GeoVisor - Plan de Proyecto

📋 Descripción General

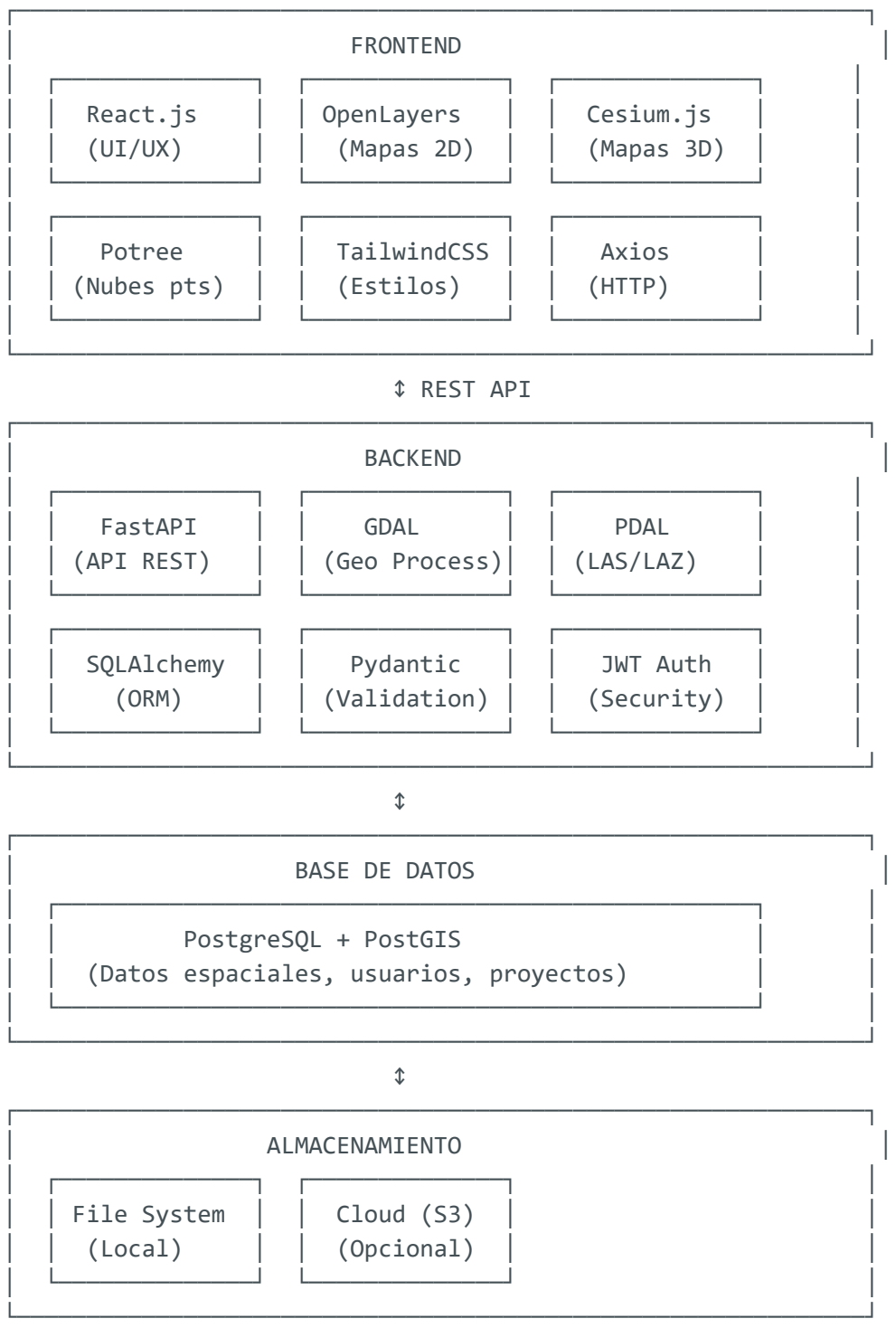
GeoVisor es una plataforma web de visualización y análisis de documentos georeferenciados que permite a los usuarios gestionar proyectos geoespaciales, visualizar diferentes tipos de datos (KML, KMZ, LAS, ortofotos, modelos Revit), realizar mediciones de área y volumen, y colaborar en proyectos con gestión de usuarios y permisos.

🎯 Objetivos del Proyecto

1. Crear un visor geoespacial web interactivo y responsive
2. Soportar múltiples formatos de datos georeferenciados
3. Implementar herramientas de medición (área, distancia, volumen)
4. Gestión de usuarios con roles y permisos
5. Organización por proyectos con datos específicos
6. Visualización 2D y 3D de datos geoespaciales
7. Interfaz intuitiva y moderna

🏗️ Arquitectura del Sistema

Arquitectura General



Modelo de Capas

1. **Capa de Presentación**: React + OpenLayers/Cesium
2. **Capa de Lógica de Negocio**: FastAPI + Python
3. **Capa de Datos**: PostgreSQL + PostGIS
4. **Capa de Almacenamiento**: Sistema de archivos / Cloud Storage

🛠️ Stack Tecnológico

Frontend

Tecnología	Versión	Propósito
React.js	18.x	Framework principal de UI
Vite	5.x	Build tool y dev server
OpenLayers	8.x	Visualización de mapas 2D
Cesium.js	1.11x	Visualización 3D y terrenos
Potree	2.x	Visualización de nubes de puntos
TailwindCSS	3.x	Framework de estilos
Axios	1.x	Cliente HTTP
React Router	6.x	Navegación SPA
Zustand	4.x	Gestión de estado
React Query	5.x	Gestión de datos del servidor

Backend

Tecnología	Versión	Propósito
Python	3.11+	Lenguaje principal
FastAPI	0.10x+	Framework web asíncrono
SQLAlchemy	2.x	ORM para base de datos
Alembic	1.x	Migraciones de BD
Pydantic	2.x	Validación de datos
GDAL	3.8+	Procesamiento de datos geoespaciales
PDAL	2.6+	Procesamiento de nubes de puntos
laspy	2.x	Lectura/escritura de archivos LAS

Tecnología	Versión	Propósito
Shapely	2.x	Operaciones geométricas
GeoAlchemy2	0.14+	Extensión espacial para SQLAlchemy
python-jose	3.x	JWT tokens
passlib	1.7+	Hashing de contraseñas
python-multipart	0.0.6+	Manejo de archivos

Base de Datos

Tecnología	Versión	Propósito
PostgreSQL	15+	Base de datos relacional
PostGIS	3.4+	Extensión espacial

DevOps & Herramientas

Tecnología	Propósito
Docker	Containerización
Docker Compose	Orquestación local
Nginx	Servidor web / Reverse proxy
Git	Control de versiones

📦 Funcionalidades del Sistema

1. Autenticación y Gestión de Usuarios

Descripción: Sistema completo de autenticación con registro, login, recuperación de contraseña y gestión de perfiles.

Funcionalidades:

- ✅ Registro de usuarios con validación de email
- ✅ Login con JWT tokens
- ✅ Recuperación de contraseña
- ✅ Gestión de perfil de usuario
- ✅ Roles y permisos (Admin, Usuario, Viewer)
- ✅ Sesiones seguras

Tecnologías:

- Backend: FastAPI, python-jose (JWT), passlib (bcrypt)
- Frontend: React, Axios, localStorage
- Base de datos: PostgreSQL

Endpoints API:

```
POST   /api/auth/register
POST   /api/auth/login
POST   /api/auth/refresh
POST   /api/auth/forgot-password
PUT    /api/auth/reset-password
GET    /api/users/me
PUT    /api/users/me
```

---

2. Gestión de Proyectos

Descripción: Organización de datos geoespaciales en proyectos con permisos y colaboración.

Funcionalidades:

- ✅ Crear, editar, eliminar proyectos
- ✅ Asignar usuarios a proyectos
- ✅ Definir permisos por proyecto (Owner, Editor, Viewer)
- ✅ Compartir proyectos
- ✅ Organización jerárquica de datos

Tecnologías:

- Backend: FastAPI, SQLAlchemy
- Frontend: React, React Query

- Base de datos: PostgreSQL

Endpoints API:

```
GET    /api/projects
POST   /api/projects
GET    /api/projects/{id}
PUT    /api/projects/{id}
DELETE /api/projects/{id}
POST   /api/projects/{id}/members
DELETE /api/projects/{id}/members/{user_id}
```

---

3. Carga y Gestión de Archivos Georeferenciados

Descripción: Sistema de carga, procesamiento y almacenamiento de diferentes formatos geoespaciales.

Funcionalidades:

- ✅ Carga de archivos KML/KMZ
- ✅ Carga de archivos LAS/LAZ (nubes de puntos)
- ✅ Carga de ortofotos (GeoTIFF, ECW, JPEG2000)
- ✅ Carga de modelos Revit (conversión a IFC/glTF)
- ✅ Validación de archivos
- ✅ Conversión automática a formatos web
- ✅ Generación de metadatos
- ✅ Thumbnails/previews
- ✅ Gestión de capas

Formatos Soportados:

Formato	Tipo	Procesamiento
KML/KMZ	Vector	GDAL → GeoJSON
LAS/LAZ	Nube de puntos	PDAL → Potree
GeoTIFF	Raster	GDAL → Tiles
SHP	Vector	GDAL → GeoJSON

Formato	Tipo	Procesamiento
IFC (Revit)	3D Model	IfcConvert → glTF

Tecnologías:

- Backend: FastAPI, GDAL, PDAL, laspy
- Frontend: React, Dropzone
- Almacenamiento: File system / S3

Endpoints API:

```
POST   /api/projects/{id}/layers/upload
GET    /api/projects/{id}/layers
GET    /api/layers/{id}
DELETE /api/layers/{id}
GET    /api/layers/{id}/download
PUT    /api/layers/{id}/metadata
```

---

4. Visor de Mapas 2D

Descripción: Visualización interactiva de datos geoespaciales en 2D con controles de navegación.

Funcionalidades:

- ✅ Visualización de capas vectoriales (KML, GeoJSON)
- ✅ Visualización de capas raster (ortofotos)
- ✅ Controles de zoom, pan, rotación
- ✅ Capas base (OpenStreetMap, Satellite, Terrain)
- ✅ Control de visibilidad de capas
- ✅ Control de opacidad
- ✅ Estilos personalizados
- ✅ Popup de información
- ✅ Leyenda de capas

Tecnologías:

- Frontend: OpenLayers 8.x, React
- Tiles: OpenStreetMap, Mapbox (opcional)

Componentes:

```
<Map2DViewer>
  <LayerControl />
  <NavigationControls />
  <ScaleBar />
  <CoordinateDisplay />
  <LayerLegend />
</Map2DViewer>
```

---

5. Visor de Mapas 3D

Descripción: Visualización 3D de terrenos, modelos y nubes de puntos.

Funcionalidades:

- ✅ Visualización de terrenos 3D
- ✅ Visualización de nubes de puntos LAS
- ✅ Visualización de modelos 3D (glTF)
- ✅ Controles de cámara 3D
- ✅ Iluminación y sombras
- ✅ Mediciones en 3D
- ✅ Perfiles de elevación

Tecnologías:

- Frontend: Cesium.js, Potree, React
- Procesamiento: PDAL (conversión LAS → 3D Tiles)

Componentes:

```
<Map3DViewer>
  <CesiumViewer />
  <PotreeViewer />
  <Camera3DControls />
  <LightingControls />
</Map3DViewer>
```

---

6. Herramientas de Medición

Descripción: Herramientas para realizar mediciones de distancia, área, perímetro y volumen.

Funcionalidades:

- ✅ Medición de distancias (2D y 3D)
- ✅ Medición de áreas
- ✅ Medición de perímetros
- ✅ Cálculo de volúmenes (corte/relleno)
- ✅ Perfiles de elevación
- ✅ Exportación de resultados
- ✅ Historial de mediciones

Tecnologías:

- Frontend: OpenLayers (2D), Cesium (3D), Turf.js
- Backend: Shapely, GDAL (cálculos complejos)

Tipos de Medición:

Herramienta	Dimensión	Algoritmo
Distancia	2D/3D	Haversine / Euclidiana
Área	2D	Polígono planar
Volumen	3D	Triangulación Delaunay
Perfil	2D/3D	Interpolación lineal

Endpoints API:

```
POST   /api/measurements/area
POST   /api/measurements/volume
POST   /api/measurements/profile
GET    /api/projects/{id}/measurements
```

---

7. Análisis Geoespacial

Descripción: Herramientas de análisis espacial avanzado.

Funcionalidades:

- ✅ Buffer/área de influencia
- ✅ Intersección de capas
- ✅ Unión de geometrías
- ✅ Diferencia de geometrías
- ✅ Análisis de proximidad
- ✅ Estadísticas zonales
- ✅ Generación de curvas de nivel

Tecnologías:

- Backend: Shapely, GDAL, PostGIS
- Frontend: Turf.js (análisis ligeros)

Endpoints API:

```
POST   /api/analysis/buffer
POST   /api/analysis/intersect
POST   /api/analysis/union
POST   /api/analysis/difference
POST   /api/analysis/contours
```

---

8. Exportación de Datos

Descripción: Exportación de capas, mediciones y análisis en diferentes formatos.

Funcionalidades:

- ✅ Exportar a KML/KMZ
- ✅ Exportar a GeoJSON
- ✅ Exportar a Shapefile
- ✅ Exportar a PDF (mapas)
- ✅ Exportar mediciones a CSV/Excel
- ✅ Exportar capturas de pantalla

Tecnologías:

- Backend: GDAL, ReportLab (PDF), openpyxl
- Frontend: html2canvas (screenshots)

Endpoints API:

```
GET    /api/layers/{id}/export?format=kml
GET    /api/projects/{id}/export?format=pdf
GET    /api/measurements/{id}/export?format=csv
```

---

9. Panel de Administración

Descripción: Panel para administradores del sistema.

Funcionalidades:

- ✅ Gestión de usuarios
- ✅ Gestión de proyectos globales
- ✅ Monitoreo de uso de almacenamiento
- ✅ Logs de actividad
- ✅ Configuración del sistema
- ✅ Estadísticas de uso

Tecnologías:

- Frontend: React, Recharts (gráficos)
- Backend: FastAPI, SQLAlchemy

Endpoints API:

```
GET    /api/admin/users
GET    /api/admin/projects
GET    /api/admin/stats
GET    /api/admin/logs
PUT    /api/admin/settings
```

---

10. Notificaciones y Actividad

Descripción: Sistema de notificaciones para colaboración.

Funcionalidades:

- ✅ Notificaciones de invitaciones a proyectos
- ✅ Notificaciones de cambios en capas
- ✅ Registro de actividad del proyecto
- ✅ Comentarios en capas

Tecnologías:

- Backend: FastAPI, WebSockets (opcional)
- Frontend: React, React Toastify

📁 Estructura del Proyecto

```
geovisor/
├── backend/
│   ├── app/
│   │   ├── __init__.py
│   │   ├── main.py
│   │   ├── config.py
│   │   ├── database.py
│   │   ├── models/
│   │   │   ├── __init__.py
│   │   │   ├── user.py
│   │   │   ├── project.py
│   │   │   ├── layer.py
│   │   │   └── measurement.py
│   │   ├── schemas/
│   │   │   ├── __init__.py
│   │   │   ├── user.py
│   │   │   ├── project.py
│   │   │   └── layer.py
│   │   ├── api/
│   │   │   ├── __init__.py
│   │   │   ├── deps.py
│   │   │   ├── auth.py
│   │   │   ├── users.py
│   │   │   ├── projects.py
│   │   │   ├── layers.py
│   │   │   ├── measurements.py
│   │   │   └── analysis.py
│   │   ├── core/
│   │   │   ├── __init__.py
│   │   │   ├── security.py
│   │   │   └── config.py
│   │   ├── services/
│   │   │   ├── __init__.py
│   │   │   ├── gdal_service.py
│   │   │   ├── pdal_service.py
│   │   │   ├── file_service.py
```

```
│   │   │   └── measurement_service.py
│   │   └── utils/
│   │       ├── __init__.py
│   │       └── helpers.py
│   ├── alembic/
│   ├── tests/
│   ├── requirements.txt
│   ├── Dockerfile
│   └── .env.example
├── frontend/
│   ├── public/
│   ├── src/
│   │   ├── components/
│   │   │   ├── auth/
│   │   │   ├── layout/
│   │   │   ├── map/
│   │   │   │   ├── Map2D.jsx
│   │   │   │   ├── Map3D.jsx
│   │   │   │   ├── LayerControl.jsx
│   │   │   │   └── MeasurementTools.jsx
│   │   │   ├── projects/
│   │   │   └── common/
│   │   ├── pages/
│   │   │   ├── Login.jsx
│   │   │   ├── Dashboard.jsx
│   │   │   ├── ProjectView.jsx
│   │   │   └── Admin.jsx
│   │   ├── hooks/
│   │   ├── services/
│   │   │   └── api.js
│   │   ├── store/
│   │   ├── utils/
│   │   ├── App.jsx
│   │   └── main.jsx
│   ├── package.json
│   ├── vite.config.js
│   ├── tailwind.config.js
│   └── Dockerfile
├── docker-compose.yml
├── .gitignore
├── README.md
└── PROJECT_PLAN.md
```

📅 Roadmap de Desarrollo

Fase 1: Fundamentos (Semanas 1-2)

- ☐ Configuración del entorno de desarrollo
- ☐ Estructura de proyecto backend y frontend

- ☐ Configuración de base de datos PostgreSQL + PostGIS
- ☐ Sistema de autenticación básico (registro, login)
- ☐ Gestión básica de usuarios

Fase 2: Gestión de Proyectos (Semanas 3-4)

- ☐ CRUD de proyectos
- ☐ Sistema de permisos y roles
- ☐ Asignación de usuarios a proyectos
- ☐ Dashboard de proyectos

Fase 3: Carga de Archivos (Semanas 5-6)

- ☐ Sistema de carga de archivos
- ☐ Procesamiento de KML/KMZ
- ☐ Procesamiento de archivos LAS/LAZ
- ☐ Procesamiento de ortofotos (GeoTIFF)
- ☐ Generación de metadatos

Fase 4: Visor 2D (Semanas 7-8)

- ☐ Integración de OpenLayers
- ☐ Visualización de capas vectoriales
- ☐ Visualización de capas raster
- ☐ Controles de navegación
- ☐ Control de capas
- ☐ Estilos y leyendas

Fase 5: Herramientas de Medición 2D (Semana 9)

- ☐ Medición de distancias
- ☐ Medición de áreas
- ☐ Medición de perímetros
- ☐ Exportación de mediciones

Fase 6: Visor 3D (Semanas 10-11)

- ☐ Integración de Cesium.js
- ☐ Visualización de terrenos 3D
- ☐ Integración de Potree para nubes de puntos
- ☐ Visualización de modelos 3D
- ☐ Controles de cámara 3D

Fase 7: Mediciones 3D y Volúmenes (Semana 12)

- ☐ Mediciones en 3D
- ☐ Cálculo de volúmenes
- ☐ Perfiles de elevación
- ☐ Análisis de corte/relleno

Fase 8: Análisis Geoespacial (Semana 13)

- ☐ Herramientas de buffer
- ☐ Intersección de capas
- ☐ Análisis de proximidad
- ☐ Generación de curvas de nivel

Fase 9: Exportación y Reportes (Semana 14)

- ☐ Exportación a diferentes formatos
- ☐ Generación de PDFs
- ☐ Exportación de mediciones
- ☐ Capturas de pantalla

Fase 10: Panel de Administración (Semana 15)

- ☐ Dashboard de administración

- ☐ Gestión de usuarios avanzada
- ☐ Monitoreo de sistema
- ☐ Logs y auditoría

Fase 11: Optimización y Testing (Semana 16)

- ☐ Testing unitario
- ☐ Testing de integración
- ☐ Optimización de rendimiento
- ☐ Optimización de carga de archivos grandes

Fase 12: Deployment y Documentación (Semana 17)

- ☐ Configuración de Docker
- ☐ Deployment en servidor
- ☐ Documentación de API
- ☐ Documentación de usuario
- ☐ Video tutoriales

🔒 Consideraciones de Seguridad

1. **Autenticación**: JWT tokens con refresh tokens
2. **Autorización**: RBAC (Role-Based Access Control)
3. **Validación**: Validación de archivos subidos
4. **Sanitización**: Prevención de inyección SQL y XSS
5. **HTTPS**: Comunicación encriptada
6. **Rate Limiting**: Prevención de abuso de API
7. **CORS**: Configuración adecuada de CORS

📊 Métricas de Éxito

1. **Rendimiento**:

- Carga de mapa < 2 segundos
- Procesamiento de archivos KML < 5 segundos
- Procesamiento de archivos LAS < 30 segundos (dependiendo del tamaño)

2. **Usabilidad**:
   - Interfaz intuitiva
   - Tiempo de aprendizaje < 30 minutos
3. **Escalabilidad**:
   - Soporte para 100+ usuarios concurrentes
   - Almacenamiento de proyectos ilimitados

📚 Recursos y Referencias

Documentación Oficial

- OpenLayers
- Cesium.js
- Potree
- FastAPI
- PostGIS
- GDAL
- PDAL

Tutoriales y Guías

- OpenLayers Examples
- Cesium Sandcastle
- FastAPI Tutorial

🤝 Contribución

Este proyecto está en desarrollo activo. Las contribuciones son bienvenidas siguiendo las mejores prácticas de desarrollo.

📝 Notas Adicionales

Procesamiento de Archivos Revit

- Los archivos Revit (.rvt) requieren conversión a IFC primero
- Se puede usar IfcConvert (parte de IfcOpenShell) para convertir IFC a glTF
- Alternativa: Exportar desde Revit a IFC directamente

Optimización de Nubes de Puntos

- Archivos LAS grandes (>100MB) deben convertirse a formato Potree
- Usar PDAL para decimación y filtrado
- Implementar LOD (Level of Detail) para mejor rendimiento

Almacenamiento

- Archivos pequeños (<10MB): Base de datos
- Archivos medianos (10MB-1GB): Sistema de archivos local
- Archivos grandes (>1GB): Cloud storage (S3, Google Cloud Storage)

Fecha de creación: 2025-12-16
Versión: 1.0
Autor: GeoVisor Team