

Guía de Setup - Módulo de Reservas (Bookings)

✓ Estado de Implementación

Módulo completamente implementado y listo para usar.

Archivos Creados

API Routes (7 archivos)

- ✓ `/app/api/bookings/route.ts` - GET, POST
- ✓ `/app/api/bookings/[id]/route.ts` - GET, PUT, DELETE
- ✓ `/app/api/bookings/[id]/confirm/route.ts` - POST
- ✓ `/app/api/bookings/[id]/start/route.ts` - POST
- ✓ `/app/api/bookings/[id]/complete/route.ts` - POST
- ✓ `/app/api/bookings/check-availability/route.ts` - POST
- ✓ `/app/api/bookings/stats/route.ts` - GET
- ✓ `/app/api/customers/route.ts` - GET (nuevo)

Componentes (5 archivos)

- ✓ `/components/bookings/BookingList.tsx`
- ✓ `/components/bookings/BookingForm.tsx`
- ✓ `/components/bookings/BookingCard.tsx`
- ✓ `/components/bookings/BookingCalendar.tsx`
- ✓ `/components/bookings/AvailabilityChecker.tsx`

Páginas (4 archivos)

- ✓ `/app/bookings/page.tsx` - Lista de reservas
- ✓ `/app/bookings/new/page.tsx` - Crear reserva
- ✓ `/app/bookings/[id]/page.tsx` - Detalle de reserva
- ✓ `/app/bookings/calendar/page.tsx` - Vista de calendario

Actualizaciones

- ✓ `/app/dashboard/page.tsx` - Actualizado con estadísticas de reservas
- ✓ `/prisma/seed.ts` - Actualizado con clientes y reservas de ejemplo

Documentación

- ✓ `BOOKINGS_MODULE_DOCUMENTATION.md` - Documentación completa
 - ✓ `BOOKINGS_SETUP_GUIDE.md` - Esta guía
-

Pasos para Activar el Módulo

1. Verificar Dependencias Instaladas

```
cd /home/ubuntu/rental_management

# Las siguientes dependencias ya están instaladas:
# - react-big-calendar
# - date-fns
```

2. Configurar Base de Datos

Asegúrate de tener PostgreSQL corriendo y configurado en tu `.env` :

```
DATABASE_URL="postgresql://usuario:password@localhost:5432/rental_management?
schema=public"
```

3. Ejecutar Migraciones

El modelo `Booking` ya está en el schema de Prisma. Ejecuta:

```
npx prisma migrate dev --name add_bookings_module
```

4. Generar Cliente de Prisma

```
npx prisma generate
```

5. Poblar Base de Datos (Seed)

```
npx prisma db seed
```

Esto creará:

- 4 clientes de ejemplo para cada tenant
- 6-10 reservas de ejemplo en diferentes estados
- Reservas pasadas, actuales y futuras

6. Iniciar Servidor de Desarrollo

```
npm run dev
```

7. Acceder al Módulo

Añade a tu `/etc/hosts` (si no lo has hecho):

```
127.0.0.1 demo.localhost test.localhost scooters-madrid.localhost boats-marbella.localhost
```

Accede a:

```
http://demo.localhost:3000/bookings
```

Credenciales:

- Email: `owner@demo.com`
- Password: `password123`

Verificación Post-Setup

Checklist de Verificación

- [] **Dashboard actualizado:** Visita `/dashboard` y verifica que aparecen:
 - Estadísticas de inventario
 - Estadísticas de reservas (Total, Pendientes, Confirmadas, etc.)
 - Ingresos totales y pendientes
 - Próximas reservas (si hay)
- [] **Lista de reservas:** Visita `/bookings` y verifica:
 - Se muestran todas las reservas
 - Los filtros por estado funcionan
 - La búsqueda funciona
 - Las estadísticas son correctas
- [] **Crear reserva:** Visita `/bookings/new` y verifica:
 - Se pueden seleccionar items
 - Se pueden seleccionar clientes
 - La verificación de disponibilidad funciona
 - El cálculo de precio es automático
 - Se puede crear la reserva
- [] **Detalle de reserva:** Haz click en una reserva y verifica:
 - Se muestra toda la información
 - Los botones de acción aparecen según el estado
 - Las acciones funcionan (confirmar, iniciar, completar, cancelar)
- [] **Calendario:** Visita `/bookings/calendar` y verifica:
 - Se muestran todas las reservas
 - Los eventos tienen colores según estado
 - Click en evento redirige a detalle
 - Click en slot vacío redirige a crear reserva

Características Implementadas

Gestión de Reservas

- ☒ Crear reserva con validación completa
- ☒ Ver lista de reservas con filtros
- ☒ Ver detalle completo de reserva
- ☒ Actualizar reserva
- ☒ Cancelar reserva
- ☒ Confirmar reserva (PENDING → CONFIRMED)
- ☒ Iniciar reserva (CONFIRMED → IN_PROGRESS)
- ☒ Completar reserva (IN_PROGRESS → COMPLETED)

Sistema de Disponibilidad

- ☒ Verificación automática al seleccionar fechas
- ☒ Detección de conflictos con otras reservas
- ☒ Exclusión de reservas canceladas/completadas
- ☒ Feedback visual en tiempo real

Cálculo de Precios

- ☒ Cálculo automático basado en días
- ☒ Precio = días × basePrice del item
- ☒ Sugerencia de depósito (20% del total)
- ☒ Ajuste manual permitido

Dashboard

- ☒ Estadísticas de reservas por estado
- ☒ Ingresos completados y pendientes
- ☒ Próximas reservas (7 días)
- ☒ Accesos rápidos

Vista de Calendario

- ☒ Calendario visual con react-big-calendar
- ☒ Eventos coloreados por estado
- ☒ Interacción (click en evento o slot vacío)
- ☒ Localización en español

Troubleshooting

Problema: Error “Can’t reach database server”

Solución: Asegúrate de que PostgreSQL está corriendo y que el `DATABASE_URL` en `.env` es correcto.

```
# Verifica que PostgreSQL está corriendo
sudo systemctl status postgresql

# O si usas Docker
docker ps | grep postgres
```

Problema: “Prisma Client is not generated”

Solución: Ejecuta:

```
npx prisma generate
```

Problema: No aparecen las nuevas páginas

Solución: Reinicia el servidor de desarrollo:

```
# Ctrl+C para detener
npm run dev
```

Problema: Error 404 en /bookings

Solución: Verifica que estás autenticado y accediendo con el subdominio correcto (ej: `demo.localhost:3000`).

Problema: No se ven los estilos del calendario

Solución: Verifica que `react-big-calendar` esté instalado:

```
npm list react-big-calendar
```

Si no está instalado:

```
npm install react-big-calendar date-fns
```



Datos de Ejemplo Creados por el Seed

Demo Tenant (subdomain: demo)

Clientes:

1. John Smith (PASSPORT: AB123456)
2. Maria García (DNI: 12345678X)
3. Pierre Dubois (PASSPORT: FR987654)
4. Emma Wilson (DRIVING_LICENSE: UK123456789)

Reservas:

- 1 Completada (mes pasado)
- 1 En Progreso (ayer → próxima semana)
- 1 Confirmada (mañana → próxima semana)
- 1 Pendiente (próxima semana → próximo mes)

- 1 Cancelada
- 1 Completada adicional (hace 20-15 días)

Items:

- Honda PCX 125
- Yamaha NMAX 125
- Vespa Primavera 150 (RENTED)
- Piaggio Liberty 125 (MAINTENANCE)

Scooters Madrid (subdomain: scooters-madrid)**Clientes:**

1. Ana Martínez
2. Luis Fernández

Reservas:

- 1 Confirmada (mañana)
- 1 Completada (hace 5 días)

Boats Marbella (subdomain: boats-marbella)**Clientes:**

1. Robert Johnson (UK)
2. Hans Mueller (Germany)
3. Sofia Rossi (Italy)

Reservas:

- 1 Confirmada (próximos 5-8 días) - €2,400
- 1 Pendiente (mañana)
- 1 Completada (hace 10-7 días) - €1,050

Próximos Pasos Recomendados

1. Módulo de Clientes Completo

- Crear `/app/customers/page.tsx` para lista
- Crear `/app/customers/new/page.tsx` para formulario
- Crear `/app/customers/[id]/page.tsx` para detalle

2. Sistema de Notificaciones

- Email de confirmación de reserva
- Recordatorio 24h antes
- Notificación de cancelación

3. Pagos e Invoices

- Integración con Stripe/PayPal
- Generación de facturas PDF
- Registro de pagos

4. Testing Automatizado

- Unit tests para API routes
- Integration tests para flujos completos
- E2E tests con Playwright

Soporte







Si tienes problemas durante el setup:

1. Revisa la sección **Troubleshooting** arriba
2. Consulta la documentación completa en `BOOKINGS_MODULE_DOCUMENTATION.md`
3. Verifica que todos los archivos se hayan creado correctamente
4. Asegúrate de que las dependencias estén instaladas

Resumen

El módulo de Reservas está 100% implementado y listo para usar.

Una vez que configures la base de datos y ejecutes las migraciones + seed, tendrás:

-  Sistema completo de gestión de reservas
-  Verificación de disponibilidad en tiempo real
-  Cálculo automático de precios
-  Vista de calendario interactiva
-  Dashboard con estadísticas
-  Datos de ejemplo para probar

¡Disfruta del nuevo módulo de Reservas! 