATIBILIDAD	Google Chrome	Safari	Firefox	Internet Explorer	Microsoft Edge	Opera
Windows	Sí	Sí	Sí	Sí	Sí	Sí
macOS	Sí	Sí	Sí	No	No	Sí
Linux	Sí	No	Sí	No	No	Sí
Android	Sí	No	Sí	No	No	Sí
iOS	Sí	Sí	Sí	No	No	Sí
Windows Phone	No	No	No	Sí	Sí	Sí

Ilustración 24 Funcionalidades.

Ejemplo2: Alba Mora (2019). Los mejores navegadores web de 2019 [Fotografía]. Recuperado de https://www.pcworld.es/mejores-productos/internet/mejores-navegadores-web-3672988/

9.2.4 Compatibilidad HTML5

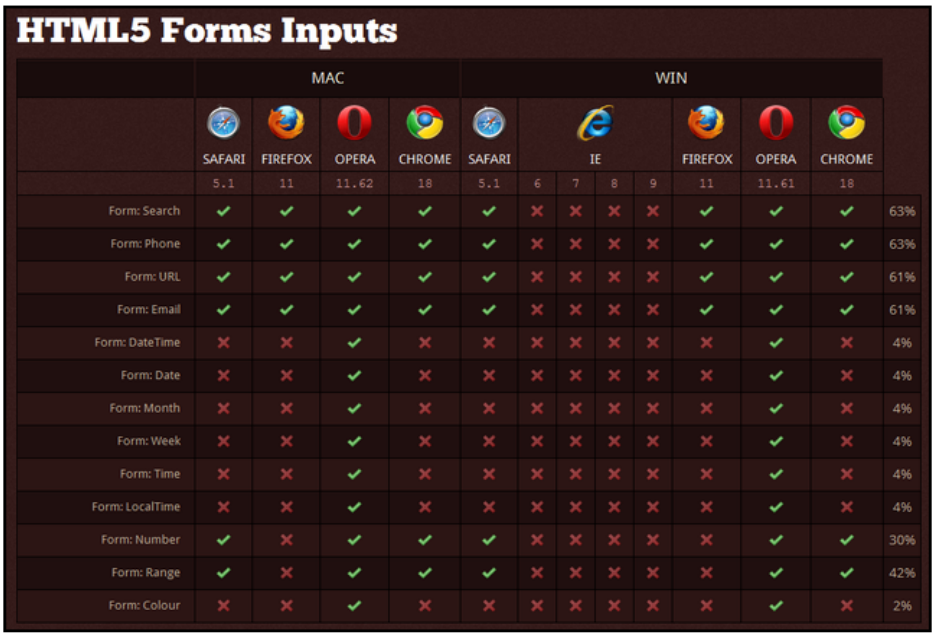


Ilustración 25 Compatibilidad de navegadores con HTML5.

Ejemplo3-4: (2019). Compatibilidad de navegadores con HTML5 ***[Fotografía]. Recuperado de http://desarrolloweb.dlsi.ua.es/cursos/2012/nuevos-estandares-desarrollo-sitios-web/compatibilidad-navegadores-html5***

9.2.5 Sistema Operativo Cliente.

Como este ítem no podemos controlar, debemos tener la máxima de compatibilidad con los navegadores que utilizara el cliente. Ya sea desde escritorio como dispositivos móviles, creando un diseño responsive design.

9.2.6 Sistema Operativo Servidor.

Hay ciertas características que el servidor debe incluir, estabilidad, facilidad en el uso, multiusuario, compatibilidad con el motor de base de datos y seguridad.

- Distribuciones de Linux Server (Centos, Fedora, Ubuntu)

9.2.7 Lenguaje de Desarrollo

El lenguaje de desarrollo del sistema de ventas debe cumplir una serie de características.

- Soporte a gran cantidad de base de datos.
- Facilidad de desarrollo de sistemas.
- Open Source.
- Fácil de administrar.
- Estable y enfocado en ambiente web.

Se presentan:

- Java.
- HTML5 BootStrap.

9.2.8 Gestor de Base de Datos.

El gestor de base de datos será el encargado de administrar toda nuestra información del sitio, donde influirá la velocidad del procesamiento, respaldos y seguridad integra de los datos.

- Estable.
- Segura.
- Escalable.
- Soporte grandes cantidades de información.
- Conexión a diferentes lenguajes de programación PDO de PHP.

La opción :

MySQL.

9.2.9 Características en Hardware.

Las características de los equipos para desarrollar este sitio, sumado a las características del servidor donde se mantendrá la base de datos y los procesamientos.

Equipo	Elemento	Capacidad
Computador 1	Memoria Disco Duro Procesador	Min 4, recomendada 8gb RAM 500 Gb Mínimo recomendó 1tb Intel I3/ AMD A6 Superior
Computador 2	Memoria Disco Duro Procesador	Min 4, recomendada 8gb RAM 500 Gb Mínimo recomendó 1tb Intel I3/ AMD A6 Superior
Computador 3	Memoria Disco Duro Procesador	Min 4, recomendada 8gb RAM 500 Gb Mínimo recomendó 1tb Intel I3/ AMD A6 Superior
Computador 4	Memoria	Min 4, recomendada 8gb RAM

	Disco Duro Procesador	500 Gb Mínimo recomendó 1tb Intel I3/ AMD A6 Superior
Computador 5	Memoria Disco Duro Procesador	Min 4, recomendada 8gb RAM 500 Gb Mínimo recomendó 1tb Intel I3/ AMD A6 Superior
Servidor 1	Memoria Disco Duro Procesador Fuente de poder Sistema Operativo	8 gb recomendada 16gb 1tb recomendado raid 0,1 Xeon 2.4 ghz (x2) 1 mínima, 2 recomendada Windows 2012 server Distribución Linux

9.2.10 Experiencia y conocimiento del Equipo.

El recurso humano con que contamos para este proyecto consta de cinco compañeros más un profesor guía.

Recurso Humano	Cinco integrantes
Experiencia	Administración de proyectos informáticos Desarrollo de sistemas Programación web Trabajo en Equipo
Conocimiento	Lenguajes de programación en ambiente web Manejo de Base de Datos

9.2.11 Alojamiento web.

El Alojamiento web, debe ser con un gran soporte técnico, compatible con los softwares que instalaremos, fácil de administrar, un respaldo automático, habilitación de cuentas de correos administrativas.

Veremos algunas características que necesitamos.

- Espacio en disco duro mínimo 1gb.
- Transferencia mensual ilimitada.
- 1 base de Datos mínimo.
- 1 cuenta FTP mínimo.
- 5 cuentas de correo empresarial mínimo.
- Certificado SSL.
- Uptime mínimo 95%.
- Backup automático.
- Cpanel.
- Linux/Windows.
- Antivirus.

9.3 Factibilidad Económica.

Los costos corresponden a lo que se debe pagar durante el primer año; esto puede variar en los siguientes años o también dependiendo del plazo por el cual se contraten los servicios distintos servicios asociados:

Descripción	Costos (Primer año)
Hosting*	$199.900 + IVA
Dominio Nic.cl	$8.361 + IVA
Certificado SSL (EV) para E-commerce**	$101.390

*Contrato con empresa local Hosting.cl con servidor dedicado para la aplicación.

**Compra en Godaddy de certificado por 1 año barra de dirección verde.

Los beneficios que se pueden obtener al implementar una aplicación pueden ser los que se detallan en el siguiente cuadro. En cuanto a los costos estos son valores aproximados ya que estos son variables en el tiempo.

Descripción	Costos (Primer año)
Aumento en Ventas	30%
Mejora en tiempos de atención	25%
Retención de clientes	70%
Mejora la oferta con respecto a la competencia	20%

Luego de revisar los cuadros anteriores podemos calcular el beneficio neto que podemos obtener a partir de todos los costos asociados:

Beneficios – Costos = Beneficios Netos

Si bien, actualmente no tenemos los números asociados a los beneficios que se producen a partir de la implementación de un nuevo canal de ventas como lo es la aplicación, ya podemos detectar varios elementos que impactan directamente los costos en el primer año de funcionamiento.

En el primer año, el propósito es lograr la fidelización del cliente, esto lo podemos lograr de dos formas: con mejores tiempos de atención y mejoras de la oferta con respecto a la competencia con lo cual garantizamos una retención de la clientela como también captar nuevos posibles clientes a través de este nuevo canal de ventas, esperando lograr un aumento en las ventas.

9.4 Pruebas del Software.

Las pruebas del software consisten en verificar el comportamiento de un programa dinámicamente a través de un grupo finito de casos de prueba, debidamente seleccionados en relación del comportamiento esperado.

La apreciación de las pruebas del software ha evolucionado hacia una forma más constructiva. Ya no se asume que realizar pruebas es una tarea que empieza solamente cuando la fase de programación se ha completado, y que tiene el único propósito de detectar errores. Las pruebas de software se ven ahora como una actividad que debería estar presente durante todo el proceso de desarrollo del software y mantenimiento, así también es una parte importante de la construcción del producto.

En la actualidad se considera que la prevención es la actitud adecuada en lo que respecta a la calidad, es mejor evitar problemas que solucionarlos. Realizar pruebas debe verse como un medio para verificar, no sólo si la prevención ha sido efectiva, si no para identificar fallos en aquellos casos en los que, por alguna razón, no lo ha sido. Incluso después de una campaña de pruebas exhaustiva, el software aún podría contener errores.

9.4.1 Herramienta De Testing.

Las pruebas automatizadas fueron ejecutadas haciendo uso de las herramientas seleccionadas: JUnit, una librería desarrollada para poder probar el funcionamiento de las clases y métodos que componen nuestra aplicación, y asegurarnos de que se comportan como deben ante distintas situaciones de entrada y Postman para el testing de API REST del aplicativo. Para realizar pruebas de rendimiento de aplicaciones se usará la herramienta JMeter. A continuación, y en primer lugar se presentan los resultados obtenidos. En las

siguientes figuras se muestran las pruebas que fueron correctamente ejecutadas, junto a los resultados obtenidos.

9.4.2 Pruebas Unitarias

Las pruebas de unidad verifican el funcionamiento aislado de las partes del software que se pueden probar independientemente. Dependiendo del contexto, estos podrían ser subprogramas individuales o un componente más grande formado por unidades muy relacionadas. Normalmente, las pruebas de unidad se realizan con acceso al código fuente y con el soporte de herramientas de depuración, pudiendo implicar a los programadores que escribieron el código.

La prueba de unidad enfoca los esfuerzos de verificación en la unidad más pequeña del diseño de software: el componente o módulo de software. Este tipo de pruebas puede realizarse en paralelo para múltiples componentes y por lo general se consideran como adjuntas al paso de codificación.

Realizado por	Hernan Gonzalez Diaz		
Objetivo de la prueba 1	Validar campos vacíos correctos (Ingrediente) Validar si todos los datos de los campos de un nuevo ingrediente vienen con información.		
Pre- requisitos	1. Base de datos disponible para hacer transacciones.		
DETALLE CASO DE PRUEBA			
Paso#	**Acción Programador**	**Respuesta esperada del sistema**	**Resultado**
1.1	Id=”1”, nombre=”prueba”, stock=”20”, imagen=”imagen”	Mensaje que indica que en la respuesta retorna el objeto de conexión de base de datos.	OK
OBSERVACIONES ADICIONALES			
● La respuesta de la prueba unitaria tiene tiempos de respuesta adecuados, a continuacion evidencia de la consola de NetBeans			
Tiempo utilizado: 0,014 sec			
Aprobado	Hernan Gonzalez Diaz		

Ilustración 26, Pruebas Unitarias.

Realizado por	Hernan Gonzalez Diaz

Objetivo de la prueba 2	Validar campos vacíos incorrectos – campo vacío (Ingrediente). Validar que al ingresar un parámetro vacío el método devuelva false.		
Pre- requisitos	2. Base de datos disponible para hacer transacciones.		
DETALLE CASO DE PRUEBA			
Paso#	**Acción Programador**	**Respuesta esperada del sistema**	**Resultado**
1.1	Id=”1”, nombre=”prueba”, stock=””, imagen=”imagen”	Mensaje que indica que en la respuesta retorna el objeto de conexión de base de datos.	OK
OBSERVACIONES ADICIONALES			
● La respuesta de la prueba unitaria tiene tiempos de respuesta adecuados, a continuacion evidencia de la consola de NetBeans.			
Tiempo utilizado: 0,014 sec			
Aprobado	Hernan Gonzalez Diaz		

Ilustración 27, Pruebas Unitarias.

Realizado por	Hernan Gonzalez Diaz
Objetivo de la prueba 3	Validar campos vacíos incorrectos – campo nulo (Ingrediente)

<table>
<tr><td colspan="2"></td><td colspan="2">Validar que al ingresar un parámetro nulo el método devuelva false.</td></tr>
<tr><td colspan="2">Pre- requisitos</td><td colspan="2">3. Base de datos disponible para hacer transacciones</td></tr>
<tr><td colspan="4">DETALLE CASO DE PRUEBA</td></tr>
<tr><td>Paso#</td><td>Acción Programador</td><td>Respuesta esperada del sistema</td><td>Resultados</td></tr>
<tr><td>1.1</td><td>Id=”1”, nombre=”prueba”, stock=”20”, imagen=null</td><td>Mensaje que indica que en la respuesta retorna el objeto de conexión de base de datos.</td><td>OK</td></tr>
<tr><td colspan="4">OBSERVACIONES ADICIONALES</td></tr>
<tr><td colspan="4">● La respuesta de la prueba unitaria tiene tiempos de respuesta adecuados, a continuacion evidencia de la consola de NetBeans</td></tr>
<tr><td colspan="2">Tiempo utilizado: 0,014 sec</td><td colspan="2"> 
</td></tr>
<tr><td>Aprobado</td><td colspan="3">Hernan Gonzalez Diaz</td></tr>
</table>

Ilustración 28, Pruebas Unitarias.

Realizado por	Hernan Gonzalez Diaz
Objetivo de la prueba 4	Validar tipos numéricos previo a conversión (Correcto).
	Validar que, al ingresar un parámetro como texto, este contenga un numero válido previo a conversión.
Pre- requisitos	4. Base de datos disponible para hacer transacciones

DETALLE CASO DE PRUEBA			
Paso#	**Acción Programador**	**Respuesta esperada del sistema**	**Resultado**
1.1	num=”15”	Mensaje que indica que en la respuesta retorna el objeto de conexión de base de datos.	OK
OBSERVACIONES ADICIONALES			
● La respuesta de la prueba unitaria tiene tiempos de respuesta adecuados, a continuacion evidencia de la consola de NetBeans			
Tiempo utilizado: 0,014 sec			
Aprobado	Hernan Gonzalez Diaz		

Ilustración 29, Pruebas Unitarias.

Realizado por	Hernan Gonzalez Diaz
Objetivo de la prueba 5	Validar tipos numéricos previo a conversión (Incorrecto). Validar que, al ingresar un parámetro como texto, y este no contenga un número válido previo a conversión nos devuelva false.
Pre- requisitos	5. Base de datos disponible para hacer transacciones.
DETALLE CASO DE PRUEBA	

Paso#	Acción Programador	Respuesta esperada del sistema	Resultado
1.1	num=”prueba”	Mensaje que indica que en la respuesta retorna el objeto de conexión de base de datos.	OK
OBSERVACIONES ADICIONALES			
● La respuesta de la prueba unitaria tiene tiempos de respuesta adecuados, a continuacion evidencia de la consola de NetBeans.			
Tiempo utilizado: 0,014 sec			
Aprobado	Hernan Gonzalez Diaz		

Ilustración 30, Pruebas Unitarias.

Realizado por	Hernan Gonzalez Diaz
Objetivo de la prueba 6	Prueba unitaria para validar la conexión hacia la base de datos MySQL.
Pre- requisitos	6. Base de datos disponible para hacer transacciones.

DETALLE CASO DE PRUEBA

Paso#	Acción Programador	Respuesta esperada del sistema	Resultado
1.1	Ejecutar el Test File de la clase TestUnitarioDB	Mensaje que indica que en la respuesta retorna el objeto de conexión de base de datos.	OK

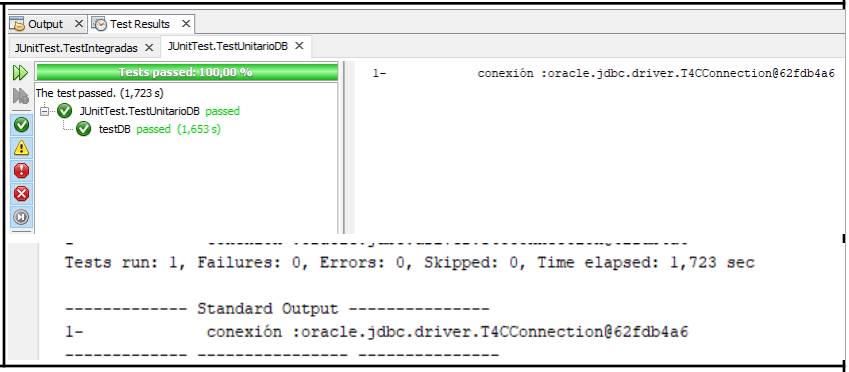
	que ejecuta la prueba unitaria		
OBSERVACIONES ADICIONALES			
La respuesta de la prueba unitaria tiene tiempos de respuesta adecuados, a continuacion evidencia de la consola de NetBeans.			
Tiempo utilizado: 1,723 sec			
Aprobado	Hernan Gonzalez Diaz		

Ilustración 31, Pruebas Unitarias.

Realizado por	Hernan Gonzalez Diaz
Objetivo de la prueba 7	Validar la integración del test2Total() que retorna el total del costo del plato.
Pre- requisitos	1. Base de datos disponible para hacer transacciones. 2. Que existan registros de los platos en la base de datos. 3. Servicio restFull disponible en el servidor web. 4. Ingresar el parámetro monto e Id de plato.

	5. Método dao ejecuta la transacción a la base de datos para obtener los registros.

DETALLE CASO DE PRUEBA			
Paso#	**Acción Programador**	**Respuesta esperada del sistema**	**Resultado**
1.2	Ejecutar el Test File de la clase TestUnitarioTotalCuenta que ejecuta la prueba unitaria.	El mensaje del segundo método test2Total(), primero ejecuta validación si el objeto de conexión es distinto de null, se pasa por parámetro el id del plato, en la respuesta se obtiene el costo del plato.	OK

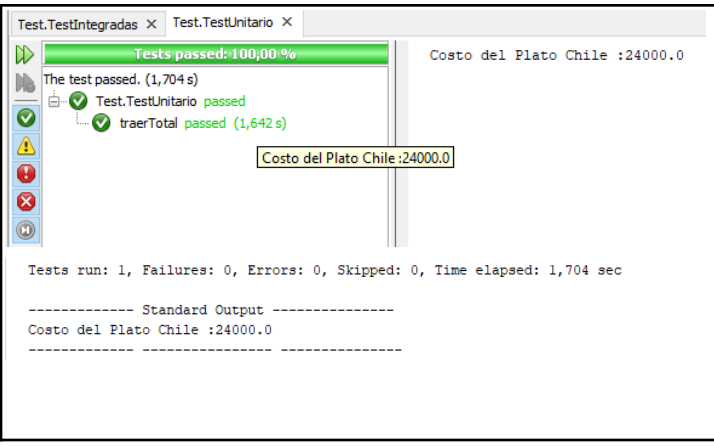
OBSERVACIONES ADICIONALES	
La respuesta de la prueba unitaria tiene tiempos de respuesta adecuados, a continuacion evidencia de la consola de NetBeans.	
Tiempo utilizado: 1,704 sec	
Aprobado	Hernan Gonzalez Diaz

Ilustración 32, Pruebas Unitarias.

Realizado por	Hernan Gonzalez Diaz
Objetivo de la prueba 8	Validar la integración del test3Conversion() que retorna el total del costo del plato

Pre- requisitos	1. Base de datos disponible para hacer transacciones. 2. Que existan registros de los platos en la base de datos. 3. Servicio restFull disponible en el servidor web. 4. Ingresar el parámetro monto e Id de plato. 5. Método dao ejecuta la transacción a la base de datos para obtener los registros.

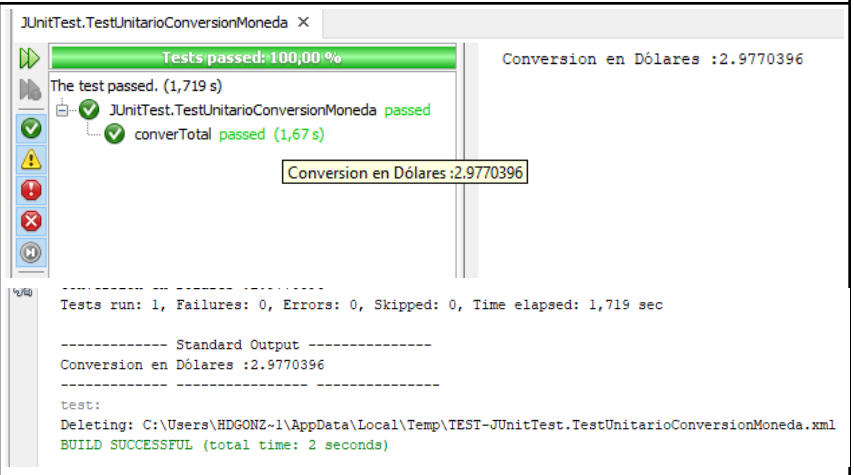
DETALLE CASO DE PRUEBA

Paso#	Acción Programador	Respuesta esperada del sistema	Resultado
1.3	Ejecutar el Test File de la clase TestUnitarioConversionMoneda que ejecuta la prueba de unitaria.	El mensaje del tercer método test3Conversion(), solo continua la ejecución si el costo total del plato es mayor a cero, si es así, en la respuesta entrega la moneda del cliente y la conversión del costo de plato en la moneda del cliente.	OK

OBSERVACIONES ADICIONALES

La respuesta de la prueba unitaria tiene tiempos de respuesta adecuados, a continuacion evidencia de la consola de NetBeans.

Tiempo utilizado:
1,719 sec



Aprobado	Hernan Gonzalez Diaz

Ilustración 33, Pruebas Unitarias.

9.4.3 Pruebas de Integración

Las pruebas de integración son una técnica sistemática para construir la arquitectura del software mientras se llevan a cabo pruebas para descubrir errores asociados con la interfaz. El objetivo es tomar los componentes probados de manera individual y construir una estructura de programa que se haya dictado por diseño.

Una prueba de integración es el proceso de verificar la interacción entre componentes de software. Estrategias clásicas de integración, como arriba-abajo o abajo-arriba, se usan tradicionalmente, con un software estructurado jerárquicamente.

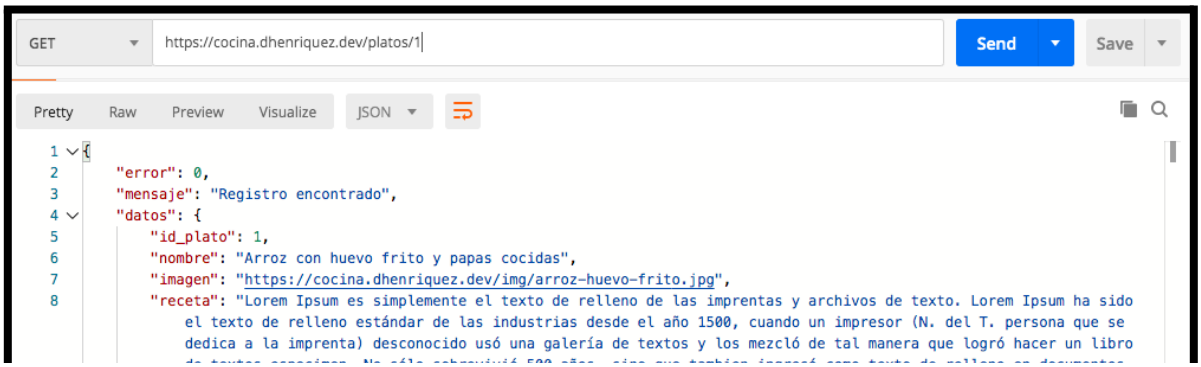
Las estrategias modernas de integración están dirigidas por la arquitectura, lo que supone integrar los componentes de software o subsistemas basándose en caminos de funcionalidad identificada. Las pruebas de integración son una actividad continua, que sucede en cada fase en que los ingenieros de software tienen que hacer abstracciones de las perspectivas del nivel que se están integrando. Con la excepción de software sencillo y pequeño, las estrategias de pruebas de integración sistemáticas e incrementales es preferiblemente probar todos los componentes juntos al final, lo que se conoce como, pruebas en “Big-Bang”.

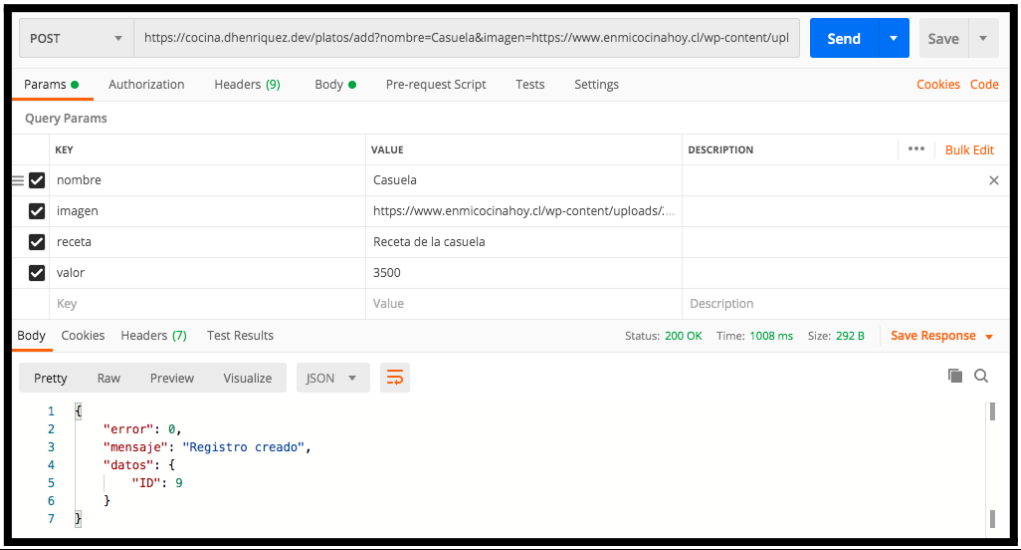
Las pruebas de integración que se presentan a continuación fueron realizadas al Webservice.

EndPoint	Descripción	Entrada	Salida	Estado
https://cocina.dhenriquez.dev/platos/{ID}	Se pasa como parámetro string como ID	hola	Error No se encontró registro (img 1)	Correcto

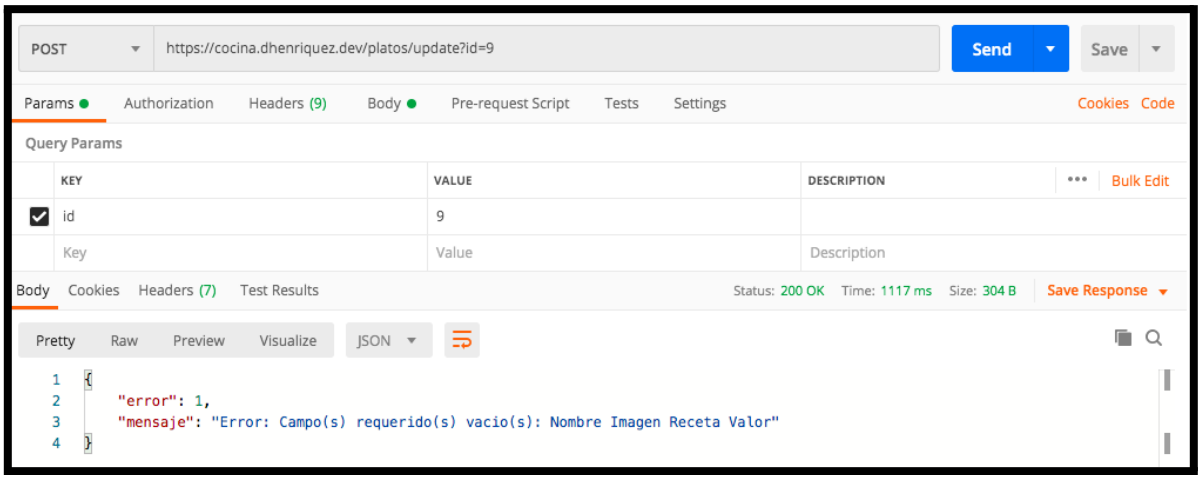
EndPoint	Descripción	Entrada	Salida	Estado
https://cocina.dhenriquez.dev/platos/add	Se envía sin parámetros		Error campos requeridos (img 3)	Correcto

EndPoint	Descripción	Entrada	Salida	Estado
https://cocina.dhenriquez.dev/platos/{ID}	Se pasa como parámetro id 1	1	Datos de plato 1 (img 2)	Correcto

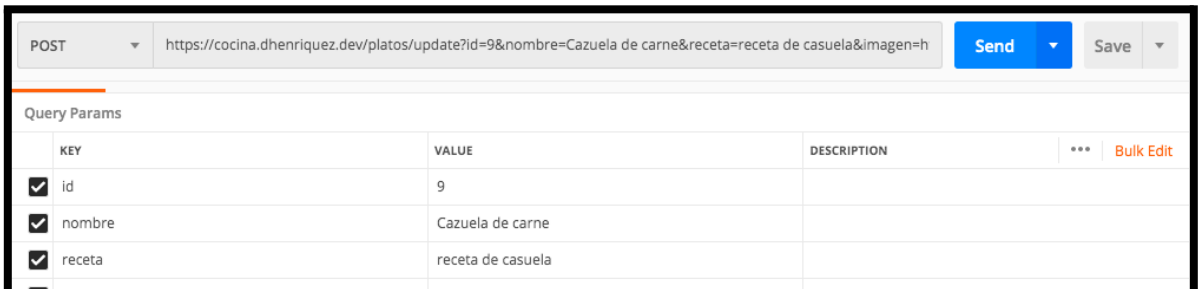




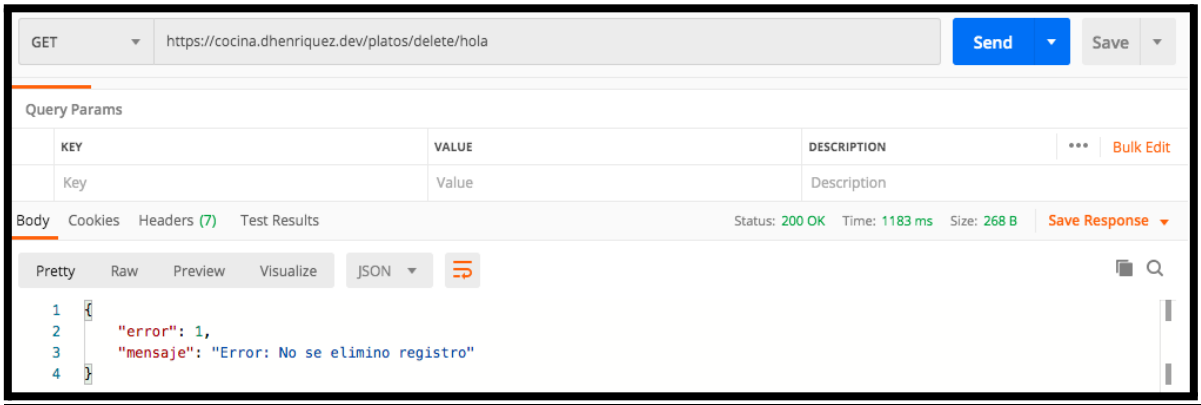
EndPoint	Descripción	Entrada	Salida	Estado
https://cocina.dhenriquez.dev/platos/update	Se envía solo el ID como parámetro	9	Error campos requeridos (img 5)	Correcto



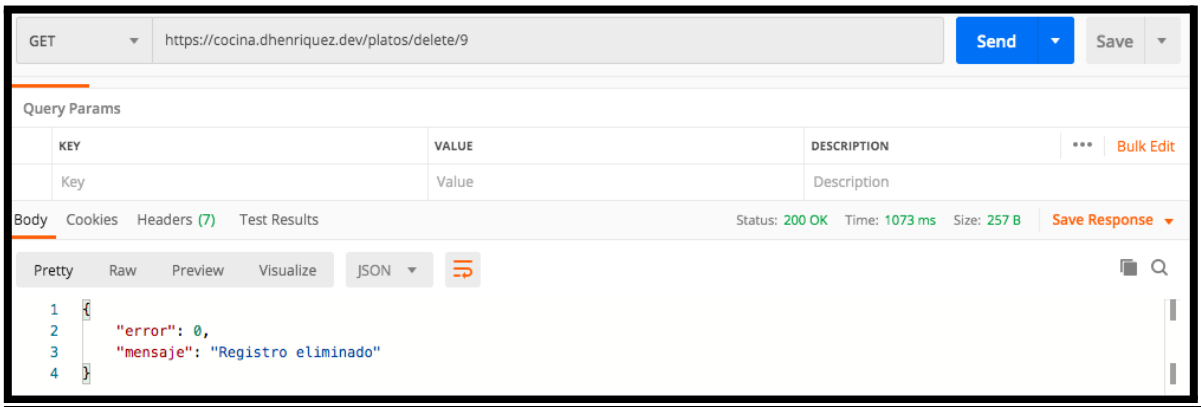
EndPoint	Descripción	Entrada	Salida	Estado
https://cocina.dhenriquez.dev/platos/update	Se envían todos los parámetros requeridos	id, nombre, imagen url, receta, valor	Registro actualizado (img-6)	Correcto



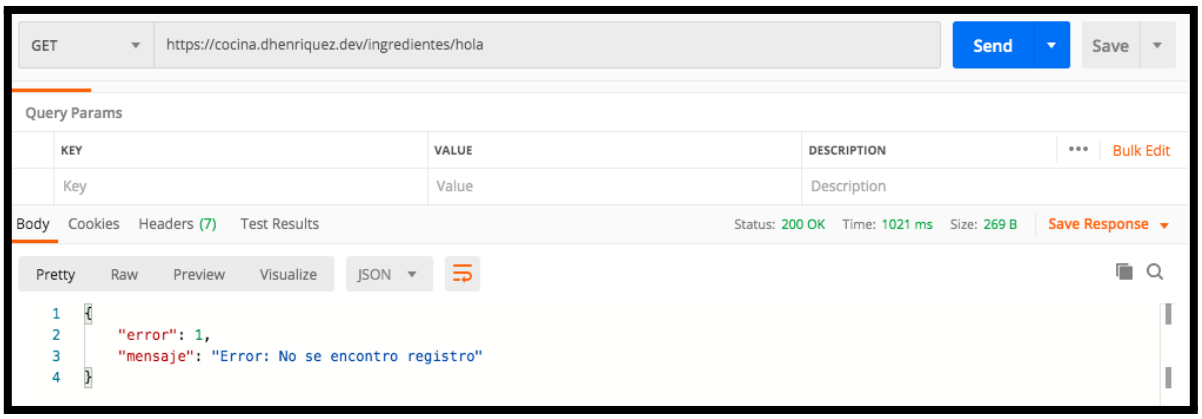
EndPoint	Descripción	Entrada	Salida	Estado
https://cocina.dhenriquez.dev/platos/delete/{ID}	Se pasa como parámetro string como ID	hola mundo	Error no se elimino registro (img-7)	Correcto



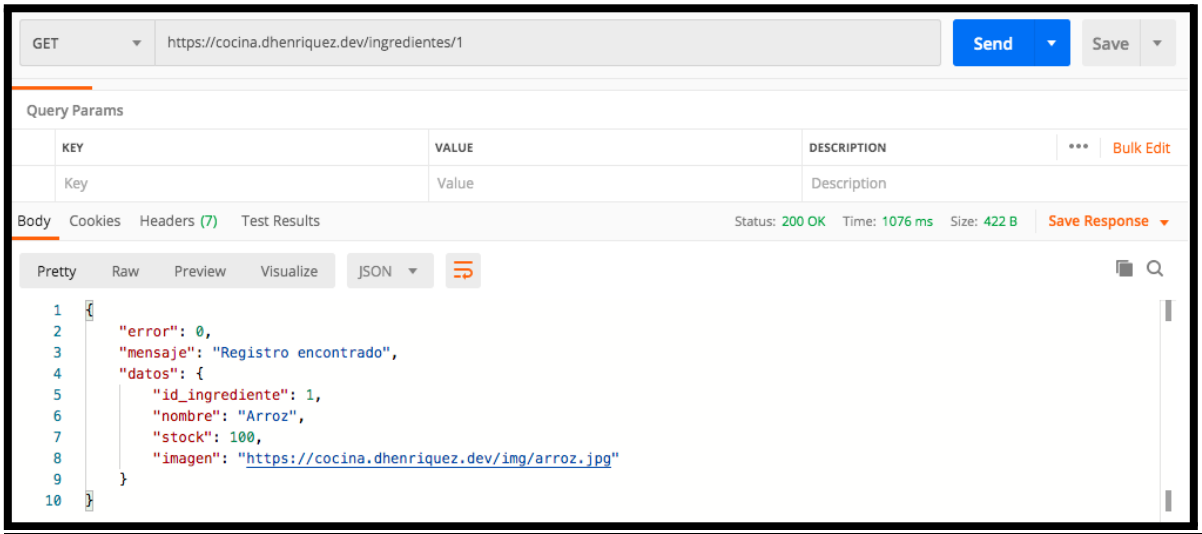
EndPoint	Descripción	Entrada	Salida	Estado
https://cocina.dhenriquez.dev/platos/delete/{ID}	Se pasa como parámetro id 9	9	Registro eliminado (img-8)	Correcto



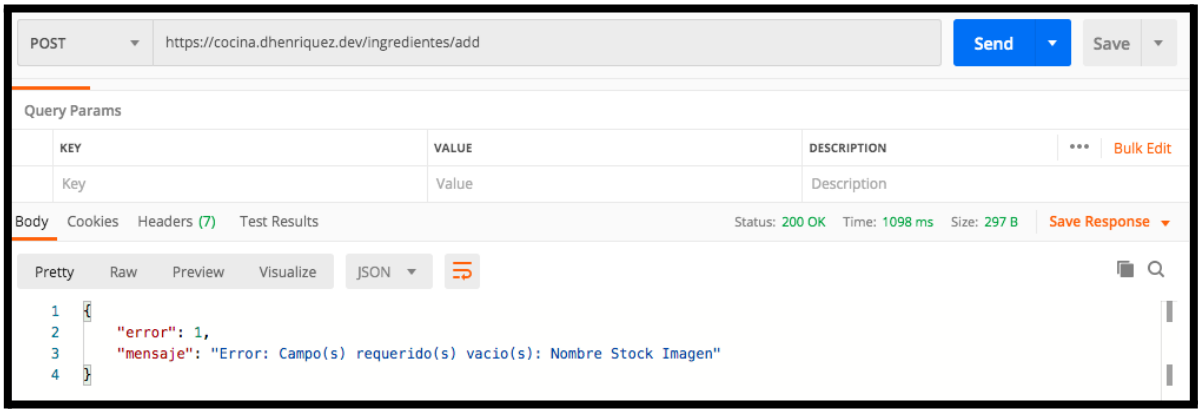
EndPoint	Descripción	Entrada	Salida	Estado
https://cocina.dhenriquez.dev/ingredientes/{ID}	Se pasa como parámetro string como ID	hola	Error No se encontró registro (img 9)	Correcto



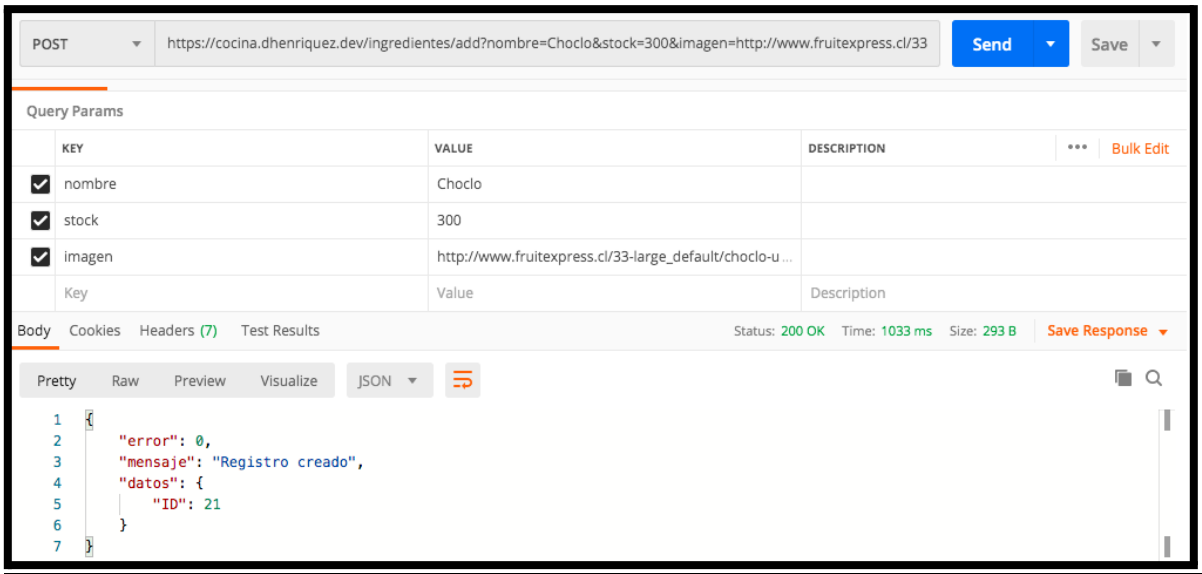
EndPoint	Descripción	Entrada	Salida	Estado
https://cocina.dhenriquez.dev/ingredientes/{ID}	Se pasa como parámetro id 1	1	Datos de ingrediente 1 (img-10)	Correcto



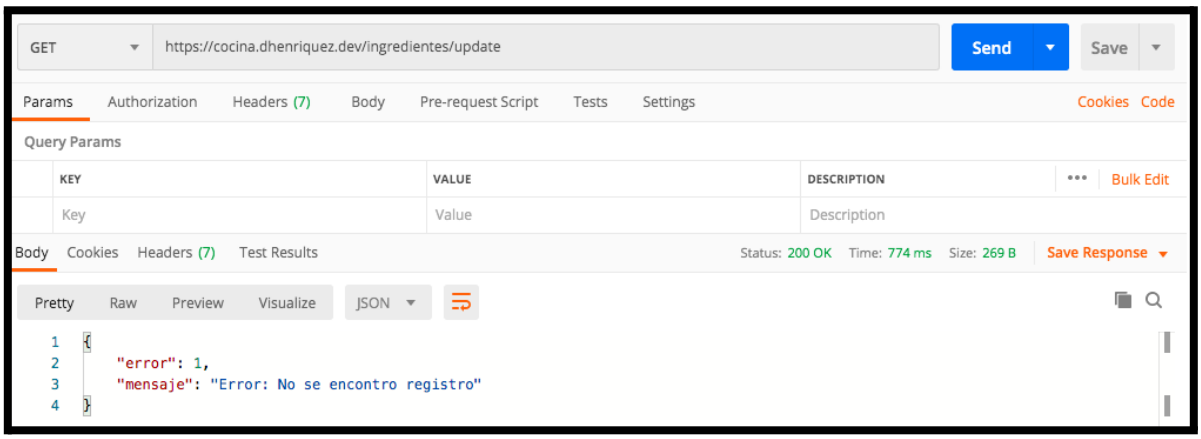
EndPoint	Descripción	Entrada	Salida	Esta
https://cocina.dhenriquez.dev/ingredientes/add	Se envía sin parámetros		Error campos requeridos (img 11)	Corre



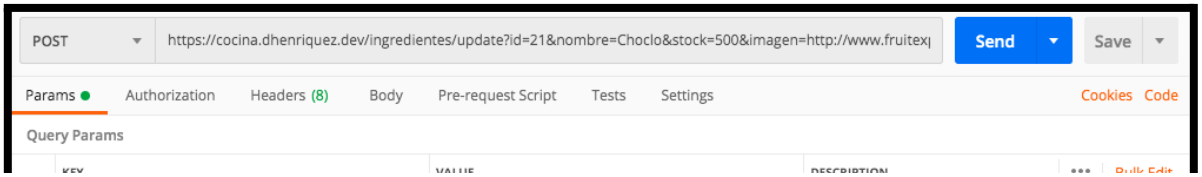
EndPoint	Descripción	Entrada	Salida	Estado
https://cocina.dhenriquez.dev/ingredientes/add	Se envía parámetro nombre, stock, imagen	nombre, stock imagen	Registro creado 21 (img-12)	Correcto



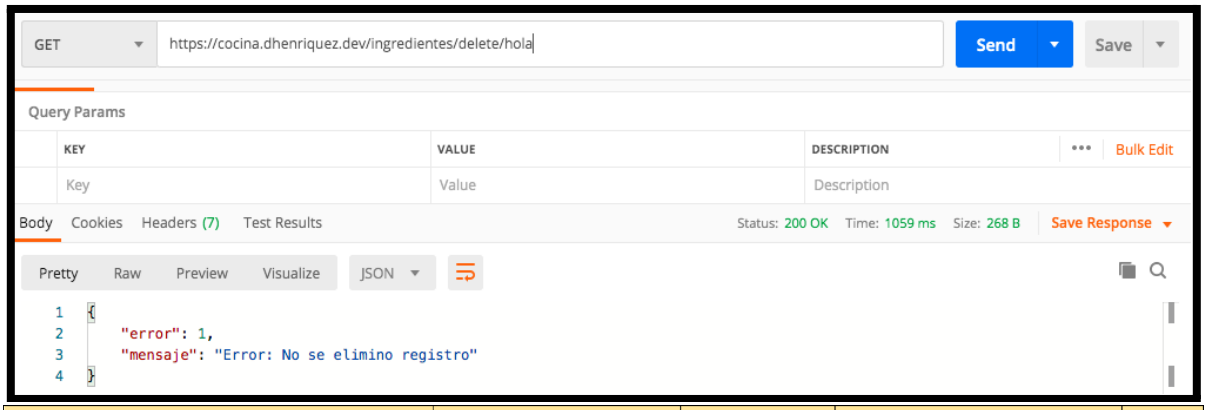
EndPoint	Descripción	Entrada	Salida	Esta
https://cocina.dhenriquez.dev/ingredientes/update	Se envía sin parámetros		Error campos requeridos (img 13)	Corre



EndPoint	Descripción	Entrada	Salida	Estado
https://cocina.dhenriquez.dev/ingredientes/update	Se envían parámetros id nombre, stock imagen	id, nombre, stock, imagen	Registro actualizado (img-14)	Correcto



EndPoint	Descripción	Entrada	Salida	Estado
https://cocina.dhenriquez.dev/ingredientes/delete/{ID}	Se pasa como parámetro string como ID	hola	Error no se elimino registro (img-15	Correcto



EndPoint	Descripción	Entrada	Salida	Estado
https://cocina.dhenriquez.dev/ingredientes/delete/{ID}	Se pasa como parámetro id 21	21	Registro eliminado (img-16)	Correcto

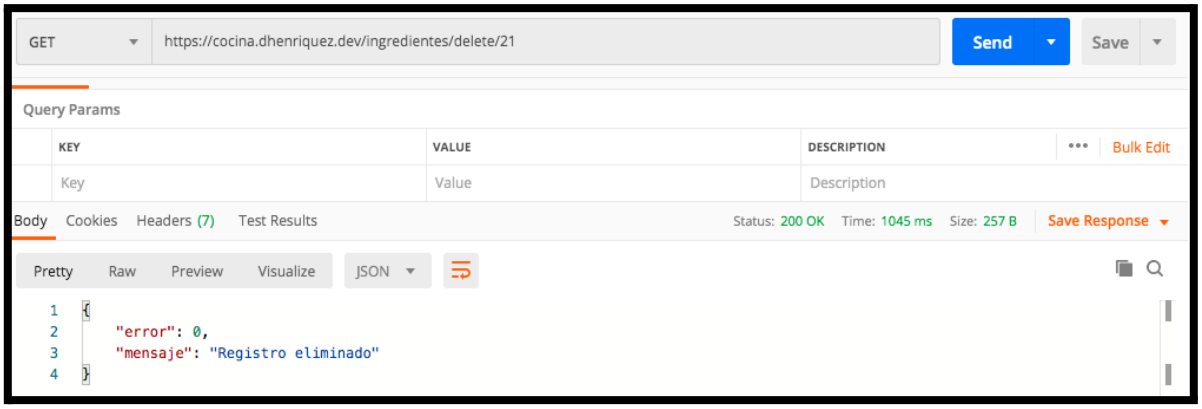


Ilustración 34, Pruebas de Integración.

Realizado por	Hernan Gonzalez Diaz
Objetivo de la prueba	Validar la integración de los tres métodostest1DB(), test2Total(), test3Conversion().
Pre- requisitos	1. Base de datos disponible para hacer transacciones. 2. Que existan registros de los platos en la base de datos. 3. Servicio restFull disponible en el servidor web. 4. Ingresar el parámetro monto e Id de plato. 5. Método dao ejecuta la transacción a la base de datos para obtener los registros.

DETALLE CASO DE PRUEBA			
Paso#	**Acción Programador**	**Respuesta esperada del sistema**	**Resultado**

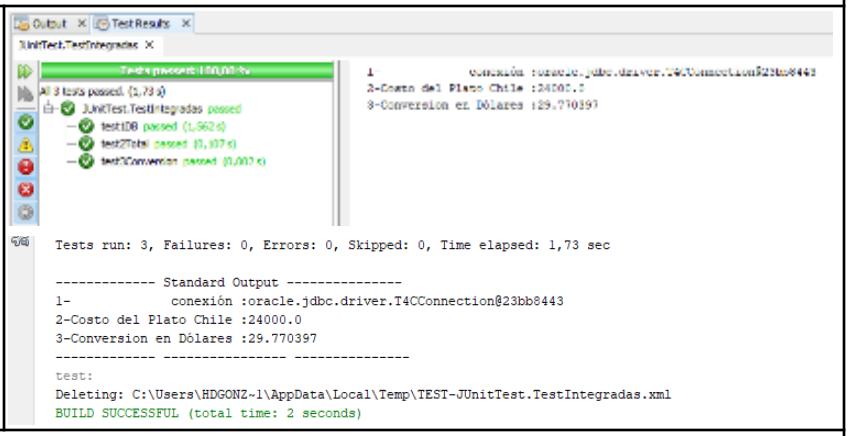
1.1	Ejecutar la prueba Test File de Junit Test de la clase **TestIntegradas** que ejecuta la prueba de integración secuencial llamando de los métodos: test1DB() test2Total() test3Conversion()	1- El mensaje del primer método test1DB() retorna el objeto de conexión de a la base datos, el objeto conexión es instanciado a la propiedad conInt que se mantiene activo durante la ejecución del Test File.	OK
1.2	Continua la ejecución del segundo test de la secuencia test2Total()	2- El mensaje del segundo método test2Total(), primero ejecuta validación si la el objeto de conexión es distinto de null, se pasa por parámetro el id del plato, en la respuesta se obtiene el costo del plato.	OK
1.3	Continua la ejecución del segundo test de la secuencia test3Conversion()	3- El mensaje del tercer método test3Conversion(), solo continua la ejecución si el costo total del plato es mayor a cero, si es así, en la respuesta entrega la moneda del cliente y la conversión del costo de plato en la moneda del cliente.	OK
OBSERVACIONES ADICIONALES			
La respuesta de la prueba integrada de los Test run, tiene tiempos de respuesta adecuados, a continuacion evidencia de la consola de NetBeans			
Tiempo utilizado: 1,73 sec			
Aprobado	Hernan Gonzalez Diaz		

Ilustración 35, Pruebas de Integración.

9.4.4 Pruebas de sistema.

Las pruebas del sistema se ocupan del comportamiento de un sistema completo. La mayoría de los fallos funcionales deberían haber sido identificados antes, durante las fases de pruebas de unidad y pruebas de integración. Las pruebas de sistema se consideran normalmente como las apropiadas para comparar el sistema con los requisitos no funcionales del sistema, como seguridad, rendimiento, exactitud, velocidad y confiabilidad. Las interconexiones externas con otras aplicaciones, utilidades, dispositivos de hardware o con el sistema operativo, también se evalúan en este nivel.

I. Módulo de “Ingredientes”: Este módulo tiene como flujo de prueba integral las siguientes funcionalidades:

- Agregar Ingrediente.
- Modificar Ingrediente.
- Eliminar Ingrediente.

En base a los puntos anteriormente se realizará una prueba integral paso a paso con el flujo correspondiente.

Agregar Ingrediente: Se procede agregar un ingrediente nuevo “Pimiento Rojo”

```
public String agregar(String nombre, int stock, String imagen) {
    try {
        Ingredientes i = new Ingredientes();
        i.setIdIngrediente(em.createQuery("select max(i.idIngrediente) + 1 from Ingredientes i", Integer.class).getSingleResult());
        i.setNombre(nombre);
        i.setStock(stock);
        i.setImagen(imagen);
        em.persist(i);
        return "Ingrediente agregado!.";
    } catch (Exception e) {
        return "Error: " + e;
    }
}
```

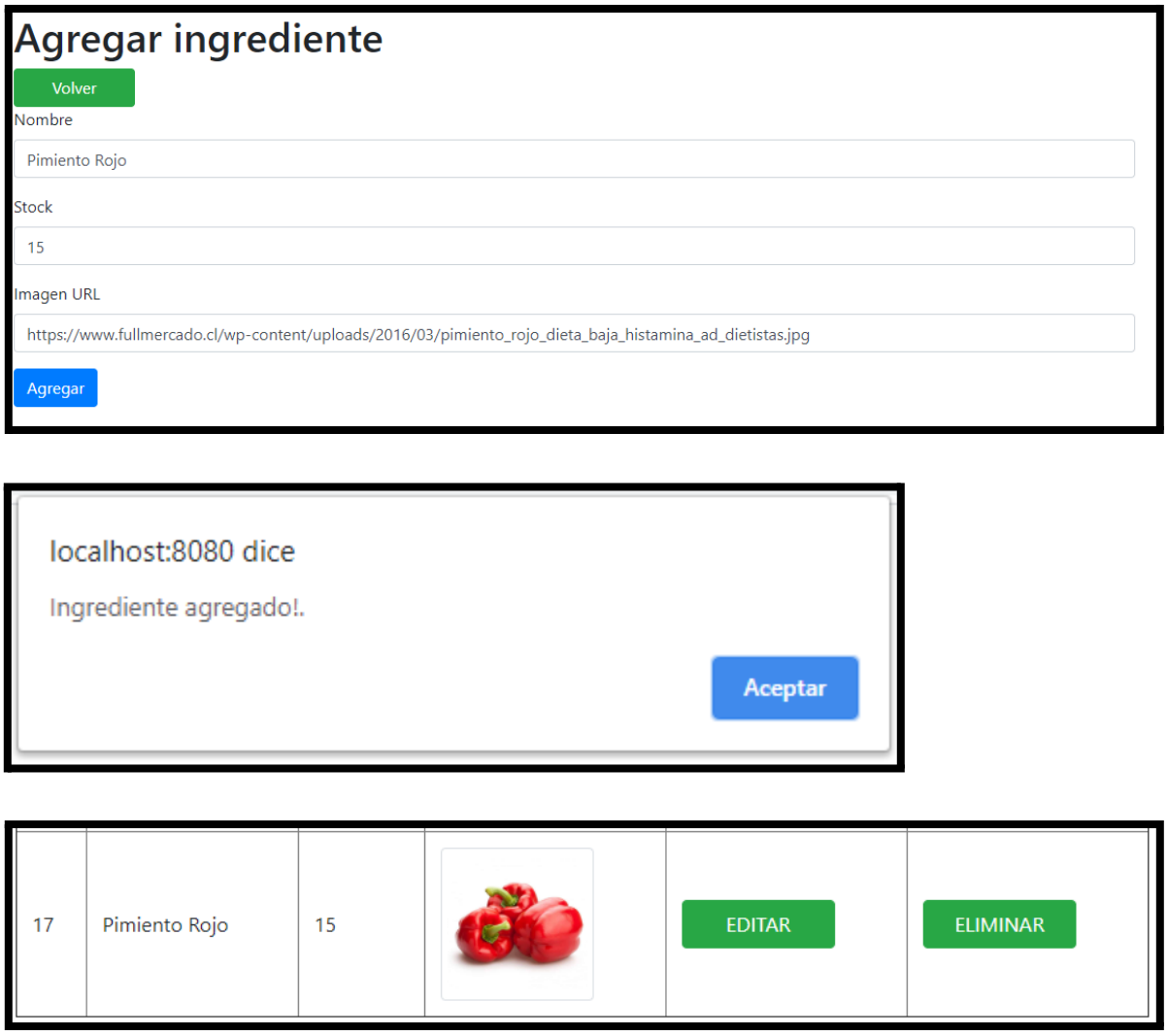


Ilustración 36, Pruebas de sistema.

Modificar Ingrediente: Se procede a modificar el stock de este ingrediente que inicialmente está en 15, ahora lo aumentaremos a 20.

```
public String editar(int id, String nombre, int stock, String imagen) {
    try {
        Ingredientes i = em.find(Ingredientes.class, new Integer(id));
        i.setNombre(nombre);
        i.setStock(stock);
        i.setImagen(imagen);
        em.merge(i);
        return "Ingrediente actualizado!.";
    } catch (Exception e) {
        return "Error: " + e;
    }
}
```

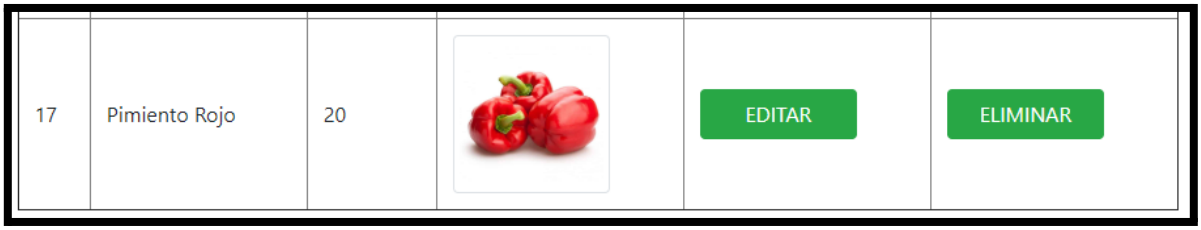


Ilustración 37, Pruebas de sistema.

Eliminar Ingrediente: Se procede a eliminar el ingrediente recién añadido “Pimiento Rojo”.

```
public String eliminar(int id) {
    try {
        Ingredientes i = em.find(Ingredientes.class, new Integer(id));
        em.remove(i);
        return "Ingrediente eliminado!.";
    } catch (Exception e) {
        return "Error: " + e;
    }
}
```

13	Frutilla	10		EDITAR	ELIMINAR
16	Papas Rusticas	29		EDITAR	ELIMINAR

Ilustración 38, Pruebas de sistema.

Realizado por	Hernan Gonzalez Diaz.
Objetivo de la prueba	JMeter Prueba de sistema carga y consultas al servicio restfull tatalCuenta y conversionMoneda.
Pre- requisitos	1. Base de datos disponible para hacer transacciones.

	2. Que existan registros de los platos en la base de datos. 3. Servicio restFull disponible en el servidor web.

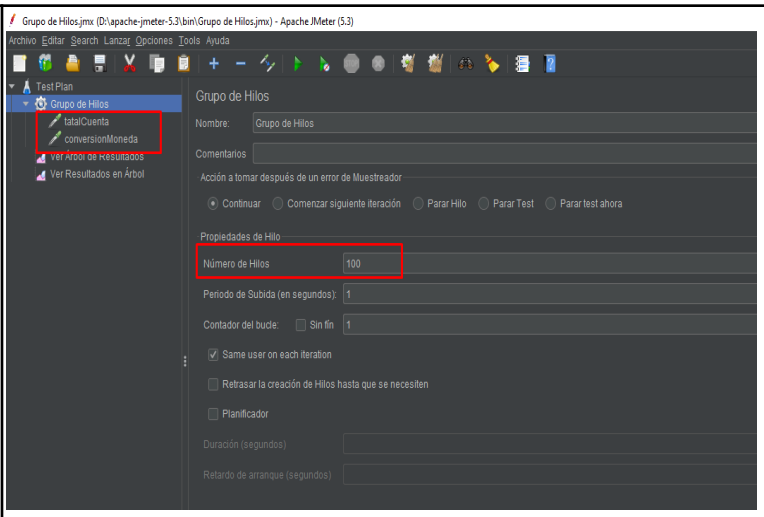
DETALLE CASO DE PRUEBA

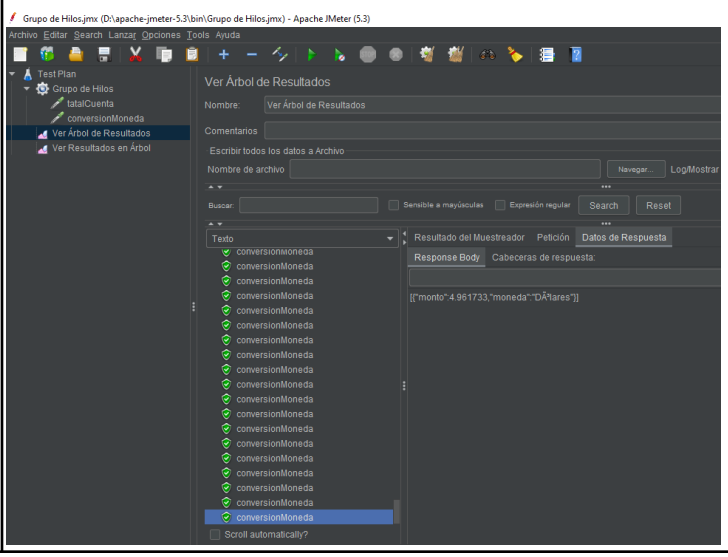
Paso#	Acción Programador	Respuesta esperada del sistema	Resultado
1.1	Ejecutar la prueba con JMeter.	Mensaje que indica que en la respuesta retorna el objeto de conexión de base de datos.	OK

OBSERVACIONES ADICIONALES

La respuesta de la prueba de sistemas tiene tiempos de respuesta adecuados, a continuacion evidencia de la consola de Jmeter

Tiempo utilizado:
1188 ms





Aprobado	Hernan Gonzalez Diaz

Ilustración 39, Pruebas de sistema.

9.4.5 Pruebas de caja Negra

Las pruebas de caja negra, también llamadas pruebas de comportamiento se enfocan en los requerimientos funcionales del software; es decir, las técnicas de prueba de caja negra le permiten derivar conjuntos de condiciones de entrada que revisarán por completo todos los requerimientos funcionales para un programa. Las pruebas de caja negra no son una alternativa para las técnicas de caja blanca. En vez de ello, es un enfoque complementario que es probable que descubra una clase de errores diferente que los métodos de caja blanca.

Las pruebas de caja negra intentan encontrar errores en las categorías siguientes: 1) funciones incorrectas o faltantes, 2) errores de interfaz, 3) errores en las estructuras de datos o en el acceso a bases de datos externas, 4) errores de comportamiento o rendimiento y 5) errores de inicialización y terminación.

La prueba de caja negra tiende a aplicarse durante las últimas etapas de la prueba. Puesto que, a propósito, la prueba de caja negra no considera la estructura de control, la atención se enfoca en el dominio de la información.

Conversión de peso a dólar

Dato de entrada	Salida esperada	Salida obtenida	Estado
$5.000	6,50	6,50	OK

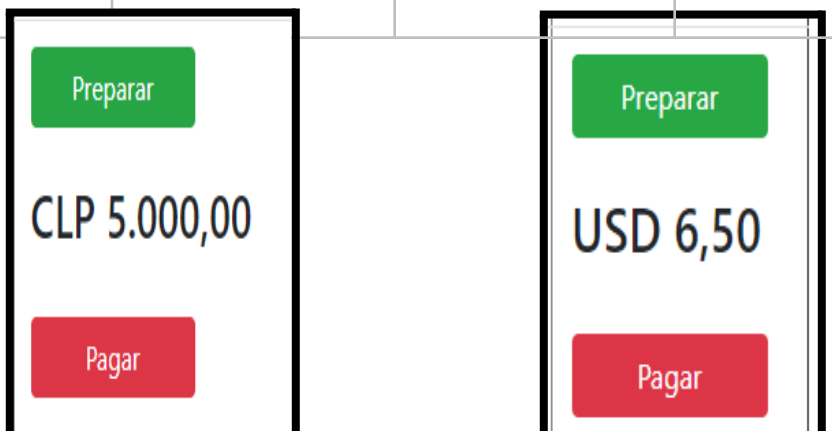


Ilustración 40, Pruebas de caja Negra.

Conversión de peso a euro

			
Dato de entrada	Salida esperada	Salida obtenida	Estado
$5.000	5,75	5,75	OK

Ilustración 41, Pruebas de caja Negra.

Agregar Ingrediente:

Dato de entrada	Salida esperada	Salida obtenida	Estado
Nombre: Lechuga Morada **Stock:** 10 **Imagen de referencia:** https://www.jardineriaon.com/wp-content/uploads/2019/10/Lechuga-morada.jpg	**Nombre:** Lechuga Morada **Stock:** 10 	**Nombre:** Lechuga Morada **Stock:** 10 	OK

Ilustración 42, Pruebas de caja Negra.

Actualizar Stock: Se actualiza el Stock de 10 a 18.

Dato de entrada	Salida esperada	Salida obtenida	Estado
Stock: 10	**Stock:** 18	**Stock:** 18	OK

Ilustración 43, Pruebas de caja Negra.

Realizado por	Hernan Gonzalez Diaz
Objetivo de la prueba	Postman – prueba de consumo el servicio restFull TotalCuenta usando
Pre- requisitos	1. Base de datos disponible para hacer transacciones. 2. Que existan registros de los platos en la base de datos. 3. Servicio restFull disponible en el servidor web. 4. Ingresar el parámetro monto e Id de plato. 5. Método dao ejecuta la transacción a la base de datos para obtener los registros.

DETALLE CASO DE PRUEBA			
Paso#	**Acción Programador**	**Respuesta esperada del sistema**	**Resultado**
1.1	Ejecutar la prueba el programa postman e ingresar el id del plato del menú	1- El mensaje de respuesta es costo del id del plato ingresado como parámetro	OK

OBSERVACIONES ADICIONALES
La respuesta de la caja negra tiene tiempos de respuesta adecuados, a continuacion evidencia de la consola de postman

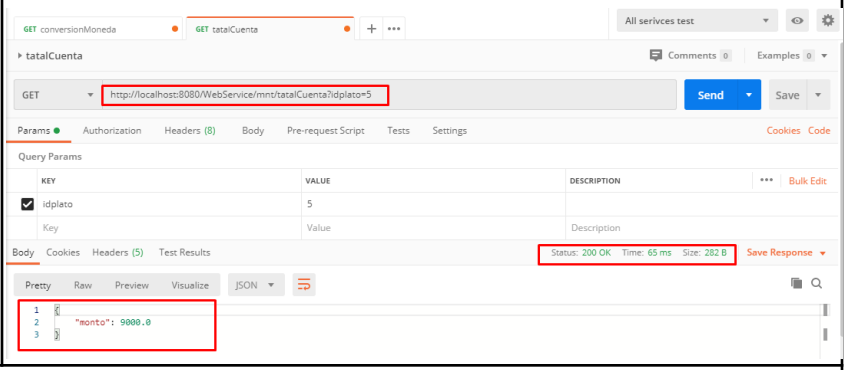
Tiempo utilizado: 65 ms	
Aprobado	Hernan Gonzalez Diaz

Ilustración 44, Pruebas de caja Negra.

9.4.6 Pruebas de Rendimiento.

Las pruebas del sistema se ocupan del comportamiento de un sistema completo. La mayoría de los fallos funcionales deberían haber sido identificados antes, durante las fases de pruebas de unitarias y pruebas de integración.

Realizado por	Hernan Gonzalez Diaz
Objetivo de la prueba	JMeter Prueba de sistema carga y consultas al servicio restfull tatalCuenta y conversionMoneda.
Pre- requisitos	4. Base de datos disponible para hacer transacciones. 5. Que existan registros de los platos en la base de datos. 6. Servicio restFull disponible en el servidor web.

DETALLE CASO DE PRUEBA			
Paso#	**Acción Programador**	**Respuesta esperada del sistema**	**Resultado**
1.1	Ejecutar la prueba con JMeter	Mensaje que indica que en la respuesta retorna el objeto de conexión de base de datos.	OK
OBSERVACIONES ADICIONALES			

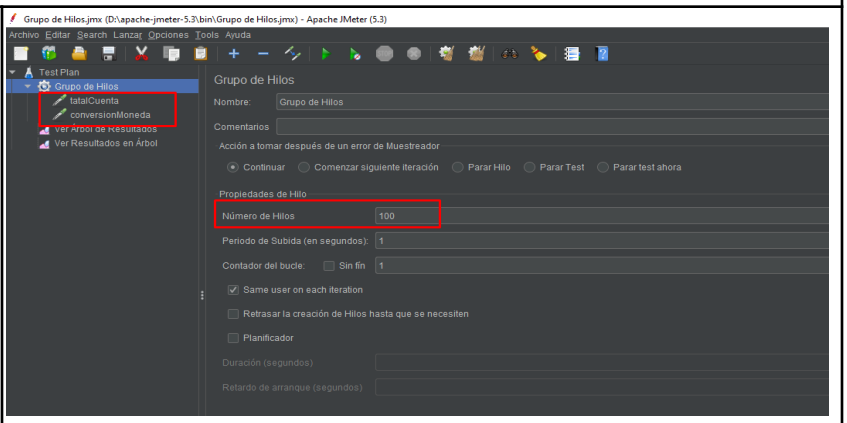
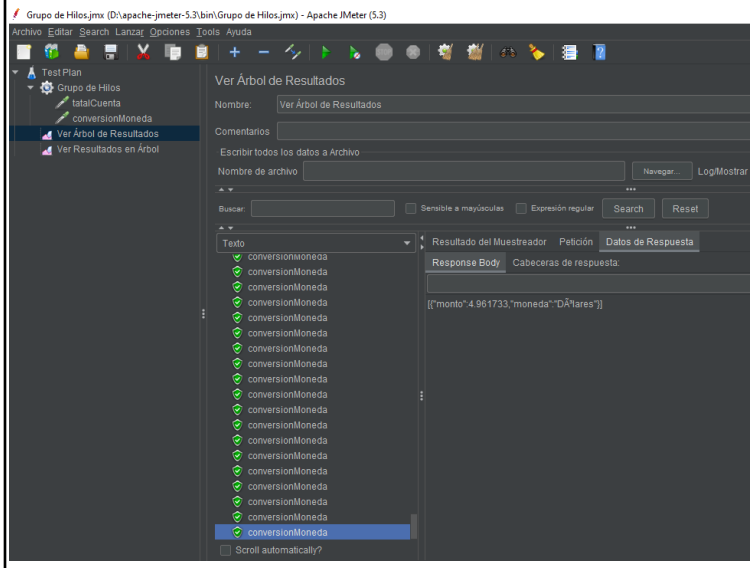
La respuesta de la prueba de sistemas tiene tiempos de respuesta adecuados, a continuacion evidencia de la consola de Jmeter.
Tiempo utilizado: 1188 ms    
Aprobado Hernan Gonzalez Diaz

Ilustración 45, Pruebas de Rendimiento.

Fecha Realización	07/06/2020
Realizado por	Hernan Gonzalez Diaz
Objetivo de la prueba	JMeter Prueba de Carga y Consultas a Base de Datos.
Pre- requisitos	1- Base de datos disponible para hacer transacciones. 2-Que existan registros de los platos en la base de datos.

DETALLE CASO DE PRUEBA			
Paso#	**Acción Programador**	**Respuesta esperada del sistema**	**Resultado**

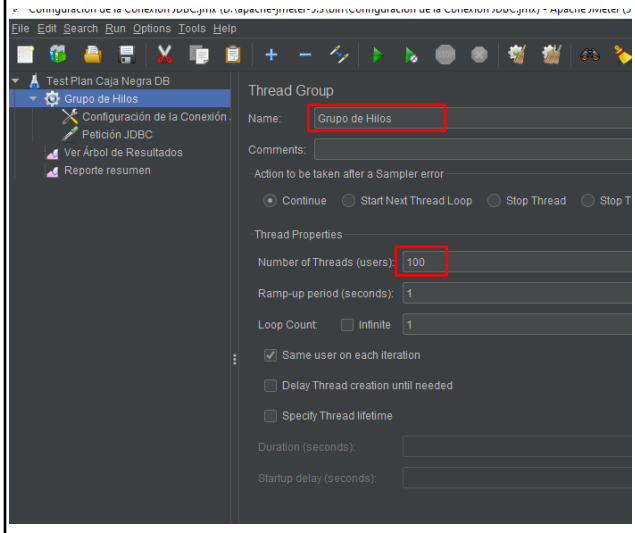
1.2	Ejecutar la prueba con JMeter	El mensaje de respuesta registro de la base de datos	OK

OBSERVACIONES ADICIONALES

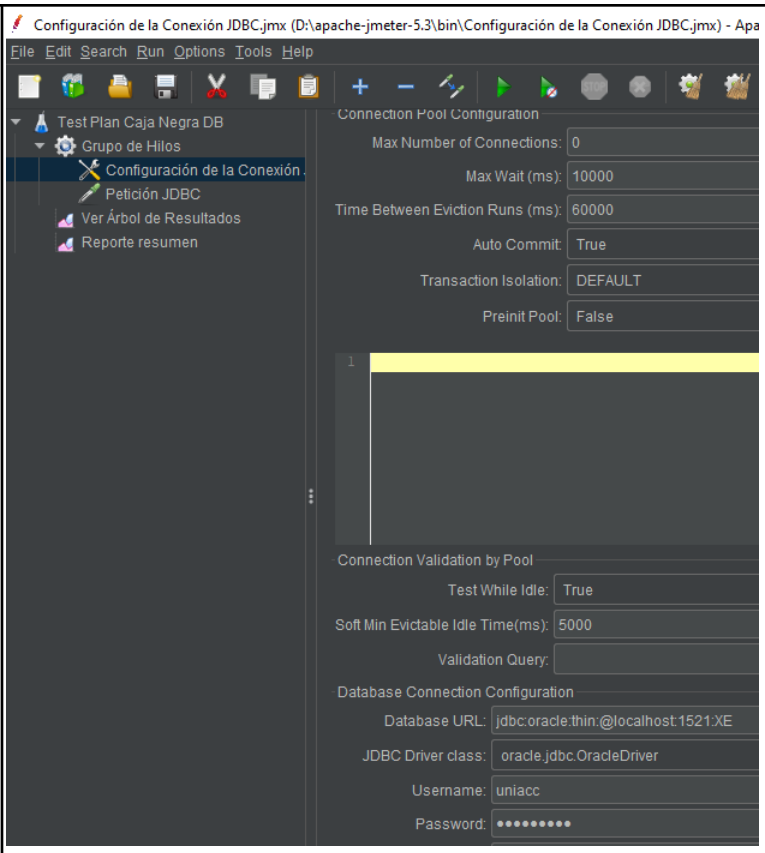
La respuesta de la prueba de caja negra tiene tiempos de respuesta adecuados, a continuacion evidencia de la consola de Jmeter

Rendimiento: 20.6 sec

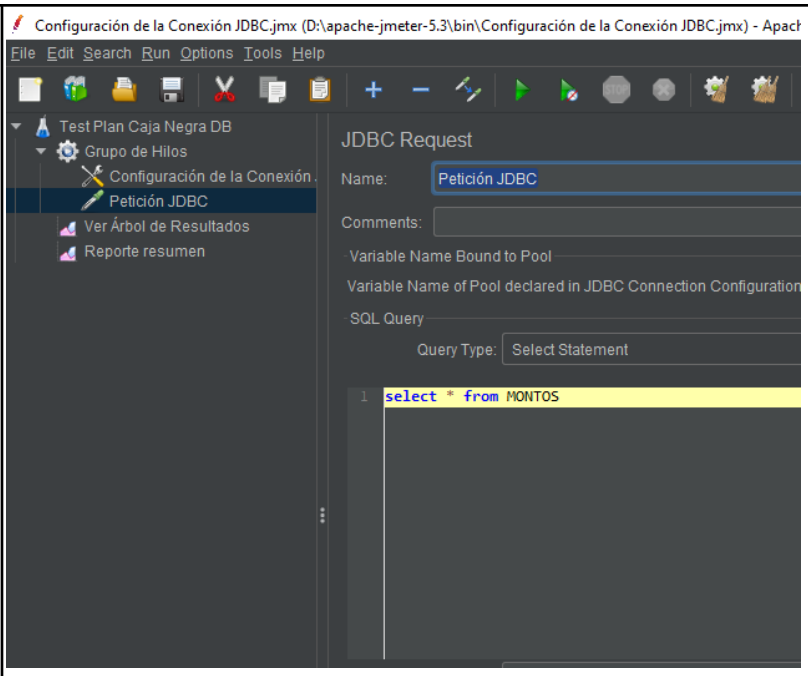
Grupo de Hilos



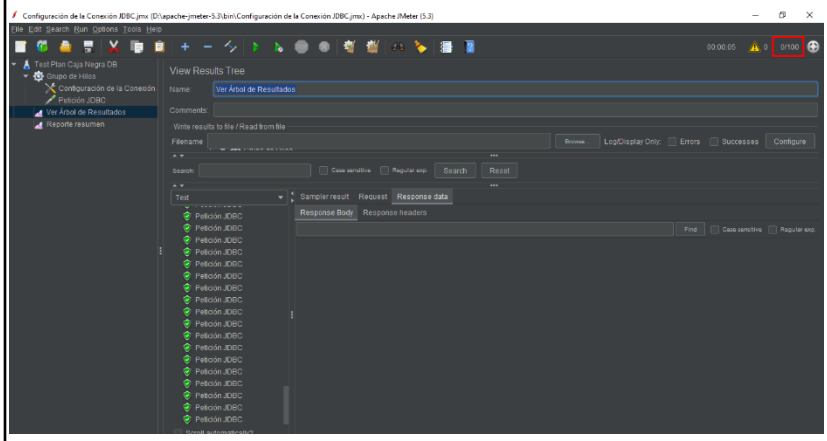
Configuración de la Conexión JDBC



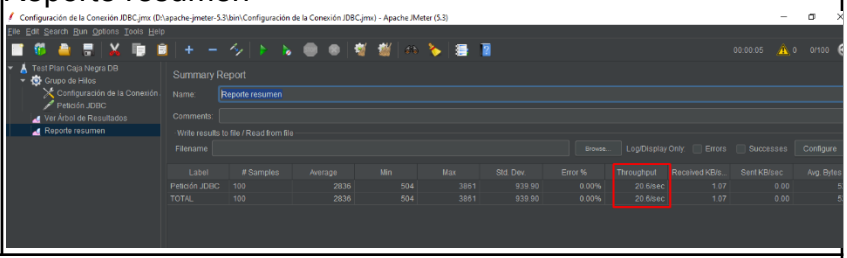
Petición JDBC



Ver Árbol de Resultados



Reporte resumen



Aprobado	Hernan Gonzalez Diaz.

Ilustración 46, Pruebas de Rendimiento.

10 Conclusión.

Con este documento dejamos clara la importancia que tiene el conocimiento de la Ingeniería de Requerimiento y con ella la Gestión de Requisitos. Sin dejar de mencionar que el resultado satisfactorio depende de una intensa comunicación entre clientes y analistas de requerimientos. La Ingeniería se encarga de establecer y mantener un acuerdo en qué el sistema debe hacer, demás proporciona al equipo de desarrollo un entendimiento de los requisitos, hasta definir los límites del sistema.

Para terminar el trabajo, se puede mencionar que fue interesante descubrir lo que buscan los ingenieros al momento de contratar a personal para el proceso de captura de requisitos, además de descubrir cuales son las habilidades que consideran necesarias para el proceso.

Se puede decir que para lograr la calidad en un producto de software es necesario integrar una metodología del proceso de pruebas al ciclo de vida del software, el cual permita realizar las actividades de lo que concierne a la calidad durante todo el proceso que lleva el desarrollo de un software. Esto implica involucrar las pruebas desde las fases de análisis para ir semblanteando el desarrollo de estas, pasando por la codificación donde pueden estar involucradas las pruebas de caja negra dependiendo del tipo de prueba que se vaya a realizar, y así hasta la fase de actualización que permita al software adaptarse a las nuevas necesidades del mercado o errores que no fueron descubiertos durante las fases previas para así garantizar la calidad y la mejora continua del software.

Por medio de un modelo o metodología que garantice la retroalimentación de la información obtenida durante el proceso de pruebas del software se puede asegurar la correcta aplicación de las pruebas de software y del plan de pruebas en el cual estén contempladas. Modelos para la aplicación de pruebas hay muchos, con sus respectivas ventajas y desventajas, pero lo importante al utilizar cualquier modelo, consiste en usar el que mejor se adecue al equipo de trabajo y el que permita alcanzar los objetivos deseados. Aunque esto signifique hacer modificaciones a uno ya establecido, pero siempre siguiéndolo al pie de la letra,

del mismo modo es necesario dejar en claro al equipo de trabajo desde etapas tempranas, como es que se utilizará. Además, nunca se debe olvidar el objetivo principal de lograr una calidad óptima que complazca al usuario final, ya que este será el que interactuará con el software y de quien dependerá que el software se utilice o quede en el abandono.

La aplicación de QA en la Ingeniería de Software aporta un enorme e indiscutido valor, siendo los casos de prueba un pilar fundamental para lograrlo. Automatizando la generación de pruebas se consigue optimizar el testing logrando mejores resultados en cuanto a tiempos y calidad. Una herramienta para dicha generación de casos de prueba debería estar disponible para cada tester en cualquier equipo de testing. La misma debe ser sencilla de utilizar, fácil de configurar y poder obtener resultados rápidos y en un formato consistente con las demás herramientas que dispone un tester en general.

11 Recomendaciones.

Se recomienda elaborar políticas respecto al desarrollo de software, considerando normativas y estándares internacionales, reglamentos y políticas institucionales, el cual menciona; “Adopción, mantenimiento y aplicación de políticas públicas y estándares internacionales para: codificación de software, nomenclaturas, interfaz de usuario, interoperabilidad, eficiencia de desempeño de sistemas, escalabilidad, validación contra requerimientos, planes de pruebas unitarias y de integración.”

Se sugiere realizar un estudio para seleccionar un modelo del proceso de desarrollo de software, una metodología ágil que apoye en el proceso de desarrollo de software, y así lograr construir un software que pueda alcanzar lo que el usuario espera.

Se recomienda que el método de caja Negra sea realizado por el área de desarrollo, ya que las personas adecuadas para la realización de pruebas de este tipo son los desarrolladores, puesto que ellos son los que mejor conocen los componentes internos del sistema, lo que permitirá que se optimice el tiempo de desarrollo del proyecto.

12 Referencias.

Vignolo, J. (2019) Tipos de integración del lado del servidor. Apunte de clase unidad 1, Universidad UNIACC.

Irigoyen, L. (2018). Etapa III del diseño de Base de Datos: Proceso de normalización del Modelo Conceptual. Base de datos. Lea esto primero (Semana 3).

Sbif.cl (s.f.). API CMF Bancos .v3. Recuperado el 15 de Mayo 2020 desde https://api.sbif.cl/index.html

Slim (s.f.). Slim a micro framework for PHP. Recuperado el 14 de Mayo 2020 desde http://www.slimframework.com

Postman (s.f.). The Collaboration Platform for API Development. Recuperado el 14 de Mayo 2020 desde https://www.postman.com

Solé, M.(2019)Pruebas de Caja blanca y Caja Negra. Apunte de clase unidad 1, Proyecto de Integración IV, Universidad UNIACC.

Vignolo, J (2019) Elaboración de Pruebas de Integración. Apunte de clase unidad 3, Proyecto de Integración III, Universidad UNIACC.

Solé, M.(2019)herramienta de pruebas en la web. Apunte de clase unidad 3, Proyecto de Integración IV, Universidad UNIACC.

Cavada J.. (2015). Delitos Informáticos. Chile y legislación extranjera. Octubre, 08,2020, de BNC Sitio web: https://www.camara.cl/verDoc.aspx?prmTIPO=DOCUMENTOCOMUNICACIONCUENTA&prmID=11020#:~:text=En%20Chile%2C%20la%20Ley%20N,el%20avance%20de%20la%20tecnología