**Plan de Pruebas para Ticketazo**

Este documento describe la estrategia general de pruebas para el sistema **Ticketazo**, una plataforma web para la compra y gestión de entradas a espectáculos. El propósito del plan es asegurar que todas las funcionalidades críticas del sitio se verifiquen de manera sistemática, minimizando riesgos y garantizando la calidad del producto.

**1. Introducción**

Ticketazo es una aplicación web que permite a usuarios finales buscar eventos, filtrarlos por distintas categorías, ver detalles y adquirir entradas. Asimismo, provee funcionalidades específicas para organizadores, quienes pueden cargar eventos, gestionar salas y revisar estadísticas de ventas. Un riguroso plan de pruebas es indispensable para validar el funcionamiento de ambas caras del sistema (público y organizadores), detectar defectos a tiempo y ofrecer una experiencia de usuario consistente.

**2. Alcance**

El plan cubre los siguientes aspectos del sistema:

* Registro e inicio de sesión de usuarios y organizadores.
* Navegación y filtros en el listado de eventos para usuarios no autenticados y autenticados.
* Proceso de selección de asientos y generación de entradas (incluyendo entradas gratuitas).
* Gestión de entradas por parte del usuario (visualización de QR, listado de entradas, cierre de sesión).
* Funciones de organizador: registro, carga de eventos, gestión de salas y revisión de estadísticas.
* Gestión de permisos y roles (usuario estándar vs. organizador con y sin permisos completos).

Se excluyen por el momento las pruebas de pasarela de pago real y la integración con servicios externos que requieran acceso a cuentas bancarias o pasarelas reales. Dichas pruebas se planificarán en fases posteriores o en entornos de staging específicos.

**3. Estrategia de Pruebas**

El enfoque de pruebas combinará métodos manuales y automatizados:

1. **Pruebas Exploratorias**: se ejecutarán durante las primeras iteraciones para identificar comportamientos inesperados y construir el backlog inicial de casos de prueba.
2. **Pruebas Funcionales**: validar cada requisito mediante casos de prueba detallados, cubriendo escenarios positivos y negativos.
3. **Pruebas de Regresión**: se automatizarán con Cypress para asegurar que la incorporación de nuevas funcionalidades no rompa lo existente. Se ejecutarán en cada despliegue.
4. **Pruebas de Smoke**: conjunto reducido de pruebas críticas (acceso al sitio, registro, login, compra) que se ejecutan tras cada implementación para determinar si el sistema está en condiciones de continuar con pruebas más profundas.
5. **Pruebas de Seguridad y Roles**: se verificarán restricciones según el rol (usuario no autenticado, usuario autenticado estándar, organizador sin permisos, organizador con permisos completos).

**4. Herramientas y Recursos**

* **Cypress**: automatización de pruebas end‑to‑end para flujos clave (registro, login, filtros, compra de entradas, carga de eventos, etc.).
* **Trello**: registro y seguimiento de defectos, asignación de prioridades y comunicación entre QA y desarrollo.
* **GitHub**: repositorio donde se alojará el código de las pruebas automatizadas, con integración continua para ejecutar suites en cada merge.
* **Exploradores soportados**: pruebas manuales en Google Chrome (escritorio y móvil), y, de ser posible, en otros navegadores para validar compatibilidad (Firefox y Edge).

**5. Organización y Roles**

* **Equipo de QA**: responsable de definir casos de prueba, ejecutarlos manualmente, automatizarlos y documentar defectos.
* **Desarrolladores**: responsables de corregir defectos reportados y proveer información sobre funcionalidades nuevas y limitaciones técnicas.
* **Product Owner**: prioriza funcionalidades a probar, define criterios de aceptación y revisa el avance del plan de pruebas.

**6. Plan de Ejecución**

1. **Diseño de casos de prueba**: basados en historias de usuario y documentos de requisitos. Se dividirán por rol y criticidad.
2. **Revisión de casos**: reunión con el equipo de desarrollo para validar que los casos cubren los flujos relevantes.
3. **Ejecución manual inicial**: se abordarán escenarios críticos y se validarán los campos obligatorios, flujos alternativos y mensajes de error.
4. **Automatización**: se priorizarán los casos de mayor criticidad y repetitividad (registro, login, filtros, compra, carga de eventos) para automatizarlos con Cypress.
5. **Ciclos de regresión**: después de cada release se ejecutarán las suites automatizadas y una ronda breve de smoke tests manuales.
6. **Reporte de defectos**: cada defecto encontrado se registrará en Trello con severidad, pasos para reproducir, evidencia y entorno. El desarrollador deberá responder con la causa raíz y la solución aplicada.

**7. Criterios de Aceptación y Salida**

* **Criterio de entrada**: el entorno debe estar configurado (servidor activo, base de datos poblada con datos de prueba) y sin errores críticos conocidos.
* **Criterios de aceptación**:
  + Todos los casos de prueba planificados deben haberse ejecutado.
  + No deben existir defectos bloqueantes ni críticos abiertos al momento de pasar a producción.
  + Las pruebas automatizadas de mayor criticidad deben aprobarse al 100 %.
* **Criterio de salida**: se aprueba el despliegue cuando los criterios de aceptación se cumplen y el Product Owner da su visto bueno.

**8. Casos de Prueba y Criticidad**

Los casos de prueba se dividen por rol y se clasifican por la severidad de la funcionalidad que validan:

**8.1 Usuario no autenticado (visitante)**

* **Alta**
  + Buscar eventos y ver la lista de resultados.
  + Aplicar filtros (fecha, categoría, provincia, localidad) y validar que la grilla se actualiza.
  + Acceder al detalle de un evento y verificar que los datos se muestran correctamente.
  + Intentar comprar entradas y comprobar redirección al login.
* **Media**
  + Navegar paginado/infinite scroll y comprobar que se cargan eventos adicionales.
  + Verificar mensajes de error al buscar eventos inexistentes.

**8.2 Usuario autenticado estándar (sin rol de organizador)**

* **Alta**
  + Registro e inicio de sesión exitoso (incluye confirmación de email).
  + Validaciones de campos obligatorios en formularios de registro y login.
  + Recuperación de contraseña (flujo de “Olvidaste tu contraseña”).
  + Visualizar eventos, seleccionar asientos y generar entradas gratuitas.
  + Acceder a “Mis entradas” y visualizar detalles de cada ticket (incluyendo QR).
* **Media**
  + Modificar contraseña desde el perfil.
  + Cerrar sesión y asegurar que las opciones de usuario desaparecen.

**8.3 Organizador sin permisos completos**

* **Alta**
  + Registro de organizador exitoso y validaciones de campos obligatorios.
  + Mensaje de “Usuario no confirmado” hasta que un administrador active la cuenta.
  + Acceso a “Gestionar Salas” y visualización del aviso “No tienes permisos para crear un layout”.
  + Intentar cargar un evento y constatar que no puede avanzar al paso de selección de sectores.
* **Media**
  + Navegar por las secciones de organizador (Mis Eventos, Estadísticas) y documentar qué se muestra cuando no hay permisos.

**8.4 Organizador con permisos completos**

* **Alta**
  + Crear una sala desde “Gestionar Salas”, definiendo sectores y capacidad.
  + Cargar un evento con fecha, horario, género, ubicación y sectorización, y publicarlo.
  + Verificar que el evento aparece en “Mis Eventos” y que los usuarios pueden adquirir entradas.
  + Revisar estadísticas básicas de ventas.
* **Media**
  + Editar eventos existentes (cambiar horario o descripción) y validar que los cambios se reflejan.
  + Gestionar salas (editar o eliminar layouts) y comprobar las restricciones cuando hay eventos asociados.

**9. Plan de Contingencia**

En caso de fallos críticos durante la ejecución de pruebas:

1. Se registra el incidente en Trello con máxima prioridad, adjuntando evidencias y pasos para reproducirlo.
2. Se notifica de inmediato al equipo de desarrollo y se suspende la ejecución de casos relacionados.
3. Una vez corregido, se reejecuta la prueba afectada y luego el resto de la suite para asegurarse de que no haya regresiones.

**10. Conclusión**

Con el presente plan se pretende asegurar que **Ticketazo** cumpla con los requisitos funcionales y ofrezca una experiencia de usuario sólida tanto para compradores como para organizadores. La combinación de pruebas manuales exhaustivas y automatización con Cypress permitirá detectar defectos tempranamente y garantizar que futuras mejoras no introduzcan regresiones.