Regression Test Documentation Cartas Digitales

Regression Test

El Regression Test tiene como objetivo:

Verificar que las funcionalidades existentes siguen comportándose igual que antes de los cambios.

Detectar errores introducidos por optimizaciones, refactorizaciones o correcciones de bugs.

Asegurar que las modificaciones no reintroduzcan fallos antiguos (reversiones de fallos).

Confirmar que las correcciones de defectos recientes no afecten negativamente a otras partes del sistema.

Mantener la estabilidad general del software a lo largo del tiempo.

Pasos para realizar un test de regresión en QA.

1. **Definir alcance:**

Módulos a trabajar en el test de regresión: **Platos, Cartas, Canales, Publicidad, Qr.**

2. funcionalidades y escenarios cubrir:

- ☐ Filtrado de platos por Restaurante.
- ☐ Lógica de programación de cartas en el sistema.sección de creación de cartas/secciones.
- ☐ Sección de canales/programación de la carta.
- ☐ Programación de cartas por restaurante.
- ☐ Generación de código QR.
- ☐ Genere la carta de ajuste por defecto.
- ☐ La plantillas de las cartas.

☐ Funcionalidad de cartas inactivas en la sección de canales.

☐ Ajustar la columna de duración en el DAM de actualización de imagen en el administrador de archivoTraducción de secciones al idioma seleccionado Español.

☐ Implementación de Carpetas DAM.

3. Identificar riesgos y áreas críticas.

☐ Sección de crear carta/Platos.

☐ Validación en los tipos de idioma.

☐ En el modal de detalle del plato

☐ Carpetas DAM.

☐ Validación de la programación de cartas.

☐ Carga de imagenes en los platos.

4. Priorizar casos de prueba basados en impacto y probabilidad de fallo.

ID de la incidencia	Resumen	Prioridad	Tipo
TCD-137	Revisar el Responsivo (iPad Mini, Air, Pro) en las cartas generadas por código QR	Normal	Error
TCD-128	Problema en la sección de crear carta, actualmente no permite avanzar debido a una validación en los tipos de idioma.	Crítico	Error
TCD-127	Agregar una advertencia (warning) que se active cuando se cargue un video en el DAM.	Mayor	Error
TCD-120	Modificar el formato de fecha en los diferentes módulos.	Normal	Error
TCD-119	Alinear correctamente los botones de búsqueda importar y nuevo plato, y los filtros de búsqueda en todos los módulos	Normal	Error
TCD-116	Modificar el texto del botón en el modal de traducciones dentro del formulario de detalles del plato	Normal	Error
TCD-111	Revisar y corregir la visualización de la plantilla "Menú diario"	Normal	Error
TCD-110	Corregir errores en la traducción de cartas al idioma inglés en el menú	Normal	Error
TCD-109	Corregir advertencia en el menú de código QR al solicitar otro idioma (inglés)	Normal	Error
TCD-108	Corregir generación de código QR para que genere la carta de ajuste por defecto	Normal	Error

5. Establecer criterios de aceptación y umbrales de calidad.

1. Establecer criterios de aceptación y umbrales de calidad.

1) Criterios de aceptación funcional (nivel de sistema)

- **Acceso y disponibilidad**
  - El sistema web debe estar accesible 99.9% del tiempo.
  - Pautas de mantenimiento: ventanas planificadas fuera de horario comercial reducidas y notificadas con anticipación.
- **Rendimiento end-to-end**
  - Carga de la página de inicio en ≤ 3 segundos en conexión promedio (p 95 en producción).
  - Tiempos de respuesta de API (back-end) dentro de los límites establecidos (ver umbrales de calidad). **(SE REQUIERE VERIFICAR CON EL EQUIPO DE DESARROLLO).**

    - El rendering de menús en la página clave debe completar en ≤ 2 segundos en dispositivos de gama media. **(SE REQUIERE VERIFICAR CON EL EQUIPO DE DESARROLLO).**
- **Funcionalidad principal**
    - Visualización de menús: listado, filtros por categoría/acomodaciones, búsqueda, y ordenamiento deben funcionar sin errores.
    - Detalle de menú: página de detalle con todos los campos obligatorios visibles y con enlaces a operaciones relacionadas.
- **Gestión de contenido**
    - Crear/editar/eliminar menús y categorías desde la interfaz administrativa con validación de campos y permisos correspondientes.
    - Actualización en tiempo real de disponibilidad y precios cuando corresponda.
    - Importación/exportación de datos de menús en formatos estándar QR.
- **Autenticación y autorización**
    - Inicio de sesión funcional para usuarios y roles (admin, staff, usuario).
    - Roles correctamente aplicados: acciones sensibles restringidas según permisos.
    - Revocación de sesiones y manejo de tokens (renovación/expiración).
- **Accesibilidad y usabilidad**
    - Soporte básico de accesibilidad (contraste, navegación por teclado, textos alternativos de imágenes).
    - Rendimiento razonable en dispositivos móviles (responsivo) y navegadores modernos.
- **Seguridad**
    - Protección contra inyección básica, validación de entradas, y manejo seguro de tokens.
    - Cifrado de datos sensibles en tránsito (TLS) y buenas prácticas de seguridad.
- **Observabilidad y mantenimiento**
    - Logs estructurados y trazabilidad por operación (ID de pedido/consulta).
    - Monitoreo de uptime, latencias y errores con alertas básicas (Slack/Email).
    - Pruebas de regresión automatizadas ejecutadas en cada despliegue.

2) Umbrales de calidad (target) REVISAR ESTE PUNTO CON EL EQUIPO DE DESARROLLO.

- **Latencia y rendimiento**
    - UI inicial: p95 ≤ 2.5 segundos; p99 ≤ 4 segundos en staging; en producción, p95 ≤ 3 segundos.
    - API crítica (GET /menus, GET /menus/{id}): p95 ≤ 500 ms en staging; ≤ 350 ms en producción.
    - Operaciones mutation (POST/PUT/DELETE): p95 ≤ 800 ms.
    - Tiempos de renderizado en movil: tiempo total de interacción inicial ≤ 4 segundos.
- **Errores y confiabilidad**
    - Tasa de error global (5xx) < 0.5% en producción; 4xx < 1% para usos válidos.
    - Latencia de p99 durante picos no exceder 2x de p95.
- **Integridad de datos**
    - 100% de operaciones de creación/actualización con validación de esquema y reglas de negocio.

    - Consistencia entre UI y back-end: datos mostrados reflejan estado real (disponibilidad, precios, etc.).
    - Sin datos sensibles expuestos en respuestas.
- **Seguridad y cumplimiento**
    - Pruebas de autenticación/ autorización cubiertas; fallos de autorización devuelven 401/403 correctamente.
    - Escaneo de vulnerabilidades sin fallos críticos en pipelines de CI/CD.
    - Cumplimiento básico de privacidad y retención de datos según políticas.
- **Calidad de contrato y pruebas**
    - Contratos API (OpenAPI/GraphQL) cubiertos al 100% con pruebas automatizadas.
    - Pruebas de regresión automatizadas ejecutadas en cada despliegue; tasa de fallo inferior al 1%.
- Experiencia de desarrollador/operaciones

Preparación del entorno y datos.

1. Asegurar un entorno aislado para pruebas (sandbox/staging) con datos consistentes.
   https://cartas.qa.ticepro.com/
2. Crear conjuntos de datos representativos: positivos, negativos, límites y casos de error (403, 404, 429, 500).

Selección y mantenimiento de casos de prueba REVISAR ESTE PUNTO CON EL EQUIPO DE DESARROLLO.

1. Casos de regresión para operaciones CRUD básicas, autenticación, autorización, validación de entradas y manejo de errores.
2. Pruebas de integración entre servicios (por ejemplo, API A llama a API B).
3. Pruebas de rendimiento y límites (p. ej., picos de tráfico simulados).

Ejecución de pruebas REVISAR ESTE PUNTO CON EL EQUIPO DE DESARROLLO.

1. Ejecutar en el orden definido, priorizando endpoints críticos.
2. Verificar código de estado correcto, estructura del cuerpo, cabeceras y tiempos de respuesta.
3. Registrar resultados y evidencia (logs, respuestas completas, trazas).
4. Documentar incidencias con pasos reproducibles y configuraciones de entorno.

Gestión de incidencias y seguimiento

1. Crear tickets de defecto con severidad, entorno, reproducibilidad y impacto.
2. Asignar responsables y plazos; agrupar fallos por componente.
3. Re-ejecutar pruebas críticas después de correcciones y cambios de API.
4. Mantener trazabilidad entre requerimientos, casos de prueba y fallos.

Validación de cambios y cierre

1. Verificar que las correcciones no impacten otras rutas (regresión cruzada).
2. Confirmar que se cumplen criterios de aceptación (latencia, código de respuesta, formato).
3. Generar reporte de regresión específico de API (resumen, cobertura por endpoint, métricas).
4. Aprobar cierre del ciclo de regresión.

Métricas y mejora continua

1. Cobertura de regresión de API (endpoint críticos y contratos).
2. Tasa de fallo por servicio, tiempo de ciclo de regresión, tasa de re-trabajo.
3. Lecciones aprendidas y mejoras en contratos, validación de esquemas y pruebas de rendimiento.

Revision #5
Created 2025-09-04 20:27:17 UTC by Victoria Georgieff
Updated 2026-01-09 20:05:00 UTC by Victoria Georgieff