****

Índice

[Introducción 1](#_Toc181263697)

[Descripción del proyecto 2](#_Toc181263698)

[Objetivos 2](#_Toc181263699)

[Objetivos Generales 2](#_Toc181263700)

[Objetivos Específicos 2](#_Toc181263701)

[Alcances y limitaciones 3](#_Toc181263702)

[Alcances: 3](#_Toc181263703)

[Limitaciones 3](#_Toc181263704)

[Diseño de base de dato 4](#_Toc181263705)

[Planificación del proyecto 5](#_Toc181263706)

[Organización del equipo de trabajo para el levantamiento de requerimientos 6](#_Toc181263707)

[¿Qué es laravel? 6](#_Toc181263708)

[Hardware 7](#_Toc181263709)

[Software 7](#_Toc181263710)

[Metodología Seleccionada por el equipo desarrollada 8](#_Toc181263711)

[SCRUM 8](#_Toc181263712)

[Historia de Usuarios 9](#_Toc181263713)

[Historial de usuario de pedidos 10](#_Toc181263714)

[Historia de usuario menú 10](#_Toc181263715)

[Historia de usuario ingredientes de los platos 11](#_Toc181263716)

[Planificación por cada sprint (Sprint planning). 12](#_Toc181263717)

[Product backlog general de todo el proyecto. 13](#_Toc181263718)

[Estimación de tiempos por cada actividad. 15](#_Toc181263719)

[Cronograma general de todo el proyecto. 18](#_Toc181263720)

[Organización de actividades según Kanban tablero trello 20](#_Toc181263721)

[Roles del equipo de desarrollo (Organigrama del equipo), actividades asignadas por cada sprint. 23](#_Toc181263722)

[Capturas de pantalla del uso de herramientas colaborativas 28](#_Toc181263723)

[Diagrama de casos de uso y clases 29](#_Toc181263724)

[Caso de uso: Plato 29](#_Toc181263725)

[Caso de uso: Menú 30](#_Toc181263726)

[Caso de uso: Pedido 30](#_Toc181263727)

[Caso de uso: Factura 31](#_Toc181263728)

[Diccionario de caso de uso 32](#_Toc181263729)

[Diccionario de caso de uso de menú 33](#_Toc181263730)

[Diccionario de caso de uso de pedido 35](#_Toc181263731)

[Diccionario de caso de uso de Factura 36](#_Toc181263732)

[Diccionario de caso de uso de Reportes 37](#_Toc181263733)

[Repositorio donde se encuentra alojado el proyecto final del equipo, este debe ser compartido con el docente. 38](#_Toc181263734)

[Pruebas de selenium 38](#_Toc181263735)

[Testing login 38](#_Toc181263736)

[Testing reporte por fecha 39](#_Toc181263737)

[Testing crear factura 41](#_Toc181263738)

[Testing crear pedido 42](#_Toc181263739)

# Introducción

En este documento se está desarrollando un sistema de ventas para restaurantes que permitirá la gestión integral de pedidos, menús y facturación, junto con la generación de informes de ventas y reportes en formato PDF.

Así como en este documento encontraremos la descripción del problema la cual describe el sistema de ventas a elaborar para que el cliente conozca que se está desarrollando así cómo los requisitos del sistema que debe tener para usar el software así mismo mostraremos una pequeña descripción de la funcionalidad de cada pantalla atreves de las historias de usuario la cuales han sido desarrolladas por nuestro equipo de trabajo

# Descripción del proyecto

El Sistema de ventas en restaurantes es una herramienta diseñada para facilitar la gestión de ventas en restaurantes. Con esta plataforma, los restaurantes pueden registrar sus menús, incluyendo detalles como los platos, precios e ingredientes. Además, permite gestionar los pedidos y la facturación de manera eficiente, mejorando la experiencia del cliente. También ofrece la posibilidad de generar informes de ventas, lo que ayuda a los administradores a entender mejor el rendimiento del negocio y tomar decisiones informadas. Por último, el sistema puede crear reportes en formato PDF, lo que facilita el seguimiento y la documentación de las ventas

# Objetivos

## Objetivos Generales

**Optimizar la gestión del restaurante:**

Desarrollar una aplicación que automatice la gestión del menú, pedidos y facturación, mejorando la eficiencia operativa del restaurante.

## Objetivos Específicos

**Implementar un sistema de gestión de menú:**

Crear una funcionalidad que permita registrar, actualizar y organizar los platos del menú, incluyendo detalles como nombre, descripción, precio e ingredientes, para facilitar la navegación y búsqueda.

# Alcances y limitaciones

## Alcances:

1. **Creación de menú del restaurante:**

La aplicación permitirá a los empleados agregar, actualizar y organizar los platos del menú.

1. **Facilidad de Tomar Pedidos:**

Los empleados tendrán la facilidad de usar la aplicación para anotar los pedidos de los clientes.

## Limitaciones

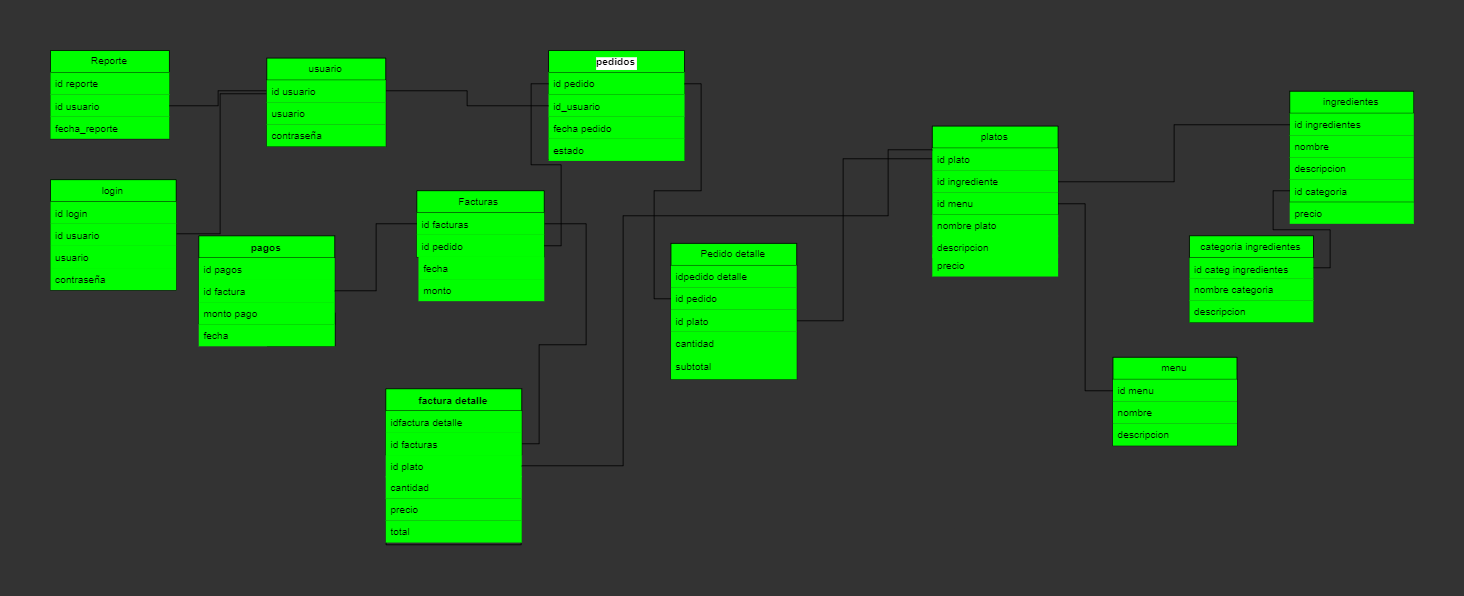
1. **Tiempo:**

Al crear la aplicación contamos con 3 meses de tiempo los que nos limita al crear la aplicación

1. **Dependencia de Internet:**

La aplicación necesitará acceso a Internet para funcionar correctamente así que, sin conexión, no se podrán registrar pedidos ni generar facturas.

# Diseño de base de dato



**Detalles de las relaciones de la base de datos del**

**sistema de menú de restaurante**

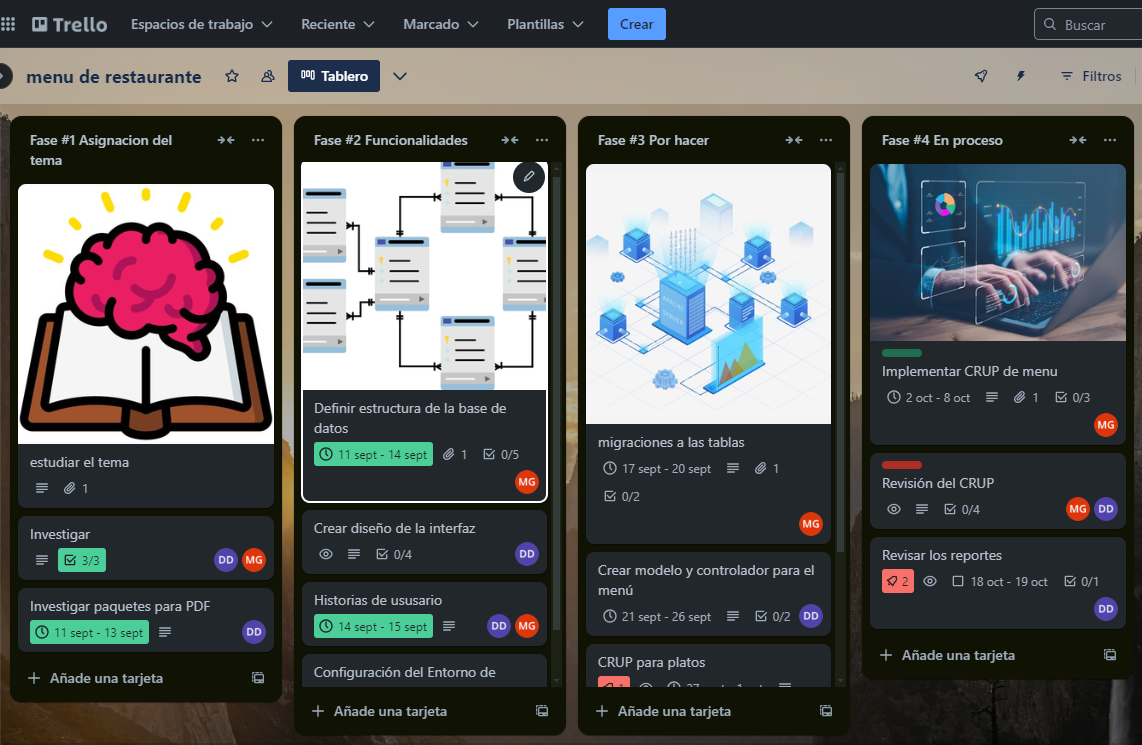
* El Usuario realiza un Pedido.
* Cada pedido se detalla en Pedido Detalle, especificando los Platos incluidos
* Los platos se conectan con los Ingredientes, que a su vez están clasificados bajo una Categoría de Ingredientes.
* El pedido genera una Factura, que tiene su propio Detalle de Factura.
* Finalmente, el Pago se realiza para completar la transacción.

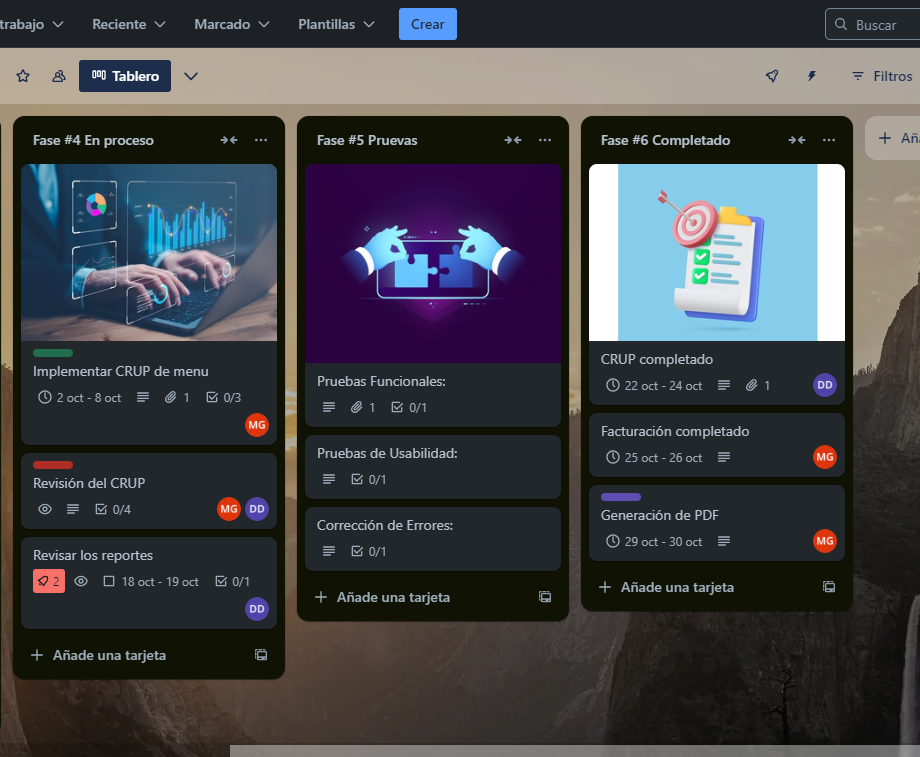
# Planificación del proyecto

En la planificación del proyecto estamos definiendo los objetivos de que se tienen que obtener en este caso se está usando trello para la planificación del sistema de menú de restaurante

**¿Qué es trello?**

Trello es una herramienta visual que permite a los equipos gestionar cualquier tipo de proyecto y flujo de trabajo, así como supervisar tareas. Añade archivos, checklists o incluso automatizaciones: personalízalo todo según las necesidades de tu equipo.





# Organización del equipo de trabajo para el levantamiento de requerimientos

Requerimientos del Sistema para un Proyecto en Laravel.

Descripción de laravel y el lenguaje que se usara

Requerimientos del equipo de Desarrollo

Las especificaciones de la máquina de desarrollo son importantes para asegurar que los desarrolladores puedan trabajar de manera eficiente y sin problemas de rendimiento.

## ¿Qué es laravel?

Laravel es un framework de aplicaciones web con una sintaxis expresiva y elegante que proporciona una estructura y un punto de partida para crear tu aplicación. Laravel es increíblemente escalable gracias a la naturaleza escalable de php y el soporte integrado de laravel para el sistema de cache.

¿Para qué se ocupa laravel?

Se ocupará para el acceso a sus recursos y organiza la lógica de autenticación. De esta manera, los propietarios de sitios Laravel pueden optimizar significativamente la autenticación.

## Hardware

**Procesador**:

* Un Intel i5/i7 o AMD Ryzen 5/7 es una buena opción para manejar el entorno de desarrollo.

**Memoria RAM**:

* 8 GB de RAM como mínimo para un desarrollo fluido. Sin embargo, para proyectos grandes o múltiples servicios corriendo en simultáneo, se recomienda tener 16 GB o más.

**Almacenamiento**:

* Un SSD para tiempos de lectura/escritura rápidos. Un espacio mínimo recomendado sería 256 GB, aunque 512 GB o más es preferible.

## Software

**Sistema Operativo**:

* **Linux** (Ubuntu/Debian) es el entorno preferido para desarrollo web y Laravel. Alternativamente, macOS o Windows 10/11 también son opciones viables.

**Servidor Web Local**:

* XAMPP, MAMP, o Laravel Valet para el manejo de un entorno de servidor local. O directamente instalar Apache o Nginx con PHP.

**PHP lenguaje de programación**:

* Laravel requiere PHP 8.1 o superior. Asegúrate de tenerlo instalado y configurado en el entorno de desarrollo.

**Composer**:

* **Composer**, el gestor de dependencias de PHP, es esencial para gestionar las librerías y paquetes en Laravel.

**Node.js o NPM/Yarn**:

* Laravel usa Node.js para manejar los activos frontend como JavaScript y CSS. Tener Node.js 14+ instalado con NPM o Yarn es necesario para compilar los assets del frontend.

**Base de datos**:

* Es ideal contar con MySQL, PostgreSQL, o SQLite para el entorno de desarrollo.

**Editor de código**:

* Un editor de texto como Visual Studio Code, PHPStorm, o Sublime Text.

# Metodología Seleccionada por el equipo desarrollada

## SCRUM

Beneficios de Scrum para el Desarrollo del Sistema de venta de restaurante

1. Entrega Continua:

- Beneficio: Entregas frecuentes de partes funcionales del sistema permiten feedback temprano y ajustes rápidos.

2. Adaptabilidad:

- Beneficio: Cambios en requisitos se pueden integrar fácilmente gracias a la revisión continua del Producto Backlog.

3. Transparencia:

- Beneficio: Reuniones diarias y revisiones permiten a todos estar al tanto del progreso y problemas.

4. Prioridad en el Valor:

- Beneficio: Funcionalidades más valiosas para los usuarios se desarrollan primero, mejorando la utilidad del sistema.

5. Mejora Continua:

- Beneficio: Reflexiones periódicas del equipo permiten optimizar procesos y resolver problemas de manera constante.

Implementación de Scrum

1. Definir el Producto Backlog:

- Crear una lista de requisitos y priorizar las funcionalidades esenciales del sistema.

2. Planificar el Primer Sprint:

- Seleccionar ítems del backlog para el primer sprint y definir un objetivo claro.

3. Desarrollar y Revisar:

- Trabajar en los ítems del Sprint Backlog y realizar una revisión al final del sprint para obtener feedback.

4. Reflexionar y Ajustar:

- Realizar una retrospectiva para identificar mejoras en el proceso y aplicar ajustes.

5. Repetir el Ciclo:

- Planificar el siguiente sprint basado en el feedback y los resultados obtenidos.

# Historia de Usuarios

## Historial de usuario de pedidos

**Historia de usuario**

**Yo:** como Denis Adonay Diaz Mejia, equipo de SCRUM como usuario final sistema gestión de ventas en restaurantes

**Quiero:** Un sistema de ventas en restaurantes

**Para:** registrar los pedidos de los clientes

**Criterio de Aceptación**

1. Debe contener un formulario donde se registrarán los pedidos
2. El sistema pedirá los siguientes campos Fecha de pedido, usuario y el estado
3. El sistema permitirá realizar el CRUD (crear, actualizar, mostrar y eliminar) los pedidos.
4. El sistema mostrar el pedido del usuario y el total de compra.
5. El sistema mostrara un mensaje si algún campo no está completado
6. Cerrar sesión

## Historia de usuario menú

**Criterio de Aceptación**

1. Debe contener un formulario donde registrarán el menú
2. El sistema pedirá los siguientes campos nombre del menú y descripción
3. El sistema permitirá realizar registrar, eliminar, actualizar y mostrar
4. El sistema mostrar un mensaje si algún campo no está completo.

**Historia de usuario**

**Yo:** Santos Enmanuel Chicas García,

maestro de SCRUM como usuario

final de sistema venta de restaurante

**Quiero:**  Un sistema de venta de restaurante

**Para:** poder Registrar los menús

de los clientes

## Historia de usuario ingredientes de los platos

**Criterio de Aceptación**

1. Debe contener un formulario donde registrarán los ingredientes
2. El sistema pedirá los siguientes campos nombre, descripción, precio
3. El sistema permitirá realizar registrar, eliminar, actualizar y mostrar
4. El sistema mostrar un mensaje si algún campo no está completo.

**Historia de usuario**

**Yo:** Denis Adonay Diaz Mejía,

equipo de SCRUM como usuario

final de sistema venta de restaurante

**Quiero:**  Un sistema de venta de restaurante

**Para:** poder Registrar los ingredientes de los platos

# Planificación por cada sprint (Sprint planning).

**Sprint 1: Diseño de la Base de Datos**

* Diseñar la estructura de la base de datos para menús, pedidos, usuarios y facturación.
* Definir las relaciones entre tablas y asegurar la integridad de los datos.
* Crear las primeras migraciones para generar las tablas necesarias.

**Sprint 2: Implementación de Migraciones y Vistas Iniciales**

* Ejecutar las migraciones y revisar la creación correcta de las tablas.
* Desarrollar las vistas para la gestión de menús y pedidos, incluyendo formularios de registro y visualización de datos.
* Asegurar la interacción correcta entre la base de datos y las vistas.

**Sprint 3: Funcionalidades de Controladores y Modelos**

* Implementar controladores para manejar el CRUD de menús, pedidos y facturación.
* Desarrollar los modelos que interactúan con las tablas correspondientes.
* Asegurar que el flujo de datos entre vistas, controladores y modelos funcione sin errores.

**Sprint 4: Sistema de Autenticación y Control de Accesos**

* Implementar el sistema de login para usuarios administradores.
* Probar el flujo completo de inicio de sesión y acceso a funcionalidades del sistema.
* Asegurar que los controladores y modelos ya implementados estén funcionando correctamente.

# Product backlog general de todo el proyecto.

**Investigación y Recolección de Información**

* Compilar datos importantes sobre las necesidades de los usuarios finales, como administradores y clientes del restaurante.

**Objetivos**

* Establecer los objetivos del sistema en términos de funcionalidad y rendimiento, como la gestión de menús, pedidos, facturación y generación de informes.

**Alcance y Desafíos**

* Especificar el alcance de las funcionalidades principales (menús, pedidos, facturación, informes).

**Estructura de base de datos**

* Diseñar la base de datos para almacenar y gestionar eficientemente los menús, pedidos, clientes, facturación y más.
* Garantizar la consistencia y fiabilidad de los datos en todas las operaciones.

**Organización y Planificación**

* Planificar entregas en ciclos de desarrollo cortos y estructurados (sprints) para una implementación eficiente.

**Asignación de Tareas**

* Definir las responsabilidades del equipo (líder del proyecto, responsables de desarrollo, encargados del producto).

**Requerimientos Funcionales**

* Definir descripciones detalladas de las funcionalidades desde la perspectiva del usuario, como el registro de pedidos, la gestión de menús, y la creación de facturas.

**Planificación en Iteraciones**

* Dividir el trabajo en pequeños ciclos de desarrollo (sprints) con metas específicas y tiempos definidos.

**Historias de usuario**

**Casos de usos**

* lista las tareas que los actores pueden realizar y está directamente relacionado con los requisitos del proceso de negocio.

**Pruebas de Selenium IDE**

* Es una herramienta para realizar pruebas de funcionamiento y detección de errores.

**Lista Prioritaria de Funcionalidades**

* Crear y gestionar una lista priorizada de las tareas y características a desarrollar, basándose en las necesidades más urgentes del sistema.
* Utilizar la lista para definir los objetivos a largo plazo y ajustar las prioridades según avance el proyecto.

**Gestión del Proyecto**

* Utilizar herramientas colaborativas como Trello o Jira para organizar tareas, asignar responsabilidades y hacer seguimiento del progreso.
* Facilitar la coordinación entre el equipo, mejorando la visibilidad de las tareas y el control del cronograma.

# Estimación de tiempos por cada actividad.

Estimación de tiempos para el desarrollo de un sistema de menú de restaurante

**Duración Total:** 12 Semanas  
**Estimación Total de Horas:** 315 horas

**Distribución de Horas Semanales**:

* Días con 5 horas de trabajo: 4 horas durante el día + 2 horas nocturnas = 6 horas/día
* Días con 8 horas de trabajo: 1 hora libre + 2 horas nocturnas = 3 horas/día

**Total, Semanal Estimado:** 24 horas

**Semana 1-2:** Definición de Requisitos (25 horas)

* **Reunión inicial con el cliente:** 5 horas
* **Recopilación de requisitos funcionales y no funcionales:** 10 horas  
  (Menú, sistema de pedidos, facturación, informes de ventas)
* **Documentación de requisitos:** 10 horas

**Semana 3:** Planificación del Proyecto (15 horas)

* **Elaboración de un cronograma detallado:** 5 horas
* **Asignación de roles y responsabilidades:** 5 horas
* **Establecimiento de hitos del proyecto:** 5 horas

**Semana 4-5:** Diseño del Sistema (24 horas)

* **Creación de diagramas de flujo de procesos:** 8 horas  
  (Flujo de pedidos, gestión de menús, facturación)
* **Diseño del modelo de datos (entidad-relación):** 8 horas  
  (Mesas, pedidos, platos, categorías)
* **Definición de la arquitectura MVC:** 8 horas

**Semana 6:** Configuración del Entorno (8 horas)

* **Instalación de Laravel en el servidor local:** 4 horas
* **Instalación de dependencias necesarias (Composer, Laravel):** 4 horas

**Semana 7:** Creación de la Base de Datos (10 horas)

* **Creación de la estructura de la base de datos usando migraciones de Laravel:** 6 horas  
  (Tablas para menú, platos, pedidos, clientes, facturación)
* **Población de la base de datos con datos iniciales (seeders):** 4 horas

**Semana 8-9:** Desarrollo del Modelo (12 horas)

* **Creación de modelos (Platos, Categorías, Pedidos, Facturas):** 6 horas
* **Implementación de relaciones entre modelos**: 6 horas  
  (Relaciones entre platos, categorías, pedidos y clientes)

**Semana 10-11**: Desarrollo del Controlador (12 horas)

* **Creación de controladores para la gestión del menú, pedidos, y facturación:** 6 horas
* **Implementación de la lógica de negocio en los controladores:** 6 horas  
  (Gestión de pedidos, generación de facturas, informes de ventas)

**Semana 12-13:** Desarrollo de la Vista (14 horas)

* **Creación de vistas usando Blade**: 8 horas  
  (Vistas para menú, pedidos, y pantalla de facturación)
* **Implementación de estilos y scripts (Laravel):** 6 horas  
  (Diseño responsivo y elementos visuales del menú)

**Semana 14-15:** Integración y Pruebas (24 horas)

* **Integración de modelo, vista y controlador:** 8 horas  
  (Comprobación del correcto funcionamiento del sistema de pedidos y facturación)
* **Realización de pruebas funcionales y unitarias:** 10 horas
* **Depuración y optimización del código:** 6 horas

**Semana 16:** Revisión de la Configuración de Producción (10 horas)

* P**ruebas finales en el entorno de producción:** 4 horas
* **Subida del sistema al servidor de producción:** 2 horas
* **Configuración del entorno de producción (cachés, base de datos):** 4 horas

**Semana 17:** Creación de Documentación Técnica y de Usuario (12 horas)

* **Documentación y entrenamiento:** 8 horas  
  (Documentación para administradores y personal del restaurante)
* **Capacitación de usuarios (talleres, manuales):** 4 horas

**Semana 18-19:** Monitoreo del Sistema (6 horas)

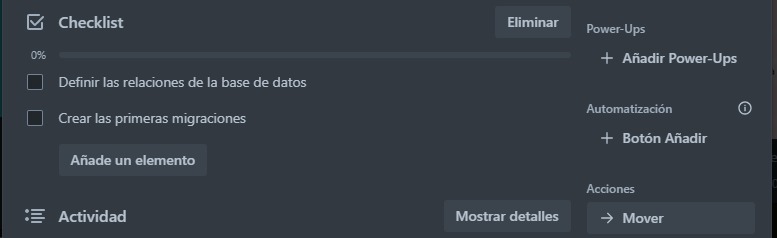
* **Revisiones periódicas:** 3 horas/semana
* **Proveer soporte técnico a usuarios**: 3 horas/semana

# Cronograma general de todo el proyecto.

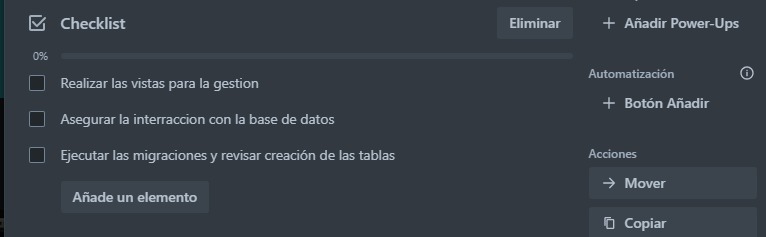
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | Semanas | | | | | | | | | | |
| Actividad | Responsable | 1 | 2 | 3 | 4 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| Fase 1: Planificación | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Definición de Requisitos | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Reunión inicial con el cliente | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Recopilación de requisitos | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| funcionales y no funcionales | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Documentación de requisitos | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Planificación del Proyecto | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Elaboración de un cronograma | Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Detallado Asignación de roles y responsabilidades | Santos García |  |  |  |  |  |  |  |  |  |  |  |
| Establecimiento de hitos del proyecto | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Fase 2: Diseño | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Diseño del Sistema Creación de diagramas de flujo de procesos | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Diseño del modelo de datos (entidad-relación) | Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Definición de la arquitectura MVC (Modelo-Vista-Controlador) | Santos García |  |  |  |  |  |  |  |  |  |  |  |
| Fase 3: Desarrollo del Sistema | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Configuración del Entorno | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Instalación de Laravel en el servidor local | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Instalación de dependencias necesarias (Composer, Laravel) | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Creación de la Base de Datos | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Creación de la estructura de la base de datos usando migraciones de Laravel | Santos García |  |  |  |  |  |  |  |  |  |  |  |
| Población de la base de datos con datos iniciales (seeders) | Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Desarrollo del Modelo | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Creación de modelos (Menús, Platillos, Pedidos) | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Implementación de relaciones entre modelos | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Desarrollo del Controlador | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Creación de controladores con php artisan make | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Implementación de la lógica de negocio en los controladores | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Desarrollo de la Vista | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Creación de vistas usando Blade (formularios, tablas) | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Implementación de estilos y scripts (Laravel para compilación) | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Fase 4: Implementación | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Integración y Pruebas | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Integración de modelo, vista y controlador | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Realización de pruebas funcionales y unitarias | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Depuración y optimización del código | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Fase 5: Despliegue | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Revisión de la configuración de producción en Laravel | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Realización de pruebas finales en el entorno de producción | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Subida del sistema al servidor de producción | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Configuración del entorno de producción (cachés, base de datos) | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Creación de Documentación técnica y de usuario | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Fase 6: Mantenimiento (Continuo) | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Monitoreo del sistema (revisiones periódicas) | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |
| Proveer soporte técnico a usuarios | Santos García/Adonay Diaz |  |  |  |  |  |  |  |  |  |  |  |

# Organización de actividades según Kanban tablero trello

**Base de datos**



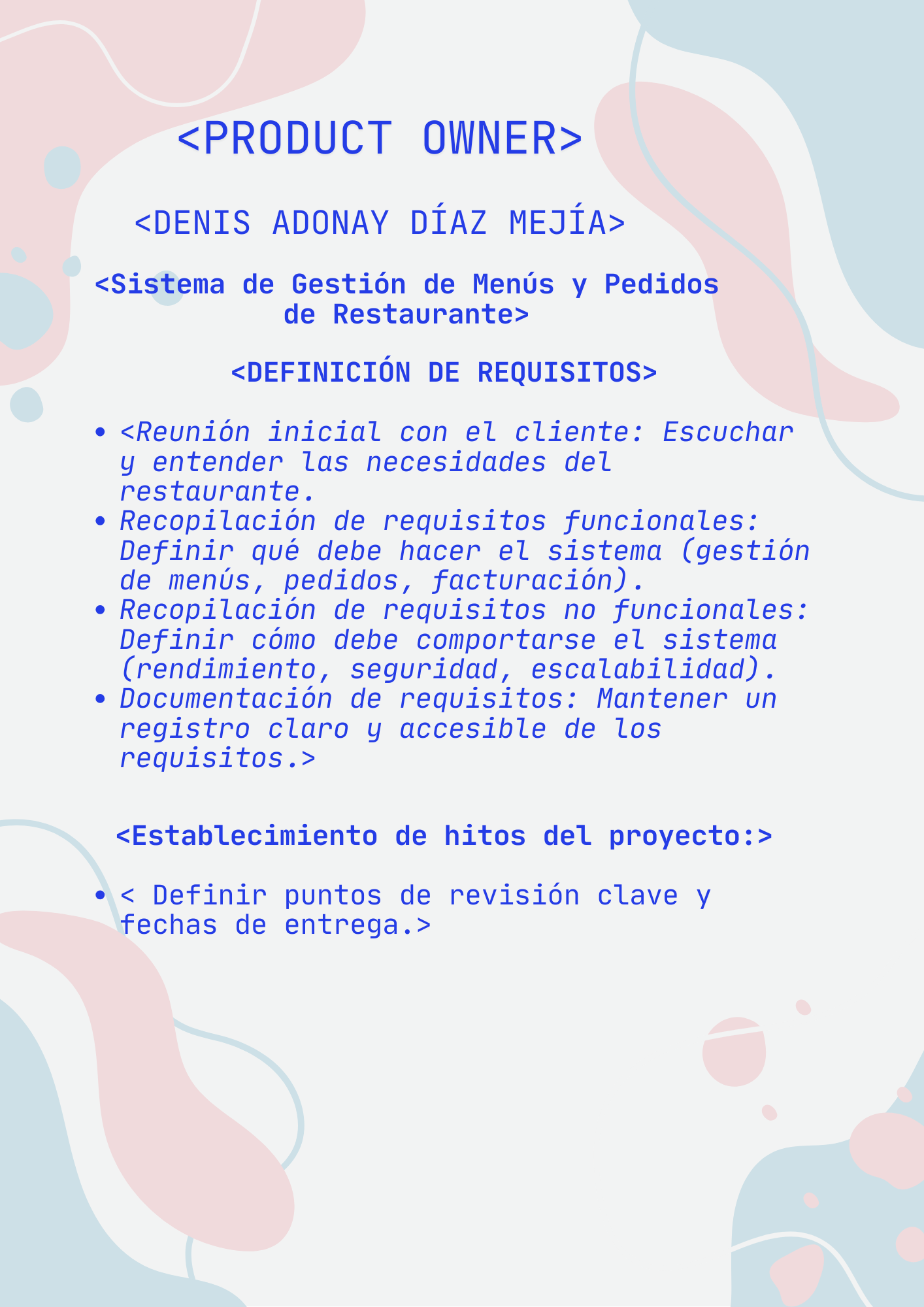
**Migraciones**

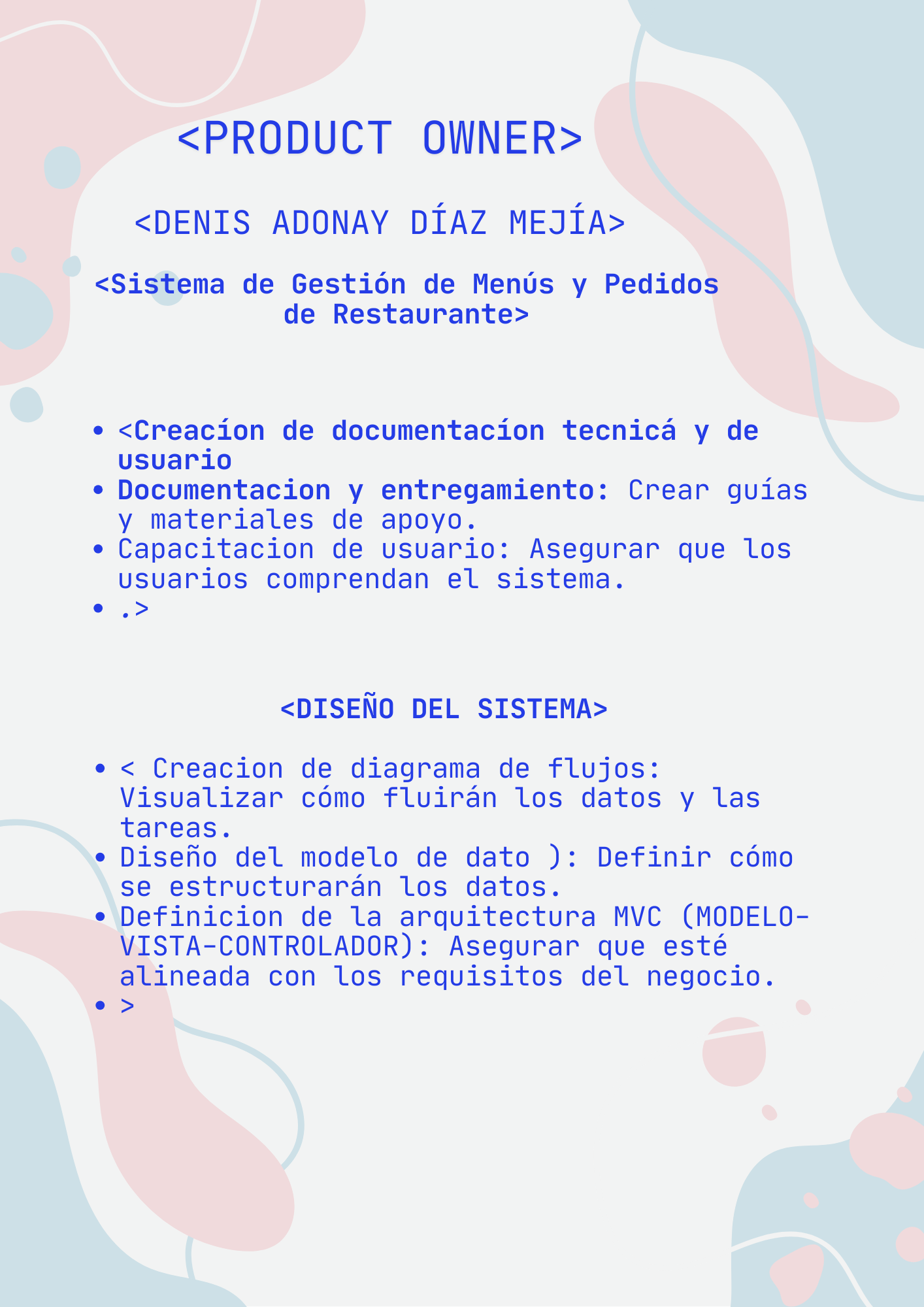


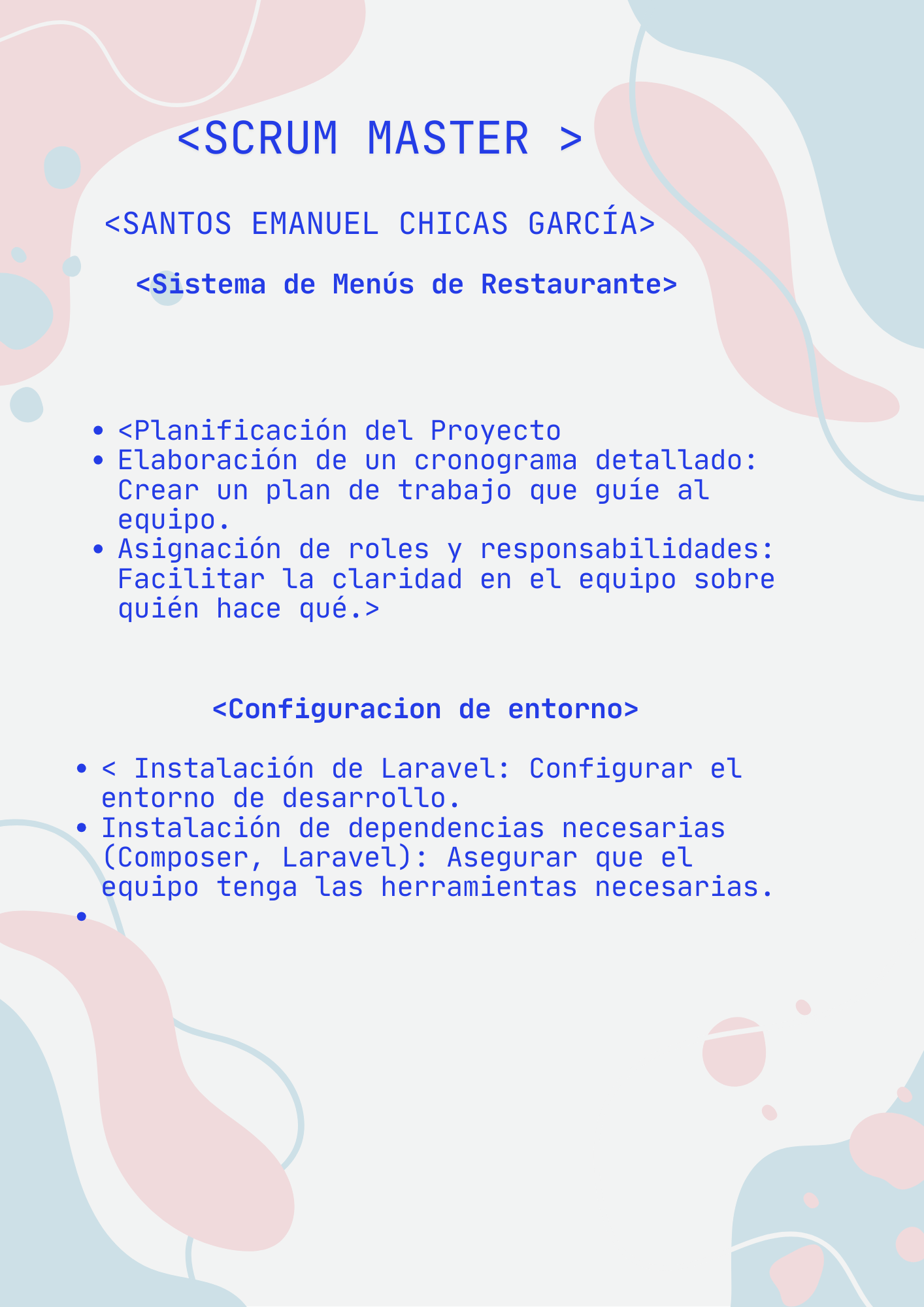
**Funcionalidades y Controladores**

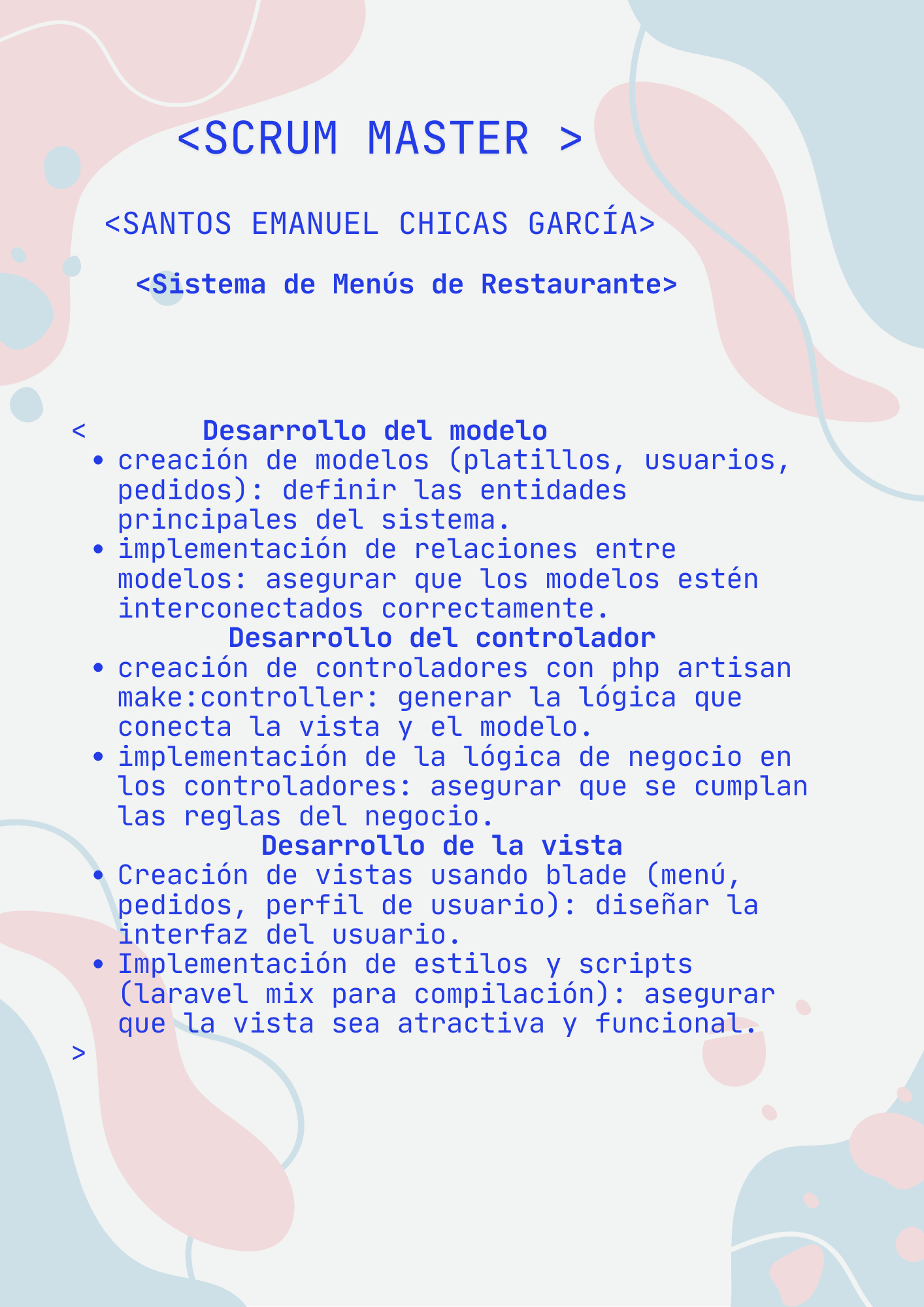


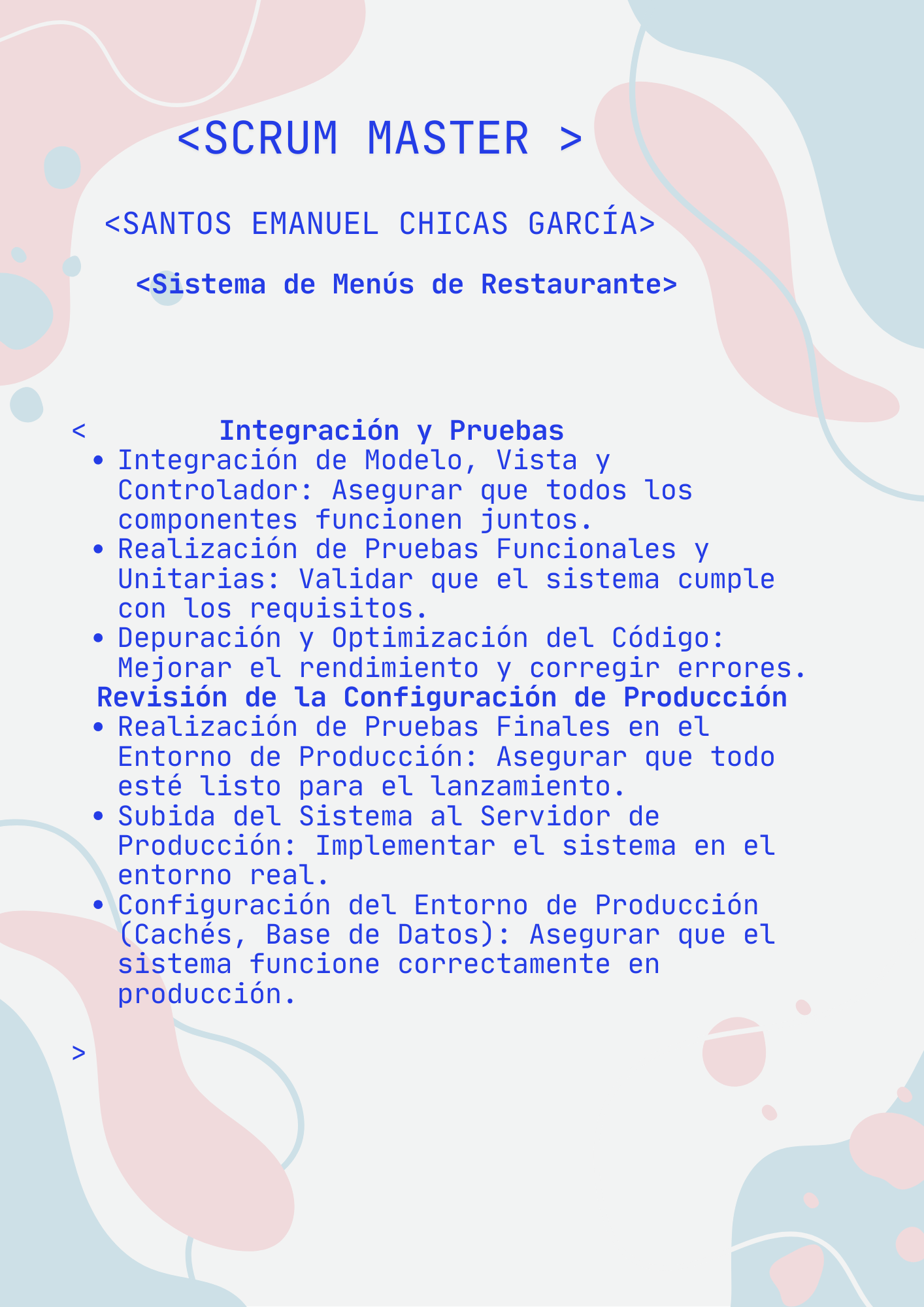
# Roles del equipo de desarrollo (Organigrama del equipo), actividades asignadas por cada sprint.





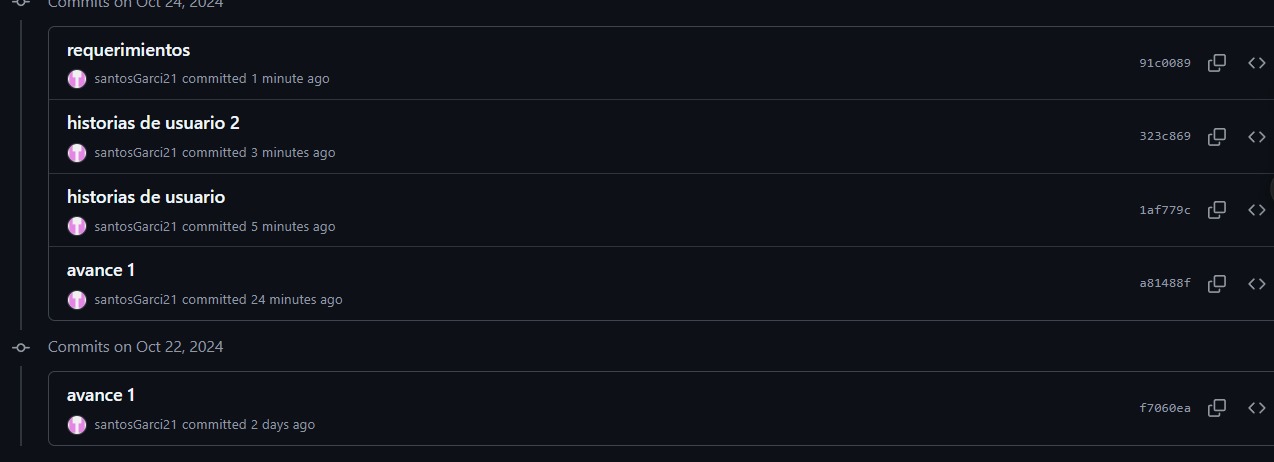






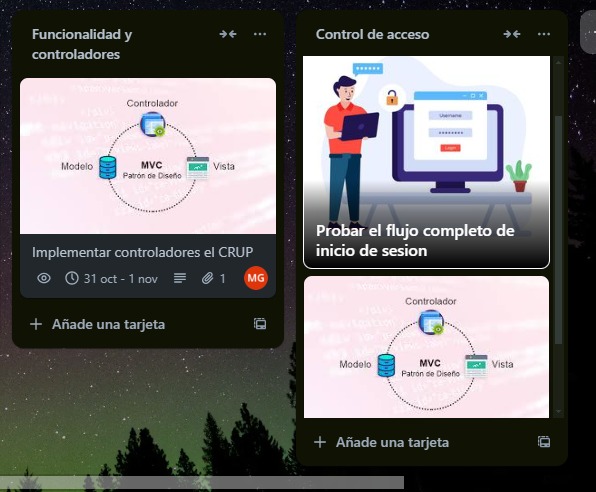
# Capturas de pantalla del uso de herramientas colaborativas

**GitHub**



**Trello**





# Diagrama de casos de uso y Diccionario de clases

## Caso de uso: Plato

Gráfico, Gráfico radial

Descripción generada automáticamente

# Diccionario de caso de uso: Plato

|  |  |
| --- | --- |
| Nombre: | Guardar plato |
| Autor: | Restaurante |
| Fecha: | 13/10/2024 |
| Descripción: El sistema permite al usuario agregar, editar o eliminar platos en el menú del restaurante. Cada plato debe tener un nombre, descripción y precio. El usuario con los permisos adecuados puede gestionar esta información a través de la interfaz web. | |
| Actores: Usuario administrador del restaurante | |
| Pre-condiciones:  El usuario debe estar registrado y autenticado en el sistema con permisos de administrador. | |
| Flujo Normal   1. El usuario accede al sistema a través de la interfaz web. 2. El usuario selecciona la opción de "guardar Platos" en el menú principal. 3. El sistema muestra le muestra el formulario para llenar del plato. 4. El usuario introduce el nombre del plato, su descripción y el precio. 5. El usuario selecciona la opción para guardar un nuevo plato. 6. El sistema valida que todos los campos obligatorios están completos y el formato de los datos es correcto. 7. El sistema guarda la información en la base de datos. 8. El sistema confirma la operación y muestra un mensaje de éxito. | |
| Flujo alterativo:  Si el usuario no completa los campos requeridos o introduce un precio con formato incorrecto (por ejemplo, letras en lugar de números), el sistema muestra un mensaje de error y solicita corregir los datos.  Flujo Alternativo 2  Si el usuario desea eliminar un plato:   1. El usuario selecciona el plato que desea eliminar. 2. El sistema solicita confirmación para eliminar el plato. 3. El usuario confirma la eliminación. 4. El sistema elimina el plato de la base de datos y muestra un mensaje de éxito. | |
| Excepciones:   * Si el usuario no tiene permisos suficientes para gestionar los platos, el sistema muestra un mensaje de error y deniega el acceso. * Si ocurre un error en la conexión a la base de datos, el sistema muestra un mensaje de error y solicita al usuario intentarlo más tarde. | |
| Pre-condiciones: El sistema guarda las opciones del menú y almacena la información en la base de datos de forma correcta. | |

## Caso de uso: Menú

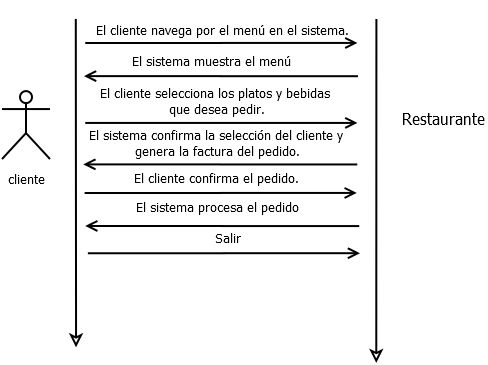
Gráfico, Gráfico radial

Descripción generada automáticamente

## Diccionario de caso de uso de menú

|  |  |
| --- | --- |
| Nombre: | Guardar menú |
| Autor: | Restaurante |
| Fecha: | 13/10/2024 |
| Descripción: El sistema permite al usuario gestionar las opciones del menú de un restaurante. Un usuario autorizado puede agregar, editar o eliminar opciones del menú, así como modificar la descripción de los platillos y su precio. | |
| Actores: Usuario administrador del restaurante | |
| Pre-condiciones: El usuario debe estar registrado en el sistema | |
| Flujo normal:   1. EL usuario selecciona la opción de registrar el menú. 2. El sistema le muestra los campos a registrar en un formulario. 3. El usuario digita los campos nombre, descripción. 4. EL usuario guarda los campos. 5. EL sistema lo redirige a la vista de mostrar donde ya están guardados | |
| Flujo alterativo:  Si el usuario intenta agregar un platillo sin completar los campos requeridos (nombre, descripción, precio), el sistema muestra un mensaje de error y solicita que complete los campos obligatorios.  Flujo Alternativo 2  Si el usuario desea eliminar una opción del menú:   1. El usuario selecciona la opción de "Eliminar". 2. El sistema solicita confirmación para proceder con la eliminación. 3. El usuario confirma la eliminación. 4. El sistema elimina el platillo de la base de datos y muestra un mensaje de éxito. | |
| Excepciones:   * Si el usuario no tiene los permisos necesarios, el sistema muestra un mensaje de error y deniega el acceso * Si se produce un error en la conexión a la base de datos, el sistema notifica el error y solicita reintentar más tarde | |
| Pre-condiciones: El sistema guarda las opciones del menú y almacena la información en la base de datos de forma correcta. | |

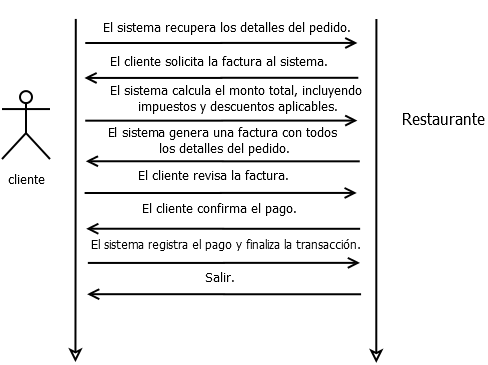
## Caso de uso: Pedido



## Diccionario de caso de uso de pedido

|  |  |
| --- | --- |
| Nombre: | Pedido |
| Autor: | Restaurante |
| Fecha: | 13/10/2024 |
| Descripción: El usuario realiza un pedido seleccionando platos del menú disponible, y el sistema procesa y gestiona la orden. | |
| Actores: Cliente, Sistema | |
| Pre-condiciones: El cliente tiene acceso al menú del restaurante y está listo para hacer su pedido | |
| Flujo normal:   1. El cliente navega por el menú en el sistema. 2. El sistema muestra el menú 3. El cliente selecciona los platos y bebidas que desea pedir. 4. El sistema confirma la selección del cliente y genera la factura del pedido. 5. El cliente confirma el pedido. 6. El sistema procesa el pedido 7. Salir | |
| Flujo alterativo:  Si un plato no está disponible:   1. El sistema notifica al cliente sobre la falta de disponibilidad. 2. El cliente puede optar por elegir un plato alternativo o cancelar el pedido. | |
| Excepciones:   1. Platos agotados (el sistema notifica al cliente). 2. Error en la confirmación del pedido (el sistema solicita al cliente que lo intente nuevamente). | |
| Pre-condiciones: El pedido se registra en el sistema y se inicia el proceso de preparación en el restaurante | |

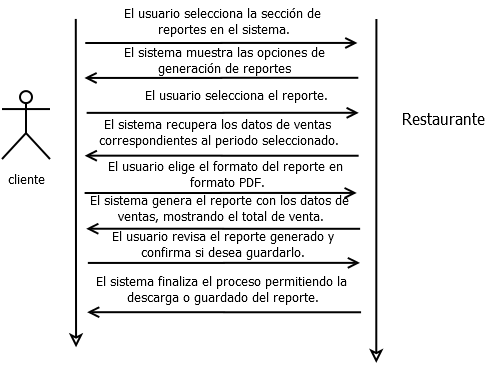
## Caso de uso: Factura

****

## Diccionario de caso de uso de Factura

|  |  |
| --- | --- |
| Nombre: | Factura |
| Autor: | Restaurante |
| Fecha: | 13/10/2024 |
| Descripción: El sistema genera una factura detallada basada en el pedido realizado por el cliente, incluyendo los platos, bebidas, impuestos y descuentos aplicables. | |
| Actores: Cliente, Sistema | |
| Pre-condiciones: El pedido ha sido completado y servido. | |
| Flujo normal:   * 1. El cliente solicita la factura al sistema.   2. El sistema recupera los detalles del pedido.   3. El sistema calcula el monto total, incluyendo impuestos y descuentos aplicables.   4. El sistema genera una factura con todos los detalles del pedido.   5. El cliente revisa la factura.   6. El cliente confirma el pago.   7. El sistema registra el pago y finaliza la transacción.   8. Salir. | |
| Flujo alterativo 1:  Error en el cálculo de la factura   1. El cliente solicita la factura. 2. El sistema recupera los detalles del pedido. 3. El sistema detecta un error en los datos del pedido. 4. El sistema notifica al cliente y solicita una corrección. 5. El cliente o el administrador corrige los datos. 6. El sistema recalcula el total y genera una nueva factura. 7. El cliente revisa la nueva factura y confirma el pago. 8. Salir. | |
| Excepciones:   1. Error en la conexión del sistema para generar la factura (el sistema notifica al cliente para intentar nuevamente). 2. Cliente solicita un cambio o ajuste antes de confirmar el pago (se recalcula la factura y se emite una nueva). | |
| Pre-condiciones:   1. El pedido ha sido completado y registrado en el sistema. 2. El sistema de facturación está operativo y accesible. | |

**Caso de uso: Reporte**

****

## Diccionario de caso de uso de Reportes

|  |  |
| --- | --- |
| Nombre: | Reportes |
| Autor: | Restaurante |
| Fecha: | 13/10/2024 |
| Descripción: El sistema genera un reporte detallado de las ventas del restaurante, incluyendo la información de los pedidos, | |
| Actores: | |
| Pre-condiciones:   * Existen ventas registradas en el sistema para el periodo seleccionado. * El sistema de reportes está operativo y accesible. * Los datos de ventas están actualizados y sin errores. * El usuario tiene permisos para generar reportes. | |
| Flujo normal:   1. El usuario selecciona la sección de reportes en el sistema. 2. El sistema muestra las opciones de generación de reportes 3. El usuario selecciona el reporte. 4. El sistema recupera los datos de ventas correspondientes al periodo seleccionado. 5. El usuario elige el formato del reporte en formato PDF. 6. El sistema genera el reporte con los datos de ventas, mostrando el total de venta. 7. El usuario revisa el reporte generado y confirma si desea guardarlo. 8. El sistema finaliza el proceso permitiendo la descarga o guardado del reporte. | |
| Flujo alterativo:  No hay ventas registradas en el reporte seleccionado   1. El usuario selecciona la opción de generar reporte. 2. El usuario define el reporte. 3. El sistema detecta que no hay ventas registradas en ese periodo. 4. El sistema notifica al usuario que no existen datos para generar el reporte. 5. El usuario elige un nuevo periodo o sale del sistema. | |
| Excepciones:   * Error en la conexión o acceso a los datos de ventas (el sistema notifica al usuario e intenta nuevamente). * El reporte generado contiene errores o datos faltantes (el sistema notifica y solicita revisar los datos) | |
| Pre-condiciones:   * Toda información ya a sido completado para generar el reporte | |

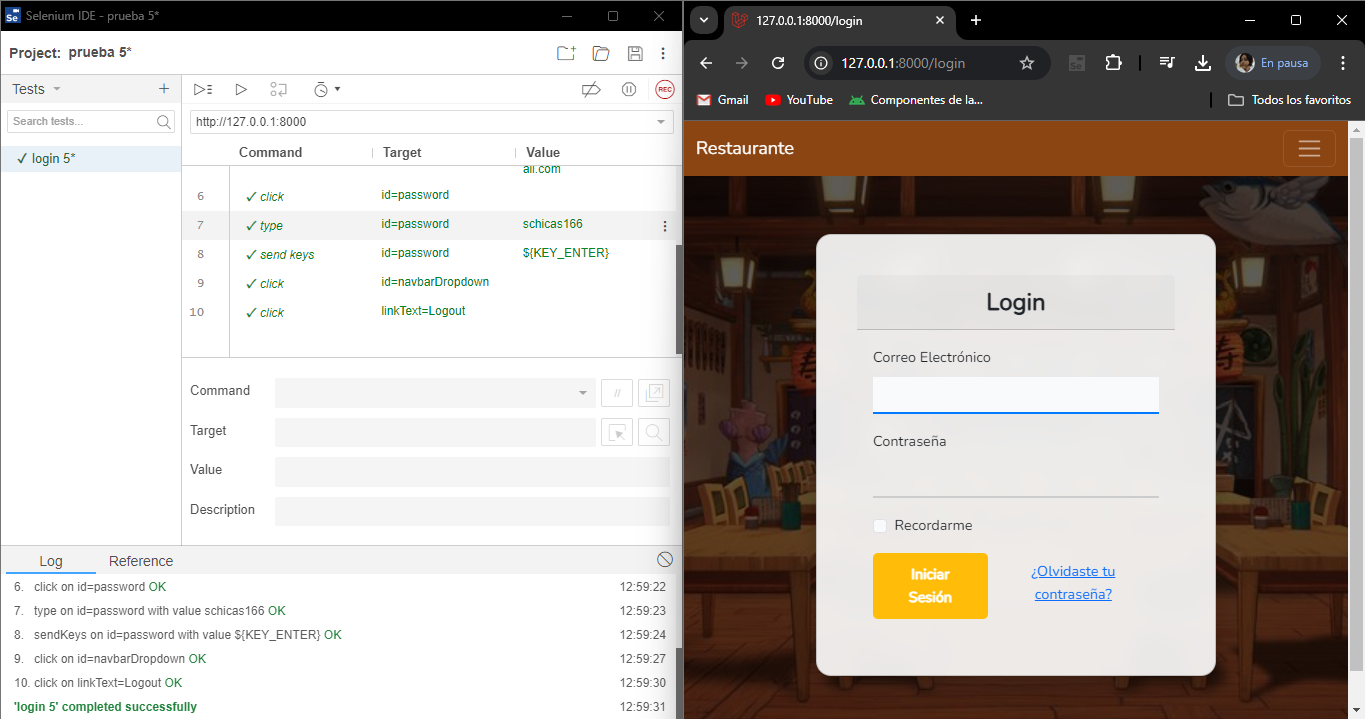
# Diccionario de caso de uso

# Repositorio donde se encuentra alojado el proyecto final del equipo, este debe ser compartido con el docente.



# Pruebas de selenium

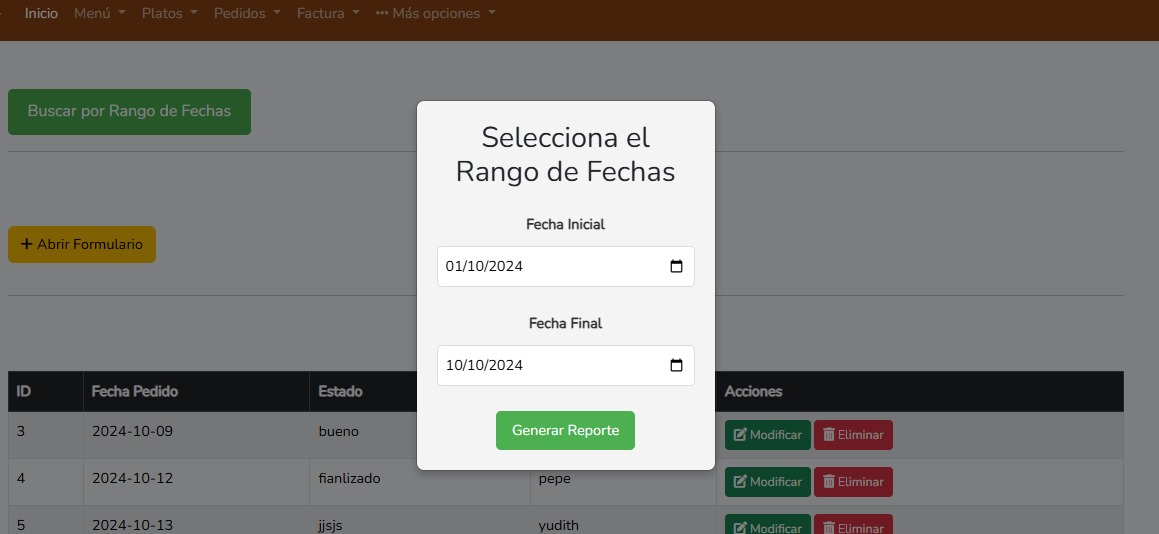
## Testing login

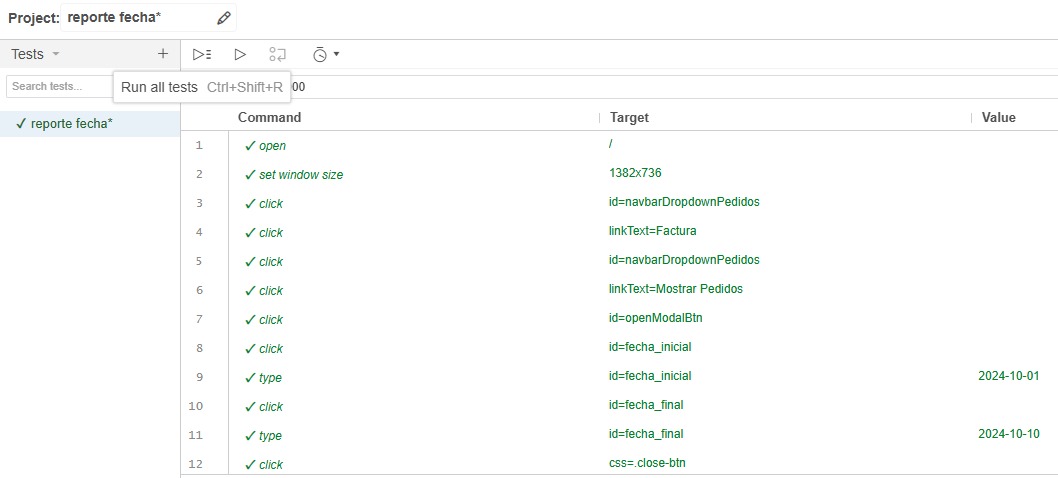


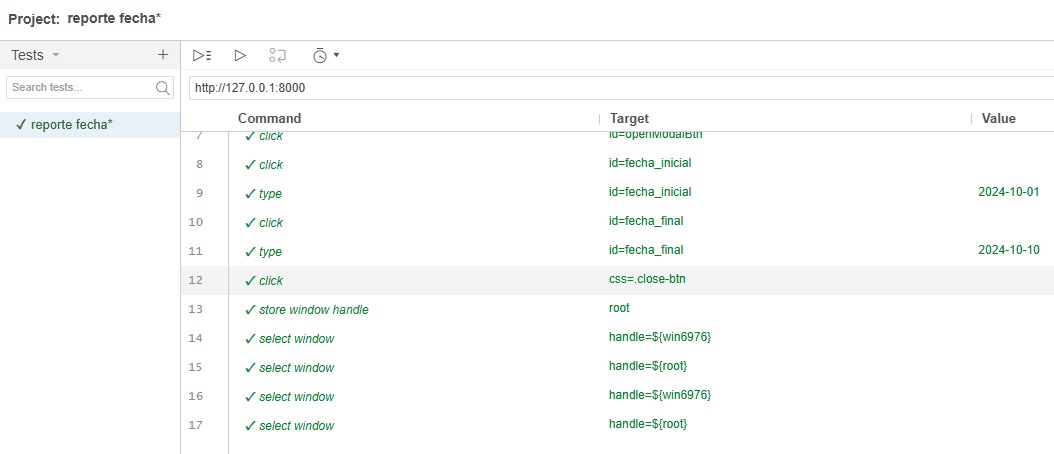
**Explicación**

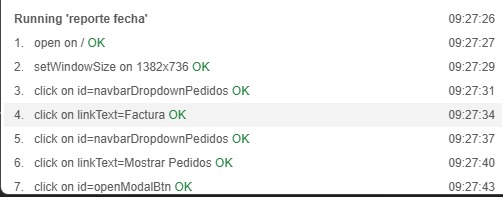
realiza una prueba de inicio de sesión: hace clic en el campo de correo electrónico y escribe "memegarcia195@gmail.com", luego hace clic en el campo de contraseña e ingresa "schicas166". Finalmente, envía la tecla "Enter" en el campo de contraseña para iniciar sesión. Cada acción se ejecuta con éxito, como se indica con "OK".

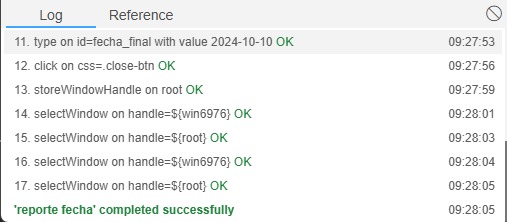
## Testing reporte por fecha









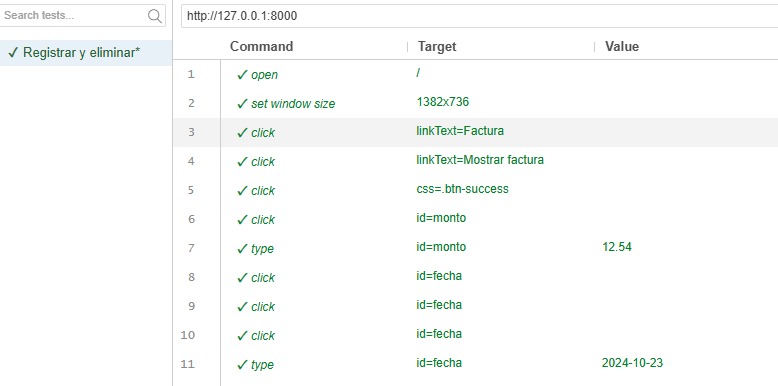


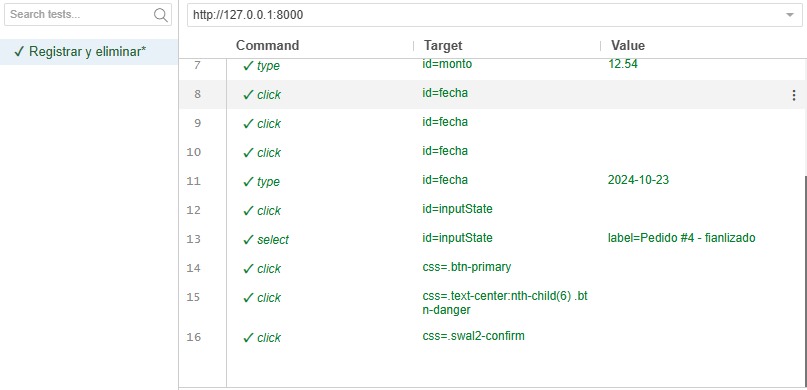
**Explicación**

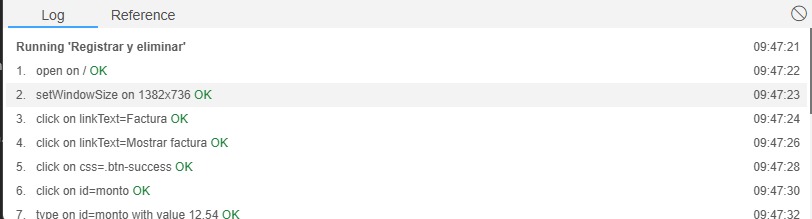
En el siguiente test comienza abriendo la página principal, ajustando el tamaño de la ventana y navegando por opciones del menú, como "Factura" y "Mostrar Pedidos". Luego, interactúa con un modal y selecciona una fecha inicial y final para mostrar el reporte de fechas por rango. También, realiza cambios de enfoque entre ventanas para simular el comportamiento del usuario en varias pestañas. Al final, la prueba concluye exitosamente, confirmando que todas las interacciones funcionan correctamente.

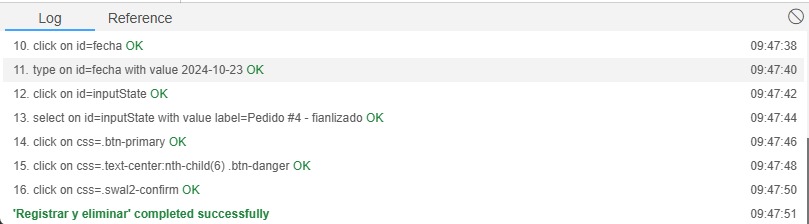
## Testing crear factura







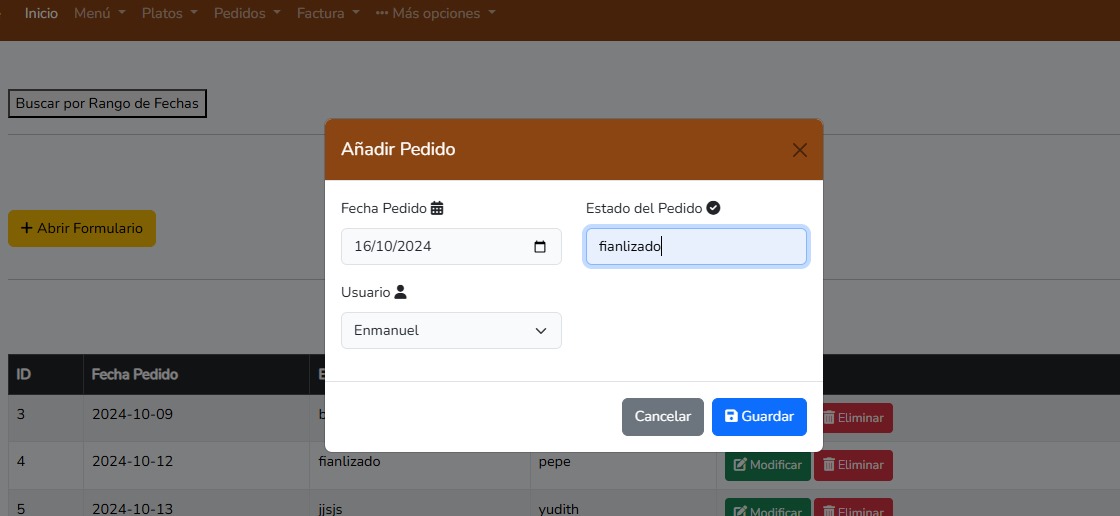


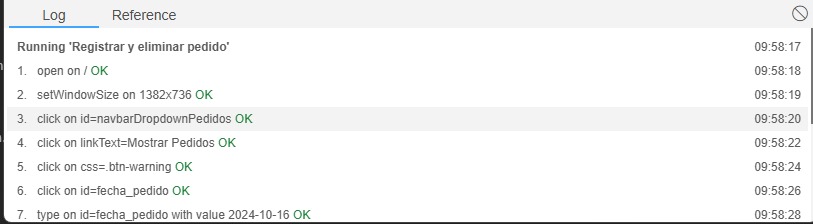
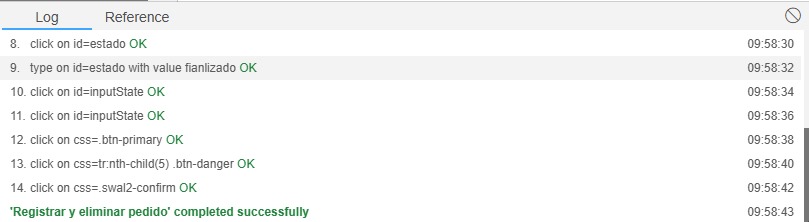
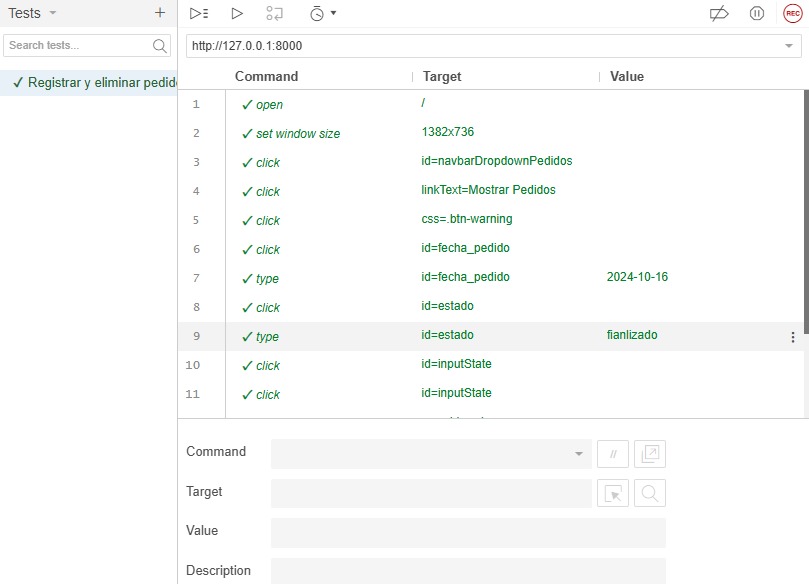


**Explicación**

En esta prueba incluye pasos como abrir la aplicación, seleccionar una factura, ingresar un monto, establecer una fecha, seleccionar un estado de pedido, y confirmar la eliminación. Todos los pasos fueron ejecutados correctamente, con un mensaje final de éxito.

## Testing crear pedido





El testeo "Registrar y eliminar pedido" verifica exitosamente las acciones de abrir la aplicación, ajustar la ventana, interactuar con los menús y botones, y eliminar un pedido en una aplicación web. Ambas partes del testeo se completan sin errores, asegurando que el sistema funcione correctamente y de manera eficiente.