

Configuración Específica para Cursor

Optimizaciones Realizadas para Desarrollo Local

Archivos de Configuración Añadidos

Configuraciones de Cursor/VSCode

- `.vscode/settings.json` - Configuraciones optimizadas para TypeScript, Tailwind CSS y Next.js
- `.vscode/extensions.json` - Extensiones recomendadas automáticamente
- `.vscode/launch.json` - Configuraciones de debugging (F5)
- `.vscode/tasks.json` - Tareas automatizadas accesibles desde Command Palette

Formateo y Calidad de Código

- `.prettierrc` - Configuración de Prettier para formateo consistente
- `.prettierignore` - Archivos excluidos del formateo
- **ESLint** - Ya configurado en el proyecto base

Variables de Entorno

- `.env.example` - Plantilla para configuración local
- **Variables sensibles protegidas** - `.env` original preservado como referencia

Scripts de Desarrollo

- `scripts/setup-local.sh` - Setup automático completo
- `scripts/dev-commands.sh` - Comandos útiles para desarrollo
- `hotel-pms.code-workspace` - Workspace multi-folder para Cursor

Funcionalidades Específicas de Cursor

AI Integration

Cursor incluye AI nativo optimizado para este proyecto:

Chat Panel (Ctrl+L)

Ejemplos de prompts útiles:

- "Explica cómo funciona el sistema de reservas"
- "Crea un nuevo endpoint para cancelar reservas"
- "Optimiza el componente de dashboard"
- "Añade validación a este formulario"

Inline Editing (Ctrl+K)

- Selecciona código y pide modificaciones
- Genera funciones completas
- Refactoriza componentes automáticamente





Composer Mode

- Crea archivos completos con AI
- Genera componentes React complejos




- Implementa nuevas funcionalidades end-to-end

Configuraciones Optimizadas





IntelliSense Mejorado

-  Autocompletado de Tailwind CSS
-  Props de componentes Radix UI
-  Tipos de Prisma automáticos
-  Rutas de Next.js inteligentes

Debugging Integrado

-  Breakpoints en API routes
-  React DevTools integration
-  Database query debugging
-  Network requests monitoring

Git Integration

-  Diff viewer integrado
-  Branch management visual
-  Commit history navigation
-  Merge conflict resolution

Paneles y Workflows Útiles

Layout Recomendado

Explorer <ul style="list-style-type: none">- Files- Git Changes- Extensions	Editor Principal <ul style="list-style-type: none">- TypeScript- React JSX- Prisma Schema
Terminal <ul style="list-style-type: none">- yarn dev- prisma studio	AI Chat Panel <ul style="list-style-type: none">- Cursor AI- Code Review

Workflow Optimizado

1. **AI Chat** para planificar features
2. **Composer** para generar código base
3. **Inline editing** para refinamientos
4. **Debug mode** para testing
5. **Git panel** para commits

Temas y Personalización

Extensions Clave para Hotel PMS

- **Tailwind CSS IntelliSense** - Autocompletado de clases
- **Prisma** - Syntax highlighting para schema
- **Thunder Client** - Testing de APIs (alternativa a Postman)









- **GitLens** - Git history avanzado
- **Error Lens** - Errores inline más visibles

Configuraciones Personalizadas

```
// Añadir a settings.json personal:
{
  "workbench.colorTheme": "One Dark Pro",
  "editor.fontFamily": "Fira Code, monospace",
  "editor.fontLigatures": true,
  "editor.fontSize": 14,
  "editor.lineHeight": 1.6,
  "terminal.integrated.fontSize": 13
}
```

Comandos Rápidos para Cursor

Command Palette (Ctrl+Shift+P)





```
> Tasks: Run Task   Start Development
> Tasks: Run Task   Open Prisma Studio
> Tasks: Run Task   Fresh Setup
> Git: Clone  Para clonar desde GitHub
> Developer: Reload Window  Reiniciar Cursor
```

AI Commands





```
Ctrl+K → "Add error handling to this function"
Ctrl+K → "Convert this to TypeScript"
Ctrl+K → "Add unit tests for this component"
Ctrl+L → "How can I optimize this database query?"
```

Optimizaciones Incluidas




Performance


-  TypeScript incremental compilation
-  Next.js Fast Refresh habilitado
-  Tailwind JIT mode
-  Prisma query optimization

Developer Experience

-  Hot reload para cambios instantáneos
-  Error overlay informativo
-  Auto-import para dependencias
-  Path mapping configurado (@/*)

Code Quality

-  ESLint con reglas de Next.js
-  Prettier para formateo automático
-  TypeScript strict mode

-  Import organization automática



Tips Específicos para este Proyecto

Navegación Rápida

- **Ctrl+P** → `dashboard` para ir a páginas del dashboard
- **Ctrl+P** → `api/` para ir a API routes
- **Ctrl+P** → `components/` para componentes
- **Ctrl+P** → `schema.prisma` para base de datos

Debugging Efectivo

- Usar breakpoints en `/api` routes para debugging backend
- React DevTools para debugging de estado
- Network tab para debugging de requests
- Prisma Studio para debugging de datos

AI Assistant Usage

- Pregunta sobre patrones del proyecto existente
- Pide explicaciones de funciones complejas
- Genera nuevos componentes siguiendo el estilo del proyecto
- Optimiza queries de base de datos



¡Proyecto Optimizado para Cursor!

El proyecto ahora incluye todas las configuraciones específicas para un desarrollo eficiente con Cursor, incluyendo AI integration, debugging avanzado, y workflows optimizados.

¡Disfruta del desarrollo con superpoderes de AI!  