# Plan de Pruebas de Aceptación

## Introducción

Este documento define el plan de pruebas de aceptación del Sistema de ticketing interno, basado en los requerimientos funcionales representados mediante historias de usuario. El objetivo es validar todas las funcionalidades que el sistema requiere para satisfacer las necesidades del usuario final. La herramienta principal para validar la API será Postman; podrán considerarse otras alternativas si es necesario.

## Alcance del Plan de Pruebas

Se validarán las funcionalidades, por medio de los endpoints de la API para cada uno de los roles del sistema, asegurando de esta manera un funcionamiento completo y apropiado de lo que es requerido por el cliente.

Los roles son:

* Empleado
* Agente
* Admin

## Ambientes de prueba

Estas pruebas serán realizadas en un entorno principal, durante el desarrollo se podrán probar las diferentes funcionalidades y se registrarán los resultados.

## Criterios de Aceptación

Los criterios de aceptación están basados en las historias de usuario, de esta forma se define lo que realmente necesita el usuario principal, según cada rol en la aplicación.

En cada criterio se definió un código de estado y una salida esperada.

## Estrategia de pruebas

Se utilizarán tres estrategias de pruebas por caso de prueba con los siguientes enfoques:

1. **Pruebas de solicitud correcta**: Se espera que operaciones exitosas (como el login) retornen un código de estado HTTP que indique éxito (por ejemplo: 200 OK).
2. **Pruebas de error**: Se considerarán casos alternativos, donde se espera que la API tenga un manejo adecuado de los errores.
3. **Pruebas de autenticación**: Se pondrá a prueba todo lo relacionado con el de acceso a los recursos de la API

## Casos de Prueba

* CP-01
* **Caso de Prueba**: Iniciar sesión correctamente
* **Endpoint**: localhost:8080/api/auth/login
* **Método HTTP**: POST
* **Descripción**: Autentica un usuario usando email y contraseña y retorna un token valido.
* **Parámetros**:

{  
 "email": "string",  
 "password": "string"  
}

* **Headers**:
  + **Content-Type**: application/json
* **Código de Respuesta Esperado**: 200 OK
* **Prueba Exitosa si**: La API devuelve un código 200 OK con un token valido

{  
 "token": "<jwt-token>"  
}

* CP-02
* **Caso de Prueba**: Inicio de sesión fallido con credenciales incorrectas
* **Endpoint**: localhost:8080/api/auth/login
* **Método HTTP**: POST
* **Descripción**: Verifica que la API rechace el inicio de sesión cuando se ingresan credenciales incorrectas, retornando un código 401 Unauthorized con un mensaje de error adecuado.
* **Parámetros**:

{  
 "email": "string",  
 "password": "string"  
}

* **Headers**:
  + **Content-Type**: application/json
* **Código de Respuesta Esperado**: 401 Unauthorized
* **Prueba Exitosa si**: La API devuelve un código 401 Unauthorized con un mensaje adecuado
* CP-03
* **Caso de Prueba**: Petición fallida cuando se accede a un endpoint con token invalido
* **Endpoint**: Todo endpoint excepto /auth/login
* **Método HTTP**: Segun el endpoint
* **Descripción**: Valida que se impida el acceso a un endpoint sin autenticación correcta
* **Parámetros**: Depende del endpoint
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 403 Forbidden
* **Prueba Exitosa si**: La API devuelve un código 403 Forbidden con un mensaje adecuado
* CP-04
* **Caso de Prueba**: Crear un ticket
* **Endpoint**: localhost:8080/api/tickets
* **Método HTTP**: POST
* **Descripción**: Este endpoint permite a un usuario autenticado crear un nuevo ticket en el sistema, proporcionando los datos requeridos como categoría, título, descripción y prioridad.
* **Parámetros**:

{  
 "usuario\_id": 123,  
 "categoria\_id": 2,  
 "titulo": "Error al iniciar sesión",  
 "descripcion": "No puedo ingresar al sistema con mis credenciales.",  
 "prioridad": "urgente"  
}

* **Headers**: Content-Type: application/json Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 201 Created
* **Prueba Exitosa si**:
  + El sistema devuelve un código de estado HTTP 201 Created.
  + La respuesta contiene un objeto JSON con los datos del ticket creado, incluyendo su ID, fecha\_creacion, y un estado inicial.

{  
 "id": 456,  
 "usuario\_id": 123,  
 "categoria\_id": 2,  
 "titulo": "Error al iniciar sesión",  
 "descripcion": "No puedo ingresar al sistema con mis credenciales.",  
 "prioridad": "urgente",  
 "estado": "Pendiente",  
 "fecha\_creacion": "2025-09-28T10:15:00Z"  
}

* CP-05
* **Caso de Prueba**: Crear un ticket con datos inválidos
* **Endpoint**: localhost:8080/api/tickets
* **Método HTTP**: POST
* **Descripción**: Este test valida que el sistema rechace correctamente la creación de un ticket cuando los datos enviados no cumplen con las reglas de validación (campos faltantes, vacíos o incorrectos).
* **Parámetros**:

{  
 "usuario\_id": 123,  
 "categoria\_id": 2,  
 "titulo": "Error al iniciar sesión",  
 "descripcion": "No puedo ingresar al sistema con mis credenciales.",  
 "prioridad": "urgente"  
}

* **Headers**: Content-Type: application/json Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 400 Bad Request
* **Prueba Exitosa si**:
  + El sistema rechaza la solicitud con un código de estado HTTP 400 Bad Request.
  + La respuesta contiene un mensaje de error indicando claramente qué campos son inválidos o faltan.
* CP-06
* **Caso de Prueba**: Crear un ticket con recurso no encontrado
* **Endpoint**: localhost:8080/api/tickets
* **Método HTTP**: POST
* **Descripción**:Este caso de prueba verifica que el sistema retorne un error 404 Not Found cuando se intenta crear un ticket con referencias a recursos inexistentes, como un usuario\_id o categoria\_id que no existen en la base de datos.
* **Parámetros**:

{  
 "usuario\_id": 123,  
 "categoria\_id": 2,  
 "titulo": "Error al iniciar sesión",  
 "descripcion": "No puedo ingresar al sistema con mis credenciales.",  
 "prioridad": "urgente"  
}

* **Headers**: Content-Type: application/json Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 404 Not Found
* **Prueba Exitosa si**:
  + El sistema retorna el código de estado HTTP 404 Not Found.
  + El cuerpo de la respuesta indica claramente cuál recurso no fue encontrado.
* CP-07
* **Caso de Prueba**: Error del servidor al crear un ticket
* **Endpoint**: localhost:8080/api/tickets
* **Método HTTP**: POST
* **Descripción**: Verificar que, ante un fallo interno al crear un ticket, el sistema devuelva 500 Internal Server Error con un mensaje genérico sin exponer detalles técnicos.
* **Parámetros**:

{  
 "usuario\_id": 123,  
 "categoria\_id": 2,  
 "titulo": "Error al iniciar sesión",  
 "descripcion": "No puedo ingresar al sistema con mis credenciales.",  
 "prioridad": "urgente"  
}

* **Headers**: Content-Type: application/json Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 500 Internal Server Error
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 500 Internal Server Error.
  + La respuesta incluye un mensaje de error genérico sin exponer detalles.
* CP-08
* **Caso de Prueba**: Ver tabla de tickets creados
* **Endpoint**: localhost:8080/api/tickets
* **Método HTTP**: GET
* **Descripción**: Este caso de prueba valida que el sistema devuelva correctamente la lista de tickets creados por los empleados
* **Parámetros**: ?estado=enProgreso&prioridad=urgente&usuario\_id=123
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 200 OK
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 200 OK.
  + La respuesta contiene un arreglo JSON con todos los tickets disponibles para el empleado autenticado

[  
 {  
 "id": 101,  
 "usuario\_id": 123,  
 "categoria\_id": 4,  
 "titulo": "Falla en sistema de correo",  
 "descripcion": "No puedo enviar correos desde ayer.",  
 "prioridad": "importante",  
 "estado": "enProgreso",  
 "fecha\_creacion": "2025-09-25T09:00:00Z",  
 "fecha\_cierre": null  
 },  
 {  
 "id": 102,  
 "usuario\_id": 123,  
 "categoria\_id": 1,  
 "titulo": "Solicitud de acceso VPN",  
 "descripcion": "Necesito acceso remoto para trabajar desde casa.",  
 "prioridad": "urgente",  
 "estado": "Cerrado",  
 "fecha\_creacion": "2025-09-20T14:30:00Z",  
 "fecha\_cierre": "2025-09-21T10:15:00Z"  
 }  
]

* CP-09
* **Caso de Prueba**: Consultar tickets con parámetros inválidos
* **Endpoint**: localhost:8080/api/tickets
* **Método HTTP**: GET
* **Descripción**: Este caso de prueba valida que el sistema maneje correctamente los errores cuando se envían parámetros de consulta (query params) con valores no permitidos o en formato incorrecto, como una prioridad inexistente o un estado no válido.
* **Parámetros**: ?estado=enProgreso&prioridad=urgente&usuario\_id=123
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 400 Bad Request
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 400 Bad Request.
  + Se rechaza la solicitud sin ejecutar la consulta.
  + La respuesta contiene un mensaje de error que describe los parámetros inválidos.
  + No se devuelve ninguna lista de tickets.
* CP-10
* **Caso de Prueba**: No se encontraron tickets ---->
* **Endpoint**: localhost:8080/api/tickets
* **Método HTTP**: GET
* **Descripción**: Este caso de prueba valida que el sistema responda adecuadamente cuando se realiza una consulta con filtros válidos pero que no coinciden con ningún ticket registrado en la base de datos.
* **Parámetros**: ?estado=enProgreso&prioridad=urgente&usuario\_id=123
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 404 Not Found
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 404 Not Found.
  + La respuesta indica claramente que no se encontraron tickets con los filtros proporcionados.
* CP-11
* **Caso de Prueba**: Error del servidor al consultar tickets
* **Endpoint**: localhost:8080/api/tickets
* **Método HTTP**: GET
* **Descripción**: Este caso de prueba valida que el sistema maneje correctamente los errores internos inesperados.
* **Parámetros**: ?estado=enProgreso&prioridad=urgente&usuario\_id=123
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 500 Internal Server Error
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 500 Internal Server Error.
  + La respuesta incluye un mensaje de error genérico sin exponer detalles.
* CP-12
* **Caso de Prueba**: Ver detalles de un ticket
* **Endpoint**: localhost:8080/api/tickets/{idTicket}
* **Método HTTP**: GET
* **Descripción**: Este caso de prueba valida que el sistema devuelva correctamente la información completa de un ticket específico al consultarlo por su ID, siempre que el ticket exista y el usuario tenga permisos para verlo.
* **Parámetros**: No aplica
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 200 OK
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 200 OK.
  + La respuesta contiene un objeto JSON con todos los detalles del ticket

{  
 "id": 456,  
 "usuario\_id": 123,  
 "categoria\_id": 2,  
 "titulo": "Falla en impresora",  
 "descripcion": "La impresora no responde a los comandos de impresión.",  
 "prioridad": "urgente",  
 "estado": "En Progreso",  
 "fecha\_creacion": "2025-09-25T09:00:00Z",  
 "fecha\_cierre": null  
}

* CP-13
* **Caso de Prueba**: Solicitud inválida al consultar detalles del ticket
* **Endpoint**: localhost:8080/api/tickets/{idTicket}
* **Método HTTP**: GET
* **Descripción**: Este caso de prueba valida que el sistema maneje correctamente los errores cuando el parámetro idTicket es inválido (por ejemplo, no es un número, es nulo o tiene un formato incorrecto).
* **Parámetros**: No aplica
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 400 Bad Request
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 400 Bad Request.
  + La respuesta contiene un mensaje de error claro indicando que el ID del ticket no es válido.
* CP-14
* **Caso de Prueba**: Ticket no encontrado
* **Endpoint**: localhost:8080/api/tickets/{idTicket}
* **Método HTTP**: GET
* **Descripción**: Este caso de prueba valida que el sistema responda adecuadamente cuando se solicita el detalle de un ticket que no existe en la base de datos.
* **Parámetros**: No aplica
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 404 Not Found
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 404 Not Found.
  + La respuesta contiene un mensaje claro que indica que el ticket no fue encontrado.
  + No se devuelve ningún objeto de ticket.
* CP-15
* **Caso de Prueba**: Error del servidor al obtener detalles del ticket
* **Endpoint**: localhost:8080/api/tickets/{idTicket}
* **Método HTTP**: GET
* **Descripción**: Este caso de prueba valida que el sistema maneje correctamente los errores internos inesperados al intentar obtener el detalle de un ticket.
* **Parámetros**: No aplica
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 500 Internal Server Error
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 500 Internal Server Error.
  + La respuesta incluye un mensaje de error genérico sin exponer detalles.
* CP-16
* **Caso de Prueba**: Ver proceso del ticket (comentarios)
* **Endpoint**: localhost:8080/api/tickets/{idTicket}/comentarios
* **Método HTTP**: GET
* **Descripción**: Este caso de prueba valida que el sistema devuelva correctamente la lista de comentarios asociados a un ticket específico, mostrando el historial del proceso desde su creación hasta la resolución.
* **Parámetros**: No aplica
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 200 OK
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 200 OK.
  + La respuesta contiene un arreglo JSON con todos los comentarios registrados para el ticket.

[  
 {  
 "id": 301,  
 "ticket\_id": 456,  
 "usuario\_id": 789,  
 "comentario": "Estamos revisando el problema.",  
 "fecha\_creacion": "2025-09-25T10:30:00Z"  
 },  
 {  
 "id": 302,  
 "ticket\_id": 456,  
 "usuario\_id": 123,  
 "comentario": "Gracias por la atención, sigo esperando.",  
 "fecha\_creacion": "2025-09-26T08:15:00Z"  
 }  
]

* CP-17
* **Caso de Prueba**: Parámetros inválidos al consultar proceso del ticket
* **Endpoint**: localhost:8080/api/tickets/{idTicket}/comentarios
* **Método HTTP**: GET
* **Descripción**: Este caso de prueba valida que el sistema maneje correctamente los errores cuando el parámetro idTicket es inválido
* **Parámetros**: No aplica
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 400 Bad Request
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 400 Bad Request.
  + La respuesta contiene un mensaje de error claro indicando que el parámetro idTicket no es válido.
  + No se devuelve ninguna lista de comentarios.
* CP-18
* **Caso de Prueba**: Proceso del ticket no encontrado ---->
* **Endpoint**: localhost:8080/api/tickets/{idTicket}/comentarios
* **Método HTTP**: GET
* **Descripción**:Este caso de prueba valida que el sistema responda adecuadamente cuando se consultan los comentarios de un ticket que no existe en la base de datos o cuando el ticket existe pero no tiene comentarios registrados.
* **Parámetros**: No aplica
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 404 Not Found
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 404 Not Found.
  + La respuesta contiene un mensaje claro que indique que no se encontraron comentarios asociados al ticket solicitado.
  + No se devuelve ningún arreglo de comentarios vacío ni datos irrelevantes.
* CP-19
* **Caso de Prueba**: Error del servidor al consultar proceso del ticket
* **Endpoint**: localhost:8080/api/tickets/{idTicket}/comentarios
* **Método HTTP**: GET
* **Descripción**: Este caso de prueba valida que el sistema maneje correctamente errores internos inesperados al intentar obtener los comentarios asociados a un ticket, como fallas en la base de datos, excepciones no controladas o problemas de conexión.
* **Parámetros**: No aplica
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 500 Internal Server Error
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 500 Internal Server Error.
  + La respuesta incluye un mensaje de error genérico sin exponer detalles.
* CP-20
* **Caso de Prueba**: Ver notificaciones del sistema
* **Endpoint**: localhost:8080/api/usuarios/{idUsuario}/notificaciones
* **Método HTTP**: GET
* **Descripción**: Este caso de prueba valida que un usuario autenticado pueda consultar la lista de notificaciones generadas por el sistema, relacionadas con sus tickets, asignaciones o actualizaciones.
* **Parámetros**: No aplica
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 200 OK
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 200 OK.
  + La respuesta incluye un arreglo con las notificaciones asociadas al usuario.
  + Si el usuario no tiene notificaciones, se retorna una lista vacía sin error.
* CP-21
* **Caso de Prueba**: Solicitud inválida al consultar notificaciones
* **Endpoint**: localhost:8080/api/usuarios/{idUsuario}/notificaciones
* **Método HTTP**: GET
* **Descripción**: Verificar que el sistema responda con error cuando se realiza una solicitud con parámetros inválidos o mal formados al consultar las notificaciones de un usuario.
* **Parámetros**: No aplica
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 400 Bad Request
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 400 Bad Request.
  + La respuesta incluye un mensaje de error claro indicando que la solicitud es inválida.
  + No se retorna información de notificaciones.
* CP-22
* **Caso de Prueba**: Usuario no encontrado
* **Endpoint**: localhost:8080/api/usuarios/{idUsuario}/notificaciones
* **Método HTTP**: GET
* **Descripción**: Validar que el sistema responda adecuadamente cuando no exista el usuario solicitado.
* **Parámetros**: No aplica
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 404 Not Found
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 404 Not Found.
  + La respuesta contiene un mensaje claro indicando que no se encontró el usuario con el idUsuario especificado.
* CP-23
* **Caso de Prueba**: Error del servidor al obtener notificaciones
* **Endpoint**: localhost:8080/api/usuarios/{idUsuario}/notificaciones
* **Método HTTP**: GET
* **Descripción**: Validar que el sistema responda de forma controlada cuando ocurre un error interno inesperado al intentar recuperar las notificaciones de un usuario.
* **Parámetros**:
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 500 Internal Server Error
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 500 Internal Server Error.
  + La respuesta incluye un mensaje de error genérico sin exponer detalles.
* CP-24
* **Caso de Prueba**: Modificar contraseña y datos personales
* **Endpoint**: localhost:8080/api/usuarios/{idUsuario}
* **Método HTTP**: PATCH
* **Descripción**: Verificar que el usuario pueda actualizar correctamente sus datos personales como email personal y su contraseña mediante una solicitud parcial de actualización.
* **Parámetros**:

{  
 "email\_personal": "nuevo.correo@example.com",  
 "contrasena": "NuevaContraseña123"  
}

* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 200 OK
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 200 OK.
  + La respuesta incluye el objeto usuario actualizado con los nuevos valores (sin mostrar la contraseña en texto plano).
* CP-25
* **Caso de Prueba**: Error 400 - Datos inválidos al modificar contraseña o perfil
* **Endpoint**: localhost:8080/api/usuarios/{idUsuario}
* **Método HTTP**: PATCH
* **Descripción**: Validar que el sistema rechace la actualización cuando los datos enviados en la solicitud son inválidos o no cumplen con las validaciones establecidas (ejemplo: email con formato incorrecto, contraseña demasiado corta, campos vacíos).
* **Parámetros**:

{  
 "email\_personal": "nuevo.correo@example.com",  
 "contrasena": "NuevaContraseña123"  
}

* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 400 Bad Request
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 400 Bad Request.
  + La respuesta incluye un mensaje claro indicando qué campos no cumplen con las validaciones (ejemplo: “El correo electrónico no es válido” o “La contraseña debe tener mínimo 8 caracteres”).
* CP-26
* **Caso de Prueba**: Error 404 - Usuario no encontrado al modificar datos
* **Endpoint**: localhost:8080/api/usuarios/{idUsuario}
* **Método HTTP**: PATCH
* **Descripción**: Verificar que el sistema devuelva un error cuando se intente actualizar los datos de un usuario que no existe en el sistema.
* **Parámetros**:

{  
 "email\_personal": "nuevo.correo@example.com",  
 "contrasena": "NuevaContraseña123"  
}

* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 404 Not Found
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 404 Not Found.
  + La respuesta incluye un mensaje indicando que el usuario no existe o no fue encontrado.
* CP-27
* **Caso de Prueba**: Error 500 - Error del servidor al actualizar información del usuario
* **Endpoint**: localhost:8080/api/usuarios/{idUsuario}
* **Método HTTP**: PATCH
* **Descripción**: Validar que el sistema maneje correctamente un fallo interno inesperado al intentar actualizar los datos de un usuario.
* **Parámetros**:

{  
 "nombre": "Nuevo Nombre",  
 "email\_personal": "nuevo.correo@example.com",  
 "contrasena": "NuevaContraseña123"  
}

* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 500 Internal Server Error
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 500 Internal Server Error.
  + La respuesta incluye un mensaje de error genérico sin exponer detalles.
* CP-28
* **Caso de Prueba**: Ver tickets asignados al agente
* **Endpoint**: localhost:8080/api/agentes/{idAgente}/tickets?estado=enProgreso&estado=pendiente
* **Método HTTP**: GET
* **Descripción**: Validar que el agente pueda consultar la lista de tickets que le han sido asignados, filtrando por estados en progreso y pendiente.
* **Parámetros**: No aplica
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 200 OK
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 200 OK.
  + La respuesta incluye una lista de tickets asignados al agente.
  + Solo se muestran los tickets con estado enProgreso o pendiente.
* CP-29
* **Caso de Prueba**: Solicitud inválida al consultar tickets asignados
* **Endpoint**: localhost:8080/api/agentes/{idAgente}/tickets?estado=enProgreso&estado=pendiente
* **Método HTTP**: GET
* **Descripción**:Validar que el sistema devuelva un error 400 Bad Request cuando se envían parámetros inválidos o mal formados al consultar los tickets de un agente.
* **Parámetros**: No aplica
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 400 Bad Request
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 400 Bad Request.
  + El mensaje de error indica que los parámetros enviados son inválidos.
  + No se retorna ninguna lista de tickets.
* CP-30
* **Caso de Prueba**: No se encontraron tickets asignados -->
* **Endpoint**: localhost:8080/api/agentes/{idAgente}/tickets?estado=enProgreso&estado=pendiente
* **Método HTTP**: GET
* **Descripción**:Validar que el sistema responda correctamente cuando el agente no tiene tickets asignados en los estados solicitados.
* **Parámetros**: No aplica
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 404 Not Found
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 404 Not Found.
  + El mensaje indica que no se encontraron tickets para el agente en los estados solicitados.
  + No se retorna ninguna lista de tickets.
* CP-31
* **Caso de Prueba**: Error del servidor al consultar tickets asignados
* **Endpoint**: localhost:8080/api/agentes/{idAgente}/tickets?estado=enProgreso&estado=pendiente
* **Método HTTP**: GET
* **Descripción**: Validar que el sistema maneje correctamente un error interno al intentar obtener los tickets asignados a un agente.
* **Parámetros**: No aplica
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 500 Internal Server Error
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 500 Internal Server Error.
  + La respuesta incluye un mensaje de error genérico sin exponer detalles.
* CP-32
* **Caso de Prueba**: Ver historial completo de tickets
* **Endpoint**: localhost:8080/api/agentes/{idAgente}/tickets?estado=cerrado
* **Método HTTP**: GET
* **Descripción**: Validar que el agente pueda consultar el historial completo de tickets que ya han sido cerrados.
* **Parámetros**: No aplica
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 200 OK
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 200 OK.
  + La respuesta incluye una lista de tickets cerrados asociados al agente.
* CP-33
* **Caso de Prueba**: Parámetros inválidos al consultar historial de tickets
* **Endpoint**: localhost:8080/api/agentes/{idAgente}/tickets?estado=cerrado
* **Método HTTP**: GET
* **Descripción**: Validar que el sistema devuelva error 400 Bad Request cuando los parámetros enviados para consultar el historial de tickets son inválidos o mal formados.
* **Parámetros**: No aplica
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 400 Bad Request
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 400 Bad Request.
  + El mensaje de error indica que los parámetros enviados son inválidos.
  + No se retorna ningún listado de tickets.
* CP-34
* **Caso de Prueba**: Historial de tickets no encontrado -->
* **Endpoint**: localhost:8080/api/agentes/{idAgente}/tickets?estado=cerrado
* **Método HTTP**: GET
* **Descripción**: Validar que el sistema responda correctamente cuando no existen tickets cerrados asociados al agente.
* **Parámetros**: No aplica
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 404 Not Found
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 404 Not Found.
  + El mensaje indica que no se encontraron tickets cerrados para el agente.
  + No se retorna ningún listado de tickets.
* CP-35
* **Caso de Prueba**: Error del servidor al obtener historial de tickets
* **Endpoint**: localhost:8080/api/agentes/{idAgente}/tickets?estado=cerrado
* **Método HTTP**: GET
* **Descripción**: Validar que el sistema gestione adecuadamente un error interno inesperado al intentar obtener el historial de tickets cerrados de un agente.
* **Parámetros**: No aplica
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 500 Internal Server Error
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 500 Internal Server Error.
  + La respuesta incluye un mensaje de error genérico sin exponer detalles.
* CP-36
* **Caso de Prueba**: Crear un comentario en un ticket
* **Endpoint**: localhost:8080/api/tickets/{ticketId}/comments
* **Método HTTP**: POST
* **Descripción**: Validar que un agente pueda agregar correctamente un comentario en un ticket asignado, garantizando que la información se guarde en el sistema.
* **Parámetros**:

{  
 "usuario\_id": 45,  
 "comentario": "Estamos revisando el problema y pronto tendrás una solución."  
}

* **Headers**: Content-Type: application/json Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 201 Created
* **Prueba Exitosa si**:
* CP-37
* **Caso de Prueba**: Datos inválidos al crear comentario
* **Endpoint**: localhost:8080/api/tickets/{ticketId}/comments
* **Método HTTP**: POST
* **Descripción**: Validar que el sistema rechace la creación de un comentario cuando los datos enviados no cumplen con el formato o campos requeridos.
* **Parámetros**:

{  
 "usuario\_id": 45,  
 "comentario": "Estamos revisando el problema y pronto tendrás una solución."  
}

* **Headers**: Content-Type: application/json Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 400 Bad Request
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 400 Bad Request.
  + La respuesta incluye un mensaje indicando que los datos son inválidos o faltan campos obligatorios.
* CP-38
* **Caso de Prueba**: Ticket no encontrado para comentar
* **Endpoint**: localhost:8080/api/tickets/{ticketId}/comments
* **Método HTTP**: POST
* **Descripción**: Validar que el sistema devuelva un error cuando se intente agregar un comentario a un ticket inexistente.
* **Parámetros**:

{  
 "usuario\_id": 45,  
 "comentario": "Estamos revisando el problema y pronto tendrás una solución."  
}

* **Headers**: Content-Type: application/json Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 404 Not Found
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 404 Not Found.
  + La respuesta incluye un mensaje indicando que el ticket no existe o no fue encontrado.
* CP-39
* **Caso de Prueba**: Error del servidor al crear comentario en el ticket
* **Endpoint**: localhost:8080/api/tickets/{ticketId}/comments
* **Método HTTP**: POST
* **Descripción**: Validar que el sistema responda adecuadamente con un error genérico cuando ocurra una falla interna al intentar crear un comentario en un ticket.
* **Parámetros**:

{  
 "usuario\_id": 45,  
 "comentario": "Estamos revisando el problema y pronto tendrás una solución."  
}

* **Headers**: Content-Type: application/json Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 500 Internal Server Error
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 500 Internal Server Error.
  + La respuesta incluye un mensaje de error genérico sin exponer detalles.
* CP-40
* **Caso de Prueba**: Actualizar prioridad de un ticket
* **Endpoint**: localhost:8080/api/tickets/{idTicket}/prioridad
* **Método HTTP**: PATCH
* **Descripción**: Validar que el sistema permita actualizar correctamente la prioridad de un ticket existente.
* **Parámetros**:

{  
 "prioridad": "importante"  
}

* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 200 OK
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 200 OK.
  + La respuesta incluye la información del ticket actualizado con la nueva prioridad.
* CP-41
* **Caso de Prueba**: Datos inválidos al actualizar prioridad del ticket
* **Endpoint**: localhost:8080/api/tickets/{idTicket}/prioridad
* **Método HTTP**: PATCH
* **Descripción**: Validar que el sistema rechace la actualización de la prioridad de un ticket cuando los datos enviados no son válidos.
* **Parámetros**:

{  
 "prioridad": "importante"  
}

* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 400 Bad Request
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 400 Bad Request.
  + La respuesta incluye un mensaje indicando que el valor de la prioridad no es válido.
* CP-42
* **Caso de Prueba**: Ticket no encontrado para actualizar prioridad
* **Endpoint**: localhost:8080/api/tickets/{idTicket}/prioridad
* **Método HTTP**: PATCH
* **Descripción**: Validar que el sistema devuelva un error cuando se intenta actualizar la prioridad de un ticket inexistente.
* **Parámetros**:

{  
 "prioridad": "importante"  
}

* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**:404 Not Found
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 404 Not Found.
  + La respuesta incluye un mensaje indicando que el ticket no existe.
* CP-43
* **Caso de Prueba**: Error del servidor al actualizar prioridad del ticket
* **Endpoint**: localhost:8080/api/tickets/{idTicket}/prioridad
* **Método HTTP**: PATCH
* **Descripción**: Validar que el sistema maneje adecuadamente un error interno al intentar actualizar la prioridad de un ticket.
* **Parámetros**:

{  
 "prioridad": "importante"  
}

* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 500 Internal Server Error
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 500 Internal Server Error.
  + La respuesta incluye un mensaje de error genérico sin exponer detalles.
* CP-44
* **Caso de Prueba**: Cerrar un ticket
* **Endpoint**: localhost:8080/api/tickets/{idTicket}/cerrar
* **Método HTTP**: PATCH
* **Descripción**: Validar que un ticket existente pueda ser cerrado correctamente, cambiando su estado a cerrado.
* **Parámetros**: No aplica.
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 200 OK
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 200 OK.
  + La respuesta incluye el ticket actualizado con estado cerrado.
* CP-45
* **Caso de Prueba**: Solicitud inválida al cerrar el ticket
* **Endpoint**: localhost:8080/api/tickets/{idTicket}/cerrar
* **Método HTTP**: PATCH
* **Descripción**: Validar que, si la solicitud enviada para cerrar un ticket contiene datos inválidos o mal formateados, el sistema la rechace.
* **Parámetros**: No aplica.
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 400 Bad Request
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 400 Bad Request.
  + La respuesta incluye un mensaje indicando que la solicitud es inválida.
* CP-46
* **Caso de Prueba**: Ticket no encontrado para cerrar
* **Endpoint**: localhost:8080/api/tickets/{idTicket}/cerrar
* **Método HTTP**: PATCH
* **Descripción**: Validar que, si se intenta cerrar un ticket inexistente o cuyo ID no está registrado en el sistema, la operación falle con el código adecuado.
* **Parámetros**: No aplica.
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**:404 Not Found
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 404 Not Found.
  + La respuesta incluye un mensaje que indique que el ticket no fue encontrado.
* CP-47
* **Caso de Prueba**: Error del servidor al cerrar el ticket
* **Endpoint**: localhost:8080/api/tickets/{idTicket}/cerrar
* **Método HTTP**: PATCH
* **Descripción**: Validar el comportamiento del sistema cuando ocurre un error interno al intentar cerrar un ticket existente.
* **Parámetros**: No aplica.
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 500 Internal Server Error
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 500 Internal Server Error.
  + La respuesta incluye un mensaje de error genérico sin exponer detalles.
* CP-48
* **Caso de Prueba**: Reasignar tickets a los agentes
* **Endpoint**: localhost:8080/api/tickets/{idTicket}
* **Método HTTP**: POST
* **Descripción**: Validar que el administrador pueda reasignar un ticket a un agente disponible.
* **Parámetros**:

{  
 "agenteId": 123  
}

* **Headers**: Content-Type: application/json Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 201 Created
* **Prueba Exitosa si**:
  + El ticket es reasignado correctamente al agente.
  + La respuesta incluye información del ticket con el nuevo agenteId asignado.
* CP-49
* **Caso de Prueba**: Datos inválidos al reasignar ticket a un agente
* **Endpoint**: localhost:8080/api/tickets/{idTicket}
* **Método HTTP**: POST
* **Descripción**: Validar que el sistema rechace la reasignación si los datos enviados no son válidos.
* **Parámetros**:

{  
 "agenteId": 123  
}

* **Headers**: Content-Type: application/json Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 400 Bad Request
* **Prueba Exitosa si**:
  + El sistema devuelve el código 400.
  + La respuesta incluye un mensaje indicando que los datos enviados son inválidos.
* CP-50
* **Caso de Prueba**: Ticket o agente no encontrado para reasignación
* **Endpoint**: localhost:8080/api/tickets/{idTicket}
* **Método HTTP**: POST
* **Descripción**: Validar que el sistema notifique cuando el ticket o el agente especificado no existan.
* **Parámetros**:

{  
 "agenteId": 123  
}

* **Headers**: Content-Type: application/json Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 404 Not Found
* **Prueba Exitosa si**:
  + El sistema responde con código 404.
  + El mensaje de error indica que no se encontró el ticket o el agente solicitado.
* CP-51
* **Caso de Prueba**: Error del servidor al reasignar ticket
* **Endpoint**: localhost:8080/api/tickets/{idTicket}
* **Método HTTP**: POST
* **Descripción**: Validar que el sistema maneje correctamente errores internos al intentar reasignar un ticket.
* **Parámetros**:

{  
 "agenteId": 123  
}

* **Headers**: Content-Type: application/json Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 500 Internal Server Error
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 500 Internal Server Error.
  + La respuesta incluye un mensaje de error genérico sin exponer detalles.
* CP-52
* **Caso de Prueba**: Crear categorías para los tickets
* **Endpoint**: localhost:8080/api/categorias
* **Método HTTP**: POST
* **Descripción**: Validar que el administrador pueda crear una nueva categoría para organizar los tickets.
* **Parámetros**:

{  
 "nombre": "Soporte Técnico",  
 "descripcion": "Problemas relacionados con hardware y software"  
}

* **Headers**: Content-Type: application/json Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 201 Created
* **Prueba Exitosa si**:
  + El sistema devuelve código 201.
  + La respuesta incluye la categoría creada con su id, nombre y descripcion.
* CP-53
* **Caso de Prueba**: Datos inválidos al crear categoría de ticket
* **Endpoint**: localhost:8080/api/categorias
* **Método HTTP**: POST
* **Descripción**: Validar que el sistema no permita crear una categoría con datos incompletos o inválidos.
* **Parámetros**:

{  
 "nombre": "Soporte Técnico",  
 "descripcion": "Problemas relacionados con hardware y software"  
}

* **Headers**: Content-Type: application/json Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 400 Bad Request
* **Prueba Exitosa si**:
  + El sistema devuelve código 400.
  + La respuesta incluye un mensaje indicando que los datos son inválidos o faltantes.
* CP-54
* **Caso de Prueba**: Recurso relacionado no encontrado al crear categoría
* **Endpoint**: localhost:8080/api/categorias
* **Método HTTP**: POST
* **Descripción**: Validar que el sistema notifique cuando se intente asociar la categoría a un recurso inexistente (ej: departamento).
* **Parámetros**:

{  
 "nombre": "Soporte Técnico",  
 "descripcion": "Problemas relacionados con hardware y software"  
}

* **Headers**: Content-Type: application/json Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 404 Not Found
* **Prueba Exitosa si**:
  + El sistema devuelve código 404.
  + La respuesta incluye un mensaje de error indicando que el recurso relacionado no existe.
* CP-55
* **Caso de Prueba**: Error del servidor al crear categoría de ticket
* **Endpoint**: localhost:8080/api/categorias
* **Método HTTP**: POST
* **Descripción**: Validar que el sistema maneje correctamente errores internos al crear una categoría.
* **Parámetros**:

{  
 "nombre": "Soporte Técnico",  
 "descripcion": "Problemas relacionados con hardware y software"  
}

* **Headers**: Content-Type: application/json Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 500 Internal Server Error
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 500 Internal Server Error.
  + La respuesta incluye un mensaje de error genérico sin exponer detalles.
* CP-56
* **Caso de Prueba**: Crear cuentas de empleados y agentes
* **Endpoint**: localhost:8080/api/usuarios
* **Método HTTP**: POST
* **Descripción**: Validar que el administrador pueda crear cuentas de usuarios con rol de empleado o agente.
* **Parámetros**:

{  
 "nombre": "Carlos Gómez",  
 "email": "carlos.gomez@empresa.com",  
 "password": "Password123",  
 "rol": "AGENTE"  
}

* **Headers**: Content-Type: application/json Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 201 Created
* **Prueba Exitosa si**:
  + El sistema devuelve código 201.
  + La respuesta incluye los datos del usuario creado (id, nombre, email, rol).
* CP-57
* **Caso de Prueba**: Datos inválidos al crear cuenta de usuario
* **Endpoint**: localhost:8080/api/usuarios
* **Método HTTP**: POST
* **Descripción**: Validar que no se permita crear un usuario con datos incompletos o inválidos.
* **Parámetros**:

{  
 "nombre": "Carlos Gómez",  
 "email": "carlos.gomez@empresa.com",  
 "password": "Password123",  
 "rol": "AGENTE"  
}

* **Headers**: Content-Type: application/json Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 400 Bad Request
* **Prueba Exitosa si**:
  + El sistema devuelve código 400.
  + La respuesta incluye un mensaje que indica qué datos son inválidos o faltan.
* CP-58
* **Caso de Prueba**: Rol o información requerida no encontrada al crear cuenta
* **Endpoint**: localhost:8080/api/usuarios
* **Método HTTP**: POST
* **Descripción**: Validar que el sistema notifique cuando el rol o información asociada no existe.
* **Parámetros**:

{  
 "nombre": "Carlos Gómez",  
 "email": "carlos.gomez@empresa.com",  
 "password": "Password123",  
 "rol": "AGENTE"  
}

* **Headers**: Content-Type: application/json Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 404 Not Found
* **Prueba Exitosa si**:
  + El sistema devuelve código 404.
  + La respuesta incluye un mensaje indicando que el rol o recurso requerido no fue encontrado.
* CP-59
* **Caso de Prueba**: Error del servidor al registrar cuenta de usuario
* **Endpoint**: localhost:8080/api/usuarios
* **Método HTTP**: POST
* **Descripción**: Validar que el sistema maneje errores internos al intentar crear una cuenta.
* **Parámetros**:

{  
 "nombre": "Carlos Gómez",  
 "email": "carlos.gomez@empresa.com",  
 "password": "Password123",  
 "rol": "AGENTE"  
}

* **Headers**: Content-Type: application/json Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 500 Internal Server Error
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 500 Internal Server Error.
  + La respuesta incluye un mensaje de error genérico sin exponer detalles.
* CP-60
* **Caso de Prueba**: Ver una tabla de todos los tickets, filtrada por estado, prioridad, agente o fecha
* **Endpoint**: localhost:8080/api/tickets?estado=?&prioridad=?&agenteId=?&fecha?
* **Método HTTP**: GET
* **Descripción**: Validar que el sistema permita consultar la lista de tickets aplicando filtros por estado, prioridad, agente o fecha.
* **Parámetros**: No aplica.
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 200 OK
* **Prueba Exitosa si**:
  + El sistema devuelve código 200.
  + La respuesta incluye una lista de tickets que cumplen con los filtros aplicados.
* CP-61
* **Caso de Prueba**: Filtros inválidos al consultar tabla de tickets
* **Endpoint**: localhost:8080/api/tickets?estado=?&prioridad=?&agenteId=?&fecha?
* **Método HTTP**: GET
* **Descripción**: Validar que el sistema rechace solicitudes con filtros inválidos o valores no reconocidos.
* **Parámetros**: No aplica.
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 400 Bad Request
* **Prueba Exitosa si**:
  + El sistema devuelve código 400.
  + La respuesta incluye un mensaje que indique qué filtros son inválidos.
* CP-62
* **Caso de Prueba**: No se encontraron tickets con los filtros aplicados
* **Endpoint**: localhost:8080/api/tickets?estado=?&prioridad=?&agenteId=?&fecha?
* **Método HTTP**: GET
* **Descripción**: Validar que el sistema informe cuando no hay tickets que coincidan con los filtros seleccionados.
* **Parámetros**: No aplica.
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 404 Not Found
* **Prueba Exitosa si**:
  + El sistema devuelve código 404.
  + La respuesta incluye un mensaje indicando que no existen tickets con los filtros aplicados.
* CP-63
* **Caso de Prueba**: Error del servidor al obtener tickets filtrados
* **Endpoint**: localhost:8080/api/tickets?estado=?&prioridad=?&agenteId=?&fecha?
* **Método HTTP**: GET
* **Descripción**: Validar que el sistema maneje correctamente errores internos al consultar la tabla de tickets.
* **Parámetros**: No aplica.
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 500 Internal Server Error
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 500 Internal Server Error.
  + La respuesta incluye un mensaje de error genérico sin exponer detalles.
* CP-64
* **Caso de Prueba**: Exportar informes del sistema en PDF
* **Endpoint**: localhost:8080/api/admin/estadisticas/exportar/pdf
* **Método HTTP**: GET
* **Descripción**: Validar que el administrador pueda exportar correctamente los informes del sistema en formato PDF.
* **Parámetros**: No aplica.
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 200 OK
* **Prueba Exitosa si**:
  + El sistema devuelve código 200.
  + El archivo PDF se descarga correctamente y contiene los datos del informe solicitado.
* CP-65
* **Caso de Prueba**: Parámetros inválidos al exportar informe
* **Endpoint**: localhost:8080/api/admin/estadisticas/exportar/pdf
* **Método HTTP**: GET
* **Descripción**: Validar que el sistema rechace solicitudes con parámetros inválidos al exportar informes en PDF.
* **Parámetros**: No aplica.
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 400 Bad Request
* **Prueba Exitosa si**:
  + El sistema devuelve código 400.
  + El mensaje de error indica que los parámetros enviados son inválidos o mal formateados.
* CP-66
* **Caso de Prueba**: Datos del informe no encontrados
* **Endpoint**: localhost:8080/api/admin/estadisticas/exportar/pdf
* **Método HTTP**: GET
* **Descripción**: Validar que el sistema notifique adecuadamente cuando no existen datos para generar el informe en PDF.
* **Parámetros**: No aplica.
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 404 Not Found
* **Prueba Exitosa si**:
  + El sistema devuelve código 404.
  + El mensaje indica que no se encontraron datos para generar el informe.
* CP-67
* **Caso de Prueba**: Error del servidor al generar archivo PDF
* **Endpoint**: localhost:8080/api/admin/estadisticas/exportar/pdf
* **Método HTTP**: GET
* **Descripción**: Validar que el sistema maneje correctamente un error interno inesperado al generar el archivo PDF.
* **Parámetros**: No aplica.
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 500 Internal Server Error
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 500 Internal Server Error.
  + La respuesta incluye un mensaje de error genérico sin exponer detalles.
* CP-68
* **Caso de Prueba**: Ver un dashboard con métricas
* **Endpoint**: localhost:8080/api/admin/estadisticas
* **Método HTTP**: GET
* **Descripción**: Validar que el administrador pueda visualizar correctamente el dashboard con métricas generales del sistema.
* **Parámetros**: No aplica.
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 200 OK
* **Prueba Exitosa si**:
  + El sistema devuelve código 200.
  + La respuesta contiene métricas en formato JSON (ej. tickets abiertos, cerrados, usuarios activos).
* CP-69
* **Caso de Prueba**: Solicitud inválida al cargar dashboard
* **Endpoint**: localhost:8080/api/admin/estadisticas
* **Método HTTP**: GET
* **Descripción**: Validar que el sistema responda adecuadamente a parámetros inválidos en la consulta de métricas.
* **Parámetros**: No aplica.
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 400 Bad Request
* **Prueba Exitosa si**:
  + El sistema devuelve código 400.
  + El mensaje de error indica que los parámetros enviados son inválidos o mal formateados.
* CP-70
* **Caso de Prueba**: Datos de métricas no encontrados
* **Endpoint**: localhost:8080/api/admin/estadisticas
* **Método HTTP**: GET
* **Descripción**: Validar que el sistema indique cuando no existen datos para mostrar métricas en el dashboard.
* **Parámetros**: No aplica.
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 404 Not Found
* **Prueba Exitosa si**:
  + El sistema devuelve código 404.
  + El mensaje indica que no se encontraron métricas en el rango solicitado.
* CP-71
* **Caso de Prueba**: Error del servidor al mostrar dashboard
* **Endpoint**: localhost:8080/api/admin/estadisticas
* **Método HTTP**: GET
* **Descripción**: Validar que el sistema maneje correctamente un error interno inesperado al mostrar métricas.
* **Parámetros**: No aplica.
* **Headers**: Authorization: Bearer <token>
* **Código de Respuesta Esperado**: 500 Internal Server Error
* **Prueba Exitosa si**:
  + El sistema responde con código HTTP 500 Internal Server Error.
  + La respuesta incluye un mensaje de error genérico sin exponer detalles.