

Pruebas de Integración – Sistema Tenis Master

Documento Completo y Detallado

Versión: 1.0

Fecha: 2025

Equipo: RedSoft – Proyecto Integrador

1. Introducción a las Pruebas de Integración

Las pruebas de integración verifican el correcto funcionamiento del sistema Tenis Master cuando múltiples componentes interactúan entre sí. Estas pruebas evalúan flujos completos que incluyen:

- Interacción entre vistas (Pug), rutas (Express) y controladores.
- Consulta e inserción en la base de datos MongoDB.
- Manejo de sesiones y roles.
- Validación de flujos reales realizados por un usuario.

Estas pruebas aseguran que el sistema funcione como un conjunto cohesivo y permita acciones consistentes desde la interfaz hasta la persistencia de datos.

2. Prueba de Integración Nº1: Creación de Reserva (Rol Cliente)

2.1 Objetivo

Validar que un usuario con rol Cliente pueda completar el formulario de reserva, y que dicha reserva sea correctamente registrada en la base de datos y mostrada en su panel de reservas.

2.2 Componentes Involucrados

- Vista: reservar.pug
- Ruta: POST /reservar (reservaRoutes.js)
- Controlador: crearReserva() (reservaController.js)
- Modelo: Reserva.js
- Vista destino: panel-reservas-cliente.pug

2.3 Pasos Realizados

Paso 1 — Inicio de Sesión:

El usuario ingresa con rol Cliente (cliente@mail.com). El sistema lo redirige al panel de cliente (
/panel-reservas).

Paso 2 — Acceso a la Vista de Reserva:

El cliente selecciona 'Reservar', lo que dispara GET /reservar.

Paso 3 — Completar Formulario:

Se cargan los siguientes datos:

- Fecha: 2025-11-20
- Hora inicio: 10:00
- Hora fin: 11:30
- Cancha: Cancha 1

Los campos de nombre/apellido aparecen precargados y bloqueados.

The screenshot shows a mobile application interface for reserving a tennis court. At the top, there's a navigation bar with a logo, the text 'TENIS MASTER', and buttons for 'Ver reservas', 'Volver', and 'Salir'. The main title 'Reservar cancha' is centered above a form. The form has four input fields: 'Apellido' (Last Name) containing 'Arias', 'Nombre' (First Name) containing 'Julieta', 'Fecha' (Date) showing '20/11/2025', and 'Cancha' (Court) showing 'Cancha 1'. Below these are two time selection fields: 'Hora inicio' (Start Time) set to '10:00' and 'Hora fin' (End Time) set to '11:30'. A large green button at the bottom right is labeled 'Confirmar reserva' (Confirm reservation).

Paso 4 — Envío del Formulario:

El usuario confirma, activando la ruta POST /reservar.

Paso 5 — Lógica del Controlador (crearReserva):

- Valida campos obligatorios.
- Normaliza fecha.
- Verifica solapamientos.
- Inserta la reserva como documento en MongoDB.
- Redirige al panel del cliente.

Paso 6 — Visualización:

Panel-reservas-cliente.pug muestra la nueva reserva en la tabla de próximas reservas.



2.4 Resultado Esperado

- La reserva se inserta en la colección 'reservas'.
- Se visualiza en el panel del cliente como estado 'activa'.
- No ocurren errores en consola.

The screenshot shows the MongoDB Compass interface. On the left, the 'Connections' sidebar lists 'cluster0.xngxfwn.mongodb.net' with its databases: 'Proyecto_Integrador' (selected), 'admin', 'config', 'local', and 'proyecto_integrador'. Inside 'proyecto_integrador', there are collections: 'pagos', 'reservas' (selected), 'sessions', and 'usuarios'. The main area displays the 'reservas' collection with 33 documents. A specific document is selected, showing the following fields:

```

_id: ObjectId('691cb56c2fdd3c88404692f9')
jugadorId: ObjectId('6913aa4c2458a7f0593ae6b9')
jugadorEmail: "julieta.arias@gmail.com"
fecha : 2025-11-20T03:00:00.000+00:00
horaInicio : "10:00"
horaFin : "11:30"
cancha : "Cancha 1"
estado : "activa"
createdBy : ObjectId('6913aa4c2458a7f0593ae6b9')
createdAt : 2025-11-18T18:05:32.580+00:00
updatedAt : 2025-11-18T18:05:32.580+00:00
__v : 0

```

2.5 Resultado Obtenido

- La reserva fue creada correctamente.
- MongoDB muestra el nuevo documento.
- La interfaz refleja la reserva con los datos correctos.
- No se detectaron errores en el servidor.

3. Prueba de Integración Nº2: Pago de Reserva (Rol Admin/Empleado)

3.1 Objetivo

Confirmar que un usuario con rol Administrador o Empleado pueda ingresar al flujo de pago de una reserva activa, y que el sistema registre el pago, actualice la reserva y muestre el cambio en el panel administrativo.

3.2 Componentes Involucrados

- Vista origen: panel-reservas-admin.pug
- Ruta de acceso al pago: GET /reservas/:id/pago
- Ruta de confirmación: POST /reservas/:id/pagar
- Controladores: mostrarPago() y pagarReserva()
- Modelos: Reserva.js y Pago.js
- Vista final: panel-reservas-admin.pug

3.3 Pasos Realizados

Paso 1 — Inicio de Sesión:

El administrador inicia sesión (admin@mail.com) y accede al panel de reservas.

Paso 2 — Selección de Reserva:

Se identifica una reserva activa con duración 10:00–11:30.

Fecha	Cancha	Inicio	Fin	Cliente	Estado	Acciones
19/11/2025	Cancha 1	10:00	11:00	Alvarez Julian	activa	Pagar Cancelar
20/11/2025	Cancha 1	10:00	11:30	Arias Julieta	activa	Pagar Cancelar
20/11/2025	Cancha 3	14:00	16:30	Herrera Gonzalo	pagado	
20/11/2025	Cancha 2	18:00	19:00	Cruz Laura	activa	Pagar Cancelar
20/11/2025	Cancha 1	18:00	20:00	Campos Jose Luis	activa	Pagar Cancelar

Paso 3 — Acceso a Pago:

Al presionar 'Pagar', se ejecuta GET /reservas/:id/pago.

La vista pago.pug muestra datos del cliente, cancha, fecha y total calculado.

Paso 4 — Confirmación de Pago:

Se elige método 'Tarjeta', se ingresan datos y se envía el formulario.

Resumen de la reserva

Cliente: Arias Julieta | Correo electrónico: julieta.arias@gmail.com | Estado: activa

Cancha: Cancha 1	Fecha: jueves, 20 de noviembre de 2025	Horario: 10:00–11:30	Importe: \$ 36000 (Precio por hora: \$ 24000)
------------------	--	----------------------	--

Método: Tarjeta

Titular: Arias Julieta	Número de tarjeta: 5555 2222 9999 8888
Vencimiento: 11/29	CVV: 456

Pagar y confirmar

Paso 5 — Lógica del Controlador (pagarReserva):

- Obtiene la reserva.
- Valida que esté activa.
- Calcula importe:

$10:00-11:30 = 90 \text{ minutos} = 3 \text{ bloques de } 30 \text{ min.}$

Precio por bloque = $24000 / 2 = 12000$.

Total = 36000.

- Inserta documento en colección 'pagos'.

CONNECTIONS (2)

Search connections

- cluster0.xngxfwn.mongodb.net
- Proyecto_Integrador
 - admin
 - config
 - local
 - proyecto_integrador
 - pagos**
 - reservas
 - sessions
 - usuarios

Type a query: { field: 'value' } or [Generate query](#)

ADD DATA **EXPORT DATA** **UPDATE** **DELETE**

```

{
  "_id": ObjectId('691cba622fdd3c8840469306'),
  "reservaId": ObjectId('691cb56c2fdd3c88404692f9'),
  "clienteId": ObjectId('6913aa4c2458a7f0593ae6b9'),
  "metodo": "tarjeta",
  "titular": "Arias Julieta",
  "ultimos4": "8888",
  "vencimiento": "11/29",
  "importe": 36000,
  "fechaPago": 2025-11-18T18:26:42.726+00:00,
  "registradoPor": ObjectId('685bde489baf98da6627d25b'),
  "createdAt": 2025-11-18T18:26:42.728+00:00,
  "updatedAt": 2025-11-18T18:26:42.728+00:00
}
  
```

- Actualiza la reserva a estado 'pagado'.
- Redirige al panel administrativo.

3.4 Resultado Esperado

- Se crea un documento en la colección 'pagos' con importe = 36000.
- La reserva cambia su estado a 'pagado'.
- La interfaz no permite volver a pagar la misma reserva.

3.5 Resultado Obtenido

- El pago se registró con éxito.
- La reserva aparece como 'pagado' en panel-reservas-admin.
- No se generaron errores en consola ni en servidor.

Fecha	Cancha	Inicio	Fin	Cliente	Estado	Acciones
19/11/2025	Cancha 1	10:00	11:00	Alvarez Julian	activa	Pagar Cancelar
20/11/2025	Cancha 1	10:00	11:30	Arias Julieta	pagado	
20/11/2025	Cancha 3	14:00	16:30	Herrera Gonzalo	pagado	
20/11/2025	Cancha 2	18:00	19:00	Cruz Laura	activa	Pagar Cancelar

4. Conclusiones Generales

Ambas pruebas de integración confirmaron que los módulos del sistema Tenis Master interactúan correctamente entre sí. Los flujos críticos —creación de reservas y pago— funcionan como un proceso sólido conectando vistas, rutas, controladores y modelos. El sistema responde adecuadamente a acciones reales de usuarios con roles distintos.