

ATLAS

Innovación al alcance de tus manos

MANUAL TECNICO

SISTEMA ATLAS COMPANY 2025

ATLAS.INOVATIONCOMPANY@GMAIL.COM

ATLAS

Innovación al alcance de tus manos

Contenido

1. ARQUITECTURA DEL SISTEMA	3
1.1 Arquitectura General:	3
1.2 Componentes Principales	3
2. TECNOLOGÍAS UTILIZADAS	4
2.1 Frontend Stack	4
2.2 Backend Stack	5
2.3 Base de Datos y Servicios	5
3. ESTRUCTURA DEL PROYECTO	6
3.1 Frontend (/frontend)	6
3.2 Backend (/backend)	7
4. CONFIGURACIÓN DE ENTORNO	8
4.1 Frontend Setup	8
4.2 Backend Setup	9
5. MODELO DE DATOS	10
6. APIs y Endpoints	11
7. AUTENTICACIÓN Y SEGURIDAD	12
7.1 Flujo de Autenticación	12
7.2 Sistema de Roles	12
7.3 Validación de Acceso	12
8. PATRONES DE DESARROLLO	13
8.1 Patron de Gestión de Información	13
8.2 Ejemplo de Patrón de Cache:	13
9. SOLUCIÓN DE PROBLEMAS	14
9.1 Problemas Comunes	14
9.2 Logs de Debugging	14
ARCHIVOS DE REFERENCIA CLAVE	15
CONTACTO TECNICO	15

ATLAS

Innovación al alcance de tus manos

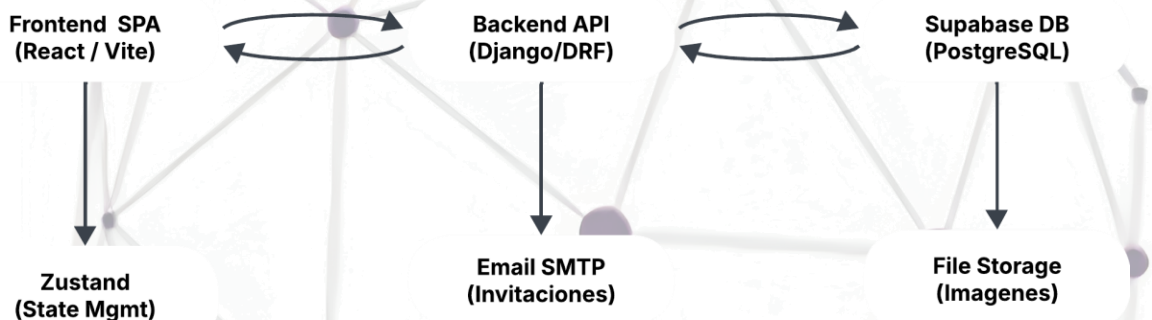
1. ARQUITECTURA DEL SISTEMA

ATLAS SOFT es un sistema de gestión para microempresas de cómputo que permite administrar proyectos, colaboradores, clientes e inventario de equipos de manera integrada y eficiente.

1.1 Arquitectura General:

El sistema ATLAS SOFT utiliza una arquitectura de **Frontend-Backend separados**:

- **Frontend:** React + Vite (SPA)
- **Backend:** Django REST Framework
- **Base de Datos:** Supabase (PostgreSQL)
- **Autenticación:** Supabase Auth
- **Storage:** Supabase Storage



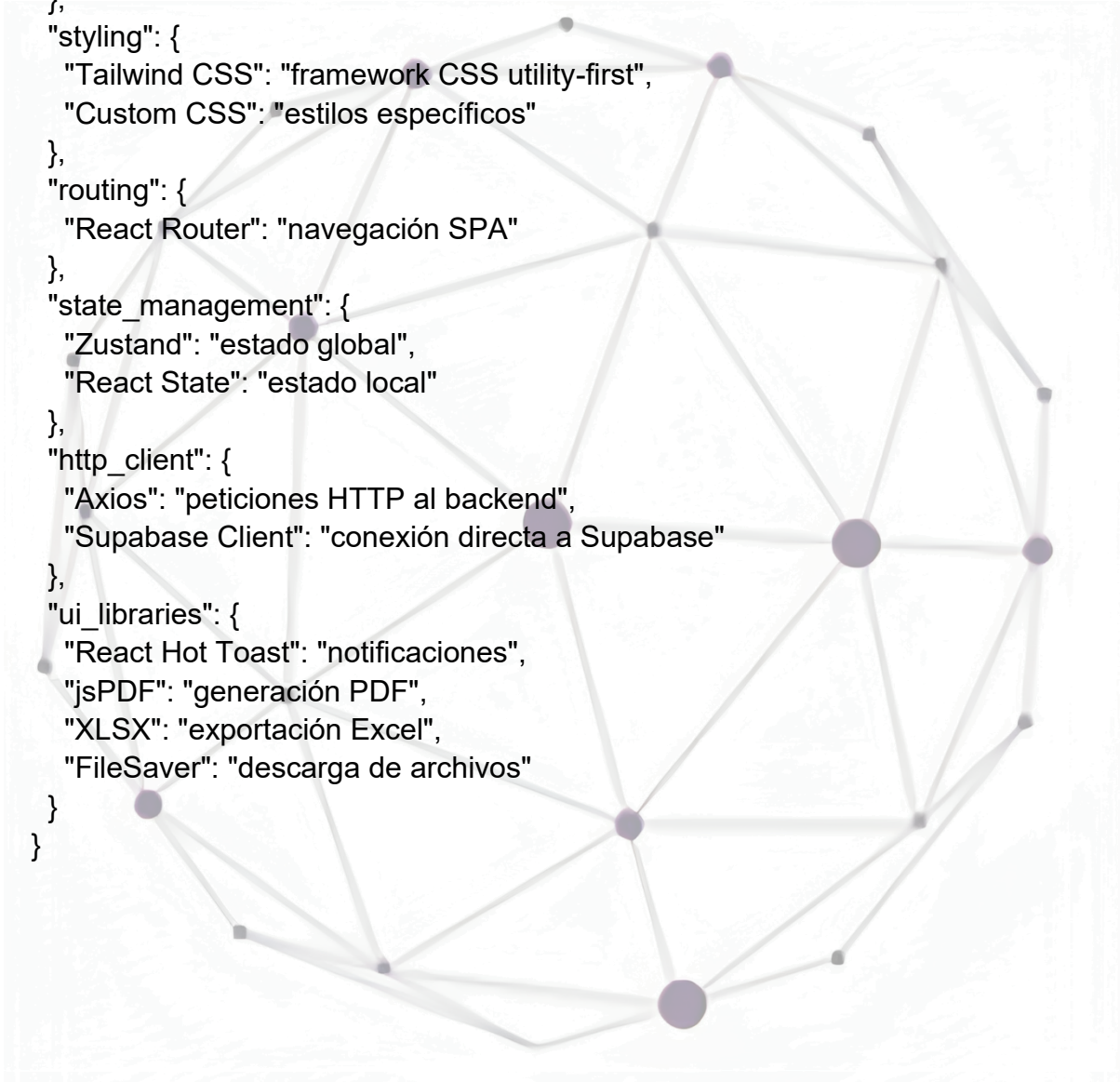
1.2 Componentes Principales

- **Frontend SPA:** React + Vite para interfaz de usuario
- **Backend API:** Django REST Framework para APIs RESTful
- **Base de Datos:** Supabase PostgreSQL para el respaldo
- **Autenticación:** Supabase Auth + Validación de roles en Django
- **Storage:** Supabase Storage para archivos e imágenes

2. TECNOLOGÍAS UTILIZADAS

2.1 Frontend Stack

```
{
  "core": {
    "React": "18+",
    "Vite": "bundler y dev server",
    "JavaScript": "ES6+"
  },
  "styling": {
    "Tailwind CSS": "framework CSS utility-first",
    "Custom CSS": "estilos específicos"
  },
  "routing": {
    "React Router": "navegación SPA"
  },
  "state_management": {
    "Zustand": "estado global",
    "React State": "estado local"
  },
  "http_client": {
    "Axios": "peticiones HTTP al backend",
    "Supabase Client": "conexión directa a Supabase"
  },
  "ui_libraries": {
    "React Hot Toast": "notificaciones",
    "jsPDF": "generación PDF",
    "XLSX": "exportación Excel",
    "FileSaver": "descarga de archivos"
  }
}
```



ATLAS

Innovación al alcance de tus manos

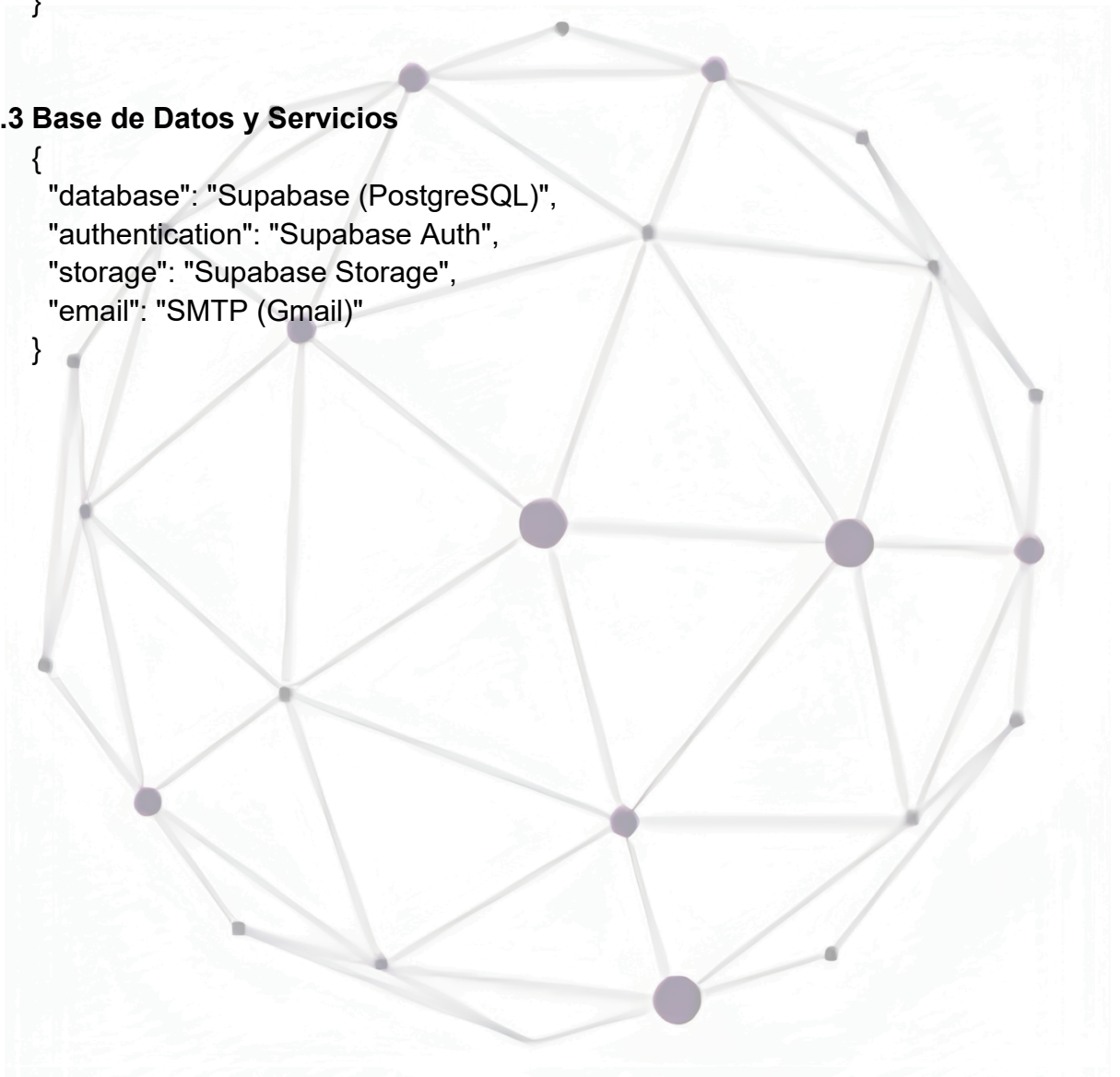
2.2 Backend Stack

Backend principal

```
{  
  "Django": "4.x",  
  "Django REST Framework": "APIs REST",  
  "django-cors-headers": "CORS handling",  
  "Python": "3.8+"  
}
```

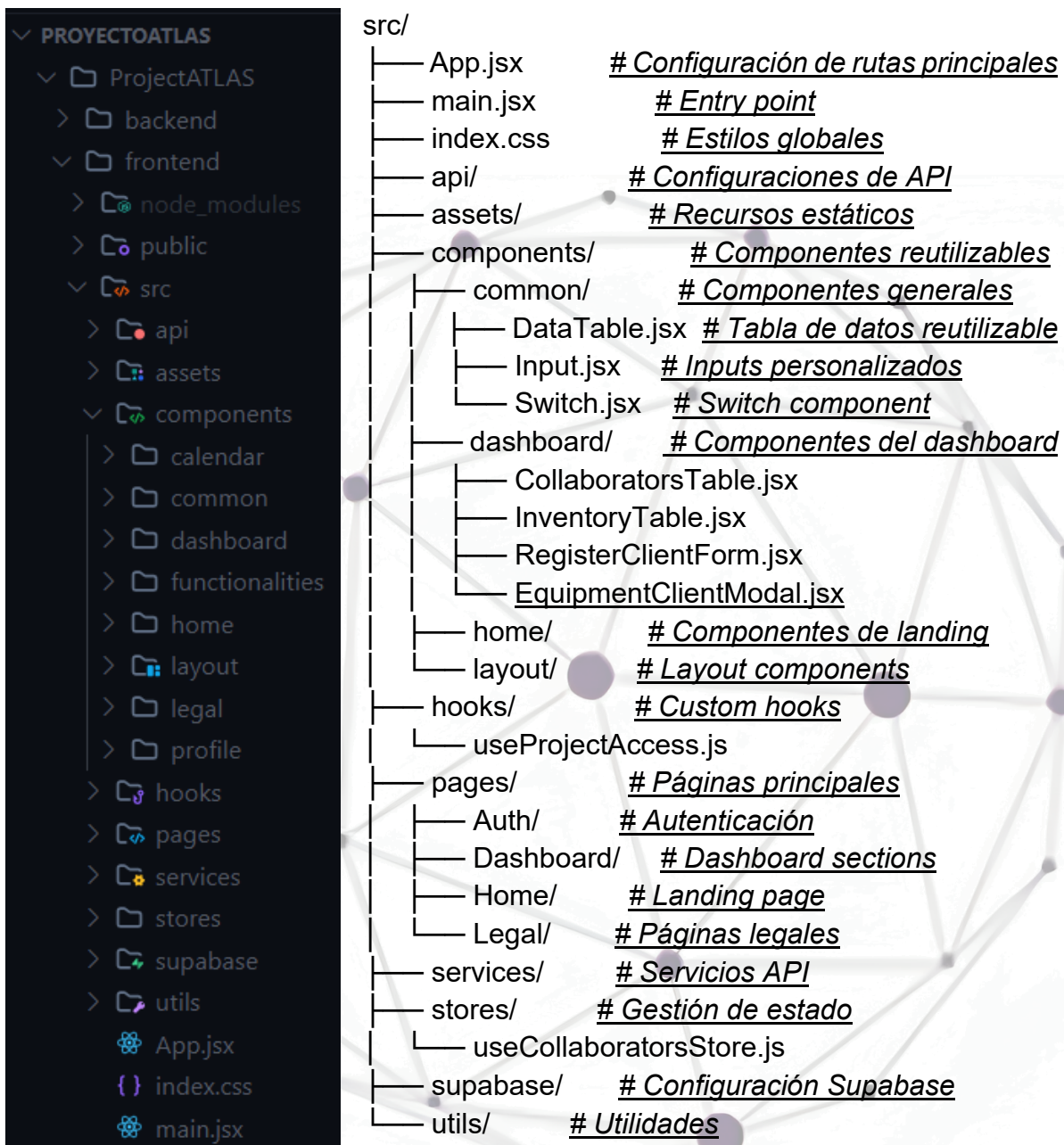
2.3 Base de Datos y Servicios

```
{  
  "database": "Supabase (PostgreSQL)",  
  "authentication": "Supabase Auth",  
  "storage": "Supabase Storage",  
  "email": "SMTP (Gmail)"  
}
```



3. ESTRUCTURA DEL PROYECTO

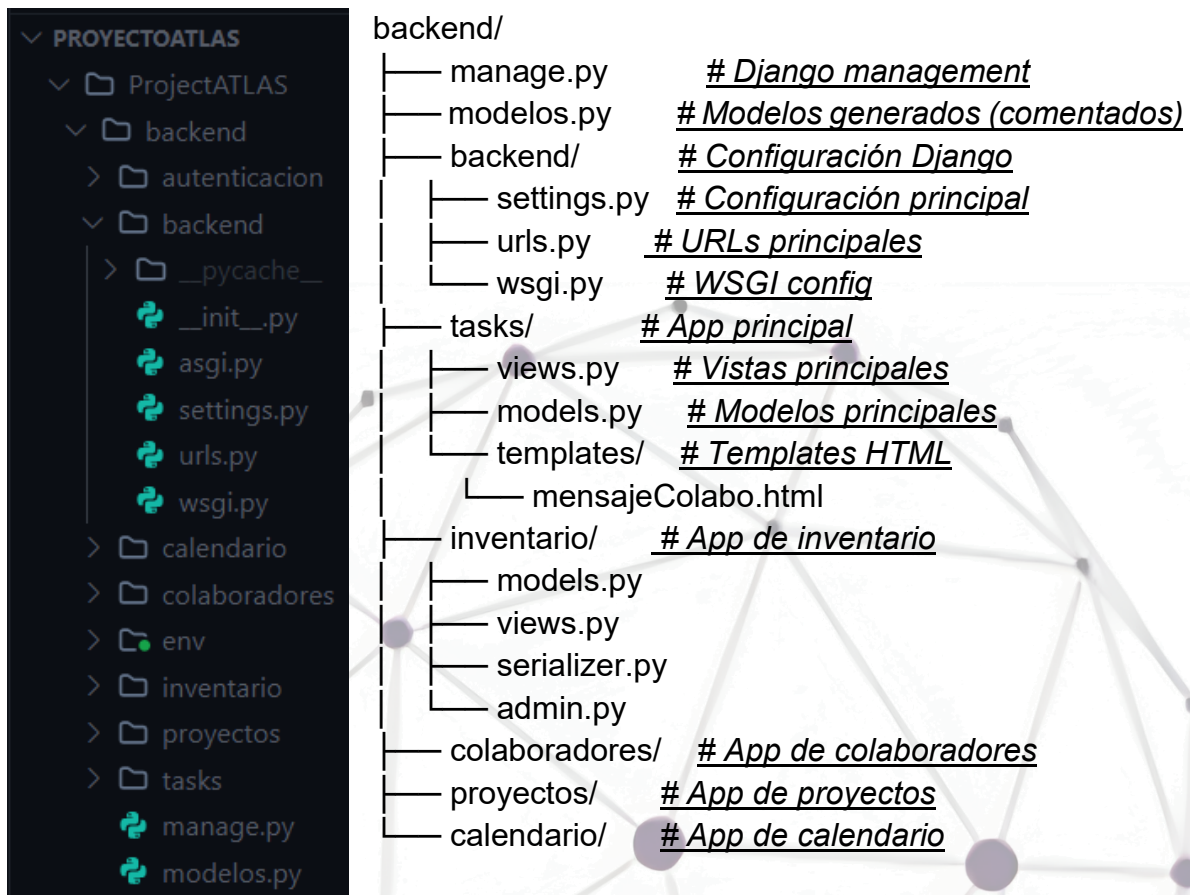
3.1 Frontend (/frontend)



ATLAS

Innovación al alcance de tus manos

3.2 Backend (/backend)



4. CONFIGURACIÓN DE ENTORNO

4.1 Frontend Setup

- **Variables de entorno (.env.local):**

VITE_SUPABASE_URL=https://tu-proyecto.supabase.co

VITE_SUPABASE_ANON_KEY=tu_supabase_anon_key

- **Instalación y Ejecución :**

cd frontend

npm install

npm run dev # Puerto 5173

- **Dependencias Principales (package.json):**

```
{  
  "dependencies": {  
    "react": "^18.0.0",  
    "react-dom": "^18.0.0",  
    "react-router-dom": "navegación",  
    "zustand": "estado global",  
    "axios": "HTTP client",  
    "@supabase/supabase-js": "Supabase client",  
    "tailwindcss": "CSS framework",  
    "react-hot-toast": "notificaciones",  
    "jspdf": "PDF generation",  
    "xlsx": "Excel export"  
  }  
}
```


ATLAS

Innovación al alcance de tus manos

4.2 Backend Setup

- **Configuración Django (settings.py):**

Configuración CORS

```
CORS_ALLOWED_ORIGINS = [  
    "http://localhost:5173",  
    "http://127.0.0.1:5173",  
]
```

```
CORS_ALLOW_ALL_ORIGINS = True # Solo para desarrollo
```

Apps instaladas

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'rest_framework',  
    'corsheaders',  
    'tasks',  
    'inventario',  
    'colaboradores',  
    'proyectos',  
    'calendario',  
]
```

Email configuration

```
EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'  
EMAIL_HOST = 'smtp.gmail.com'  
EMAIL_PORT = 587  
EMAIL_USE_TLS = True
```

- **Instalación y Ejecución:**

```
cd backend  
pip install django djangorestframework django-cors-headers  
python manage.py migrate  
python manage.py runserver # Puerto 8000
```

5. MODELO DE DATOS

Entidades Principales

Usuario

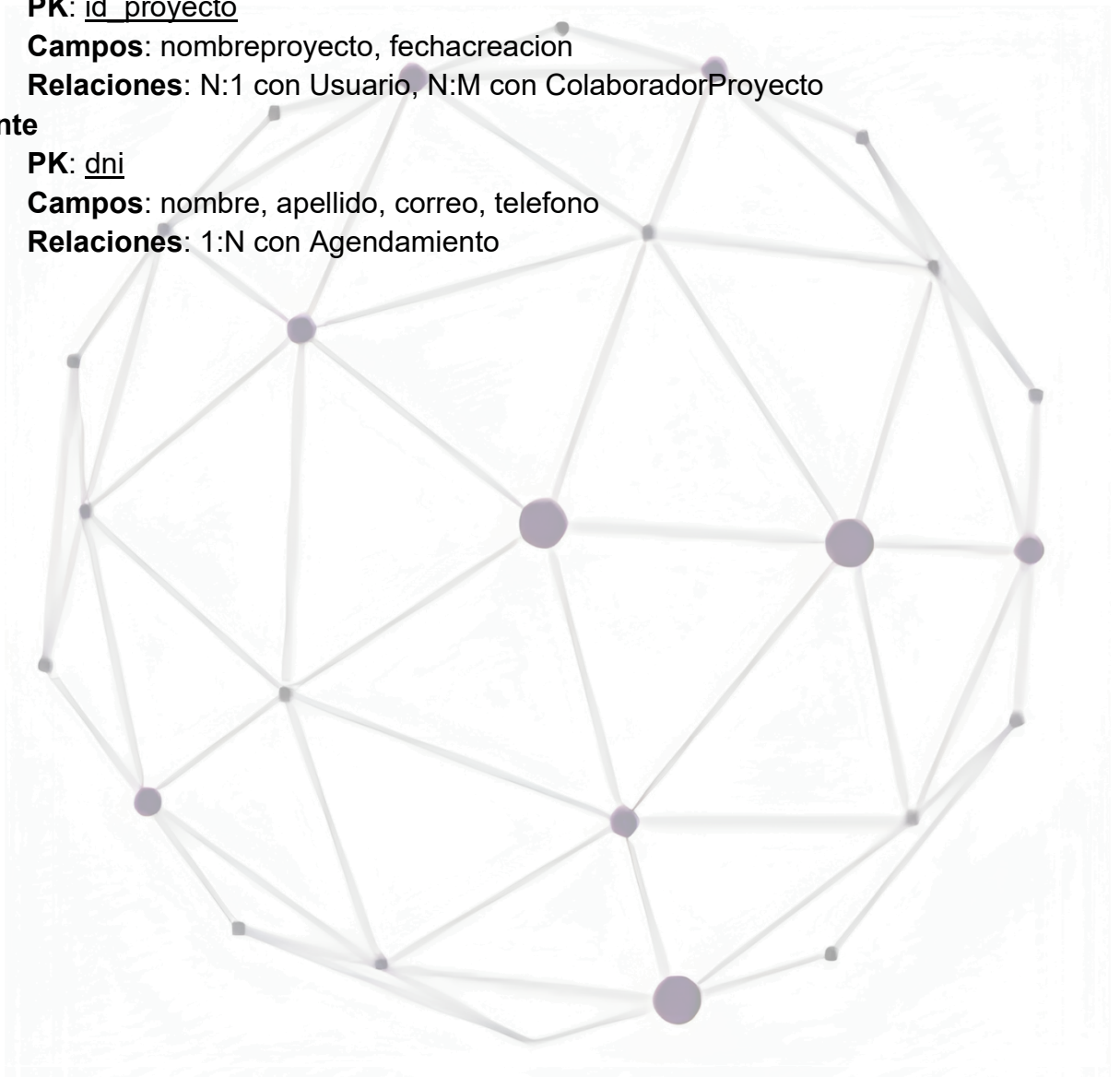
- **PK:** id_usuario
- **Campos:** nombre, apellido, correo, rol_idRol
- **Relaciones:** 1:N con Proyecto, N:M con ColaboradorProyecto

Proyecto

- **PK:** id_proyecto
- **Campos:** nombreproyecto, fechacreacion
- **Relaciones:** N:1 con Usuario, N:M con ColaboradorProyecto

Cliente

- **PK:** dni
- **Campos:** nombre, apellido, correo, telefono
- **Relaciones:** 1:N con Agendamiento



6. APIs y Endpoints

6.1 Endpoints Criticos

- **Gestión de Proyectos**

POST /tasks/api/v1/Proyecto/

GET /tasks/api/v1/proyectos_colaboradores/?id_usuario={id}

GET /tasks/api/v1/info_proyecto_colaboradores/?id_proyecto={id}

- **Gestión de Colaboradores**

POST /tasks/api/v1/invitacionColaborador/

GET /tasks/api/v1/filtro_colaborador/?id_proyecto={id}

GET

/tasks/api/v1/estado_colaborador_proyecto/?id_usuario={id}&id_proyecto={id}

- **Patron de Response**

// Ejemplo de respuesta exitosa

```
{
  "status": "success",
  "data": {
    "nombreproyecto": "Proyecto Demo",
    "fechacreacion": "2025-01-15"
  }
}
```

NOTA: Todos los endpoints requieren autenticación via Supabase Auth headers.

7. AUTENTICACIÓN Y SEGURIDAD

7.1 Flujo de Autenticación

1. **Login** → Supabase Auth
2. **Obtener perfil** → Django API
3. **Validar acceso** → Hook (useProjectAcces)

7.2 Sistema de Roles

```
const ROLES = {  
  ADMIN: 1,    // Puede invitar colaboradores  
  COLABORADOR: 2 // Solo lectura del proyecto asignado  
};
```

7.3 Validación de Acceso

- **Archivo:** useProjectAccess.js
- **Lógica:** Verificar estado “Activo” en ColaboradorProyecto
- **Fallback:** Redireccionar a Error 404

NOTA: Supabase maneja automáticamente la validación de Tokens por medio de sus JWT tokens y refresh tokens

8. PATRONES DE DESARROLLO

8.1 Patron de Gestión de Información

- **Estrategia de Cache:** Zustand con *//persist* middleware
- **Actualizaciones Optimas:** Actualización inmediata + rollback en error
- **Sistema de Eventos:** Custom events para sincronización

8.2 Ejemplo de Patrón de Cache:

// useProjectsStore.js - Patrón Cache

```
fetchProjectInfo: async (projectId) => {  
  const { lastProjectId, projectName } = get();  
  if (lastProjectId === projectId && projectName) {  
    console.log("Proyecto ya en cache, no recargando...");  
    return;  
  }  
  // Cargar desde API...  
}
```

Component Patterns

- **HOC Pattern:** DataTable para tablas reutilizables
- **Compound Components:** Modales con FloatingModal
- **Custom Hooks:** Lógica compartida (useProjectAccess)

9. SOLUCIÓN DE PROBLEMAS

9.1 Problemas Comunes

- **Error: "Colaborator State Changed"**
 - **Causa:** Event listener no registrado
 - **Solución:** Verificar `window.addEventListener('collaboratorStateChanged')`
 - **Archivo:** `SendColaborations.jsx`
- **Error 404 en Dashboard**
 - **Causa:** Usuario sin acceso activo
 - **Solución:** Verificar estado en tabla ColaboradorProyecto en Supabase
 - **Query:** `estado_colaborador_proyecto/?id_usuario={id}&id_proyecto={id}`
- **Cache No Actualiza**
 - **Causa:** Zustand persist mantiene datos obsoletos
 - **Solución:** Implementar `forceRefresh()` o limpiar `localStorage`

9.2 Logs de Debugging

// Habilitar logs en desarrollo

```
console.log("Proyecto cargado:", nombreProyecto);  
console.warn("No se pudo cargar el nombre del proyecto");
```

ATLAS

innovación al alcance de tus manos

ARCHIVOS DE REFERENCIA CLAVE

FUNCIONALIDAD	ARCHIVO PRINCIPAL
Gestión Usuario	stores/useUserStore.js
Cache Proyectos	useProjectsStore.js
Validación Acceso	useProjectAccess.js
Tabla Reutilizable	DataTable.jsx
Email Templates	mensajeColabo.html
Configuración Supabase	supabase/client.js
Layout Principal	components/layout/DashboardLayout.jsx
Configuración Django	backend/settings.py
Views Principales	backend/tasks/views.py

CONTACTO TECNICO

- **Email:** contacto@atlascompany.com
- **Teléfono:** +57 301 243 3965
- **Documentación:** Este manual técnico
- **Código:** Revisar comentarios en archivos fuente

Este manual técnico documenta la arquitectura y funcionamiento actual de ATLAS SOFT v1.0. Para obtener la información más actualizada sobre nuevas funcionalidades o cambios en el sistema, consulte los comentarios en el código fuente o contacte al equipo de desarrollo

Desarrollado con pasión para microempresas de cómputo que buscan digitalizar y optimizar sus procesos de gestión.

© 2025 Atlas Company. Todos los derechos reservados.
