



# Guía para Desarrollo Local con Cursor



## Configuración Completada para Cursor



### Archivos Agregados para Desarrollo



#### Configuraciones de Cursor/VSCoDe

- `.vscode/settings.json` - Configuraciones optimizadas para TypeScript y Next.js
- `.vscode/extensions.json` - Extensiones recomendadas para el proyecto
- `.vscode/launch.json` - Configuraciones de debugging
- `.vscode/tasks.json` - Tareas automatizadas para desarrollo



#### Variables de Entorno

- `.env.example` - Plantilla de configuración para desarrollo local
- **Variables actuales preservadas** en `.env` (para referencia)



## Instrucciones de Setup Local

### 1. Prerequisitos

```
# Asegúrate de tener instalado:  
- Node.js 18+  
- Yarn (recomendado) o npm  
- PostgreSQL 14+  
- Git  
- Cursor Editor o VSCode
```

### 2. Descomprimir y Configurar

```
# Extraer el proyecto  
tar -xzf hotel_pms_completo_TIMESTAMP.tar.gz  
# o usar herramientas gráficas para ZIP  
  
# Navegar al proyecto  
cd hotel_pms_paseo_las_mercedes
```

### 3. Abrir en Cursor

```
# Opción 1: Desde terminal  
cursor .  
  
# Opción 2: Desde Cursor  
File → Open Folder → Seleccionar hotel_pms_paseo_las_mercedes
```

## 4. Configurar Entorno de Desarrollo

```
# Navegar al directorio de la app
cd app

# Instalar dependencias
yarn install

# Configurar variables de entorno
cp .env.example .env
# Editar .env con tus configuraciones locales
```

## 5. Configurar Base de Datos Local

### Opción A: PostgreSQL Local

```
# Crear base de datos
createdb hotel_pms_dev

# Editar .env con tu URL local:
# DATABASE_URL="postgresql://usuario:contraseña@localhost:5432/hotel_pms_dev"

# Ejecutar migraciones
npx prisma migrate deploy

# Generar cliente Prisma
npx prisma generate

# Poblar con datos de prueba
npm run seed
```

### Opción B: Docker PostgreSQL

```
# Crear contenedor PostgreSQL
docker run --name hotel-pms-db \
  -e POSTGRES_DB=hotel_pms_dev \
  -e POSTGRES_USER=usuario \
  -e POSTGRES_PASSWORD=contraseña \
  -p 5432:5432 \
  -d postgres:14

# Continuar con migraciones como en Opción A
```

## 6. Iniciar Desarrollo

```
# Modo desarrollo
yarn dev

# Modo desarrollo con Turbo (más rápido)
yarn dev --turbo

# La app estará en http://localhost:3000
```

## Scripts de Desarrollo Útiles

### Scripts Disponibles

```
# Desarrollo
yarn dev                # Servidor de desarrollo
yarn dev --turbo        # Desarrollo con Turbo

# Build y testing
yarn build              # Build de producción
yarn start              # Servidor de producción
yarn lint               # Linting
yarn type-check         # Verificación de tipos (si se agrega)

# Base de datos
npx prisma studio       # Explorador visual DB
npx prisma generate     # Generar cliente
npx prisma migrate dev  # Nueva migración
npx prisma migrate reset # Reset completo
npm run seed            # Poblar datos

# Limpieza
rm -rf .next && rm -rf node_modules/.cache # Limpiar cache
```

## Extensiones Recomendadas para Cursor

### Automáticamente Sugeridas

Al abrir el proyecto, Cursor te sugerirá instalar:

- **Tailwind CSS IntelliSense** - Autocompletado para Tailwind
- **Prettier** - Formateo automático de código
- **Prisma** - Soporte para esquemas de base de datos
- **TypeScript Hero** - Mejor experiencia con TypeScript
- **Code Spell Checker** - Verificación ortográfica
- **Pretty TypeScript Errors** - Errores más legibles
- **Auto Rename Tag** - Renombrado automático de tags
- **Path Intellisense** - Autocompletado de rutas
- **SQLTools** - Conexión y queries a PostgreSQL

## Debugging en Cursor

### Configuraciones Incluidas

#### 1. Debug Server-Side

- Debuggea APIs y componentes server-side
- Puntos de interrupción en API routes


#### 2. Debug Client-Side

- Debuggea componentes React en el navegador
- Inspección de estado y props

### 3. Debug Full Stack

- Debugging completo frontend + backend
- Abre automáticamente el navegador

### Usar Debugging

1. Ir a **Run and Debug** (Ctrl+Shift+D)
2. Seleccionar configuración deseada
3. Presionar **F5** o clic en 



## Paneles Útiles en Cursor

### Panel de Tareas

- **Ctrl+Shift+P** → “Tasks: Run Task”
- Ejecutar `dev`, `build`, `prisma:studio`, etc.

### Terminal Integrado

- **Ctrl+`** para abrir terminal
- Múltiples terminales para diferentes comandos

### Explorer de Base de Datos

- Usar SQLTools para conectar a PostgreSQL
- Visualizar tablas y datos directamente en Cursor



## Personalización para tu Flujo

### Configuraciones Adicionales Recomendadas

En `settings.json` personal:

```
{
  "editor.minimap.enabled": true,
  "editor.bracketPairColorization.enabled": true,
  "workbench.iconTheme": "material-icon-theme",
  "terminal.integrated.fontSize": 14,
  "editor.fontSize": 14
}
```



## Flujo de Desarrollo Recomendado

### 1. Workflow Diario

```
# Al iniciar
cursor .           # Abrir proyecto
yarn dev           # Iniciar desarrollo

# Durante desarrollo
npx prisma studio  # Ver/editar datos
git add .          # Guardar cambios
git commit -m "... # Commit
```

## 2. Testing y Debug

- Usar debugging integrado en Cursor
- Panel de problemas para errores TypeScript
- Terminal para comandos de base de datos

## 3. Deployment

```
yarn build          # Verificar build
git push origin master # Subir a GitHub
```

## Funcionalidades Listas para Desarrollo

### ✓ 8 Módulos Principales Implementados

1. 🏠 **Gestión de Habitaciones** - /dashboard/rooms
2. 📅 **Sistema de Reservas** - /dashboard/reservations
3. ✓ **Check-in/Check-out** - /dashboard/check-in
4. 💰 **Facturación y Pagos** - /dashboard/billing
5. 👤 **Gestión de Personal** - /dashboard/staff
6. 📞 **Comunicaciones** - /dashboard/communications
7. 🏠 **Portal de Huéspedes** - /guest-portal
8. 🧹 **Housekeeping** - /dashboard/housekeeping

### 📖 APIs Disponibles

- Todos los endpoints están en /api/
- Documentación en código con tipos TypeScript
- Swagger/OpenAPI ready (se puede implementar)

## Tips para Desarrollo Eficiente

### Shortcuts Útiles en Cursor

- **Ctrl+Shift+P** - Command palette
- **Ctrl+P** - Quick file search
- **Ctrl+Shift+F** - Search in all files
- **F12** - Go to definition
- **Shift+F12** - Find all references
- **Ctrl+D** - Select next occurrence

### AI en Cursor

- **Ctrl+K** - AI code generation
- **Ctrl+L** - AI chat panel
- Usar AI para explicar código complejo
- Pedir sugerencias de mejoras

## Siguientes Pasos de Desarrollo

---

### Mejoras Inmediatas Sugeridas

1. **Configurar Stripe real** (claves de test incluidas)
2. **Setup base de datos production** (URLs incluidas como ejemplo)
3. **Configurar email real** (Resend configurado)
4. **Personalizar branding** (logo, colores, textos)

### Nuevas Funcionalidades

1. **App móvil** con React Native
2. **Notificaciones push**
3. **Integración con sistemas externos**
4. **Multi-idioma (i18n)**
5. **Analytics avanzados**

---

## ¡Proyecto Listo para Cursor!

---

El proyecto está completamente configurado para desarrollo profesional con Cursor. Todas las configuraciones, debugging, extensions y workflows están optimizados para máxima productividad.

¡Feliz coding! 

---

Sistema PMS Hotel Paseo Las Mercedes - Optimizado para Desarrollo Local