

Propuesta de Arquitectura para un Gemelo Digital de Paneles Solares

Resumen

Este documento presenta una propuesta de arquitectura para un gemelo digital aplicado a instalaciones fotovoltaicas. El sistema se orienta a estimar la generación energética actual y futura, recomendar mantenimiento preventivo, modelar el desgaste de los paneles y predecir el estado de carga de baterías. La solución integra una capa de presentación con React enfocada en tableros administrativos, una capa de lógica y orquestación con FastAPI y GraphQL, una capa de datos unificada en MongoDB (colecciones documentales y de series temporales), y una capa de inteligencia basada en modelos de aprendizaje automático. La fuente de datos meteorológicos se apoya en Open-Meteo por su apertura, variedad de variables relevantes para fotovoltaica y disponibilidad de histórico. El resultado es una arquitectura modular, escalable y lista para operar sin telemetría en tiempo real desde hardware, centrando el valor en la predicción.

Un gemelo digital es una representación virtual dinámica de un sistema físico que permite analizar su estado, predecir su comportamiento y retroalimentar su operación con base en datos. En el ámbito fotovoltaico, un gemelo digital permite anticipar fallos, optimizar la operación y mejorar la toma de decisiones. En el presente proyecto, se modela un gemelo digital para paneles solares con un enfoque de análisis y visualización; el sistema no se conecta al hardware físico y, por tanto, no dispone de medidas en tiempo real de generación. La generación 'actual' mostrada por el sistema es una estimación producida por modelos (nowcasting), y las predicciones de mediano plazo emplean pronósticos meteorológicos.

Objetivos específicos: (i) estimar la generación actual y futura, (ii) predecir necesidades de mantenimiento, (iii) estimar el desgaste de paneles, y (iv) predecir el estado de carga de baterías. Se contemplan dos roles de usuario: Administrador y Operador/Usuario.

2. Justificación de tecnologías y decisiones de diseño

La Tabla 1 sintetiza el rol, la justificación específica para este proyecto y las principales mitigaciones de riesgo.

Tecnología	Rol en la arquitectura	Justificación específica para este proyecto	Riesgos y mitigaciones
React (sin Next.js)	Capa de presentación para dashboards administrativos.	Los tableros administrativos priorizan interactividad, estados complejos en cliente y sesiones prolongadas; no exigen SEO ni renderizado del lado del servidor. React ofrece un modelo de componentes maduro y ecosistema amplio de bibliotecas de visualización y formularios, reduciendo complejidad y costo operativo.	Riesgo: sobrecarga de estado en cliente. Mitigación: patrones de estado (Context/Redux/Zustand) y code-splitting.
FastAPI (Python)	API y orquestación de negocio.	Rendimiento alto, validación estricta de datos, tipado y documentación automática. Facilita integrar pipelines de ML en Python y soporta concurrencia para ingesta meteorológica y predicción.	Riesgo: endpoints complejos. Mitigación: diseño por capas, pruebas automáticas y observabilidad.

GraphQL	Fachada de datos para consultas compuestas.	Evita sobre/infra-obtención de datos en pantallas complejas (un dashboard pide exactamente lo necesario). Esquema tipado y posibilidad de suscripciones para actualizaciones casi en tiempo real.	Riesgo: resolvers costosos. Mitigación: batching, caching y límites de profundidad.
MongoDB (documental + series temporales)	Persistencia unificada: catálogo + series temporales + predicciones.	Esquema flexible para metadatos de paneles y baterías, colecciones de series temporales con retención (TTL) y compresión, y Change Streams para notificaciones en vivo. Evita fragmentar el dato entre dos motores distintos.	Riesgo: consultas no optimizadas. Mitigación: índices compuestos, diseño de meta-campos y sharding por planta/tiempo.
Open-Meteo	Proveedor abierto de meteorología y radiación.	API abierta, sin clave por defecto, amplio conjunto de variables relevantes (GHI, DNI, DHI, GTI, temperatura, viento, nubes, precipitación) y acceso a histórico para entrenamiento y backtesting.	Riesgo: variabilidad regional. Mitigación: calibración por planta y validación cruzada.
Docker	Contenedorización por servicio.	Portabilidad y despliegue reproducible en entornos locales o nube, facilitando CI/CD y escalamiento horizontal.	Riesgo: sobrecosto en orquestación. Mitigación: automatización con compose/k8s y políticas de recursos.

3. Arquitectura propuesta

La arquitectura se organiza por capas de responsabilidad con despliegue por servicios independientes. Se busca cohesión interna, bajo acoplamiento y escalabilidad horizontal.

3.1 Capa de presentación (React)

Aplicación React orientada a dashboards administrativos, con control de estado global, ruteo en cliente y componentes de visualización. Se privilegia la rapidez en la interacción, la modularidad y la reutilización de widgets (tarjetas, tablas, series temporales, mapas). Al ser un sistema cerrado y de intranet/empresa, el SEO no es relevante.

3.2 Capa de negocio y comunicación (FastAPI + GraphQL)

FastAPI centraliza autenticación, autorización, orquestación de ingesta y ejecución de modelos. GraphQL actúa como fachada que compone datos de catálogo, clima persistente y salidas de predicción en una sola consulta por pantalla. Se habilitan suscripciones para notificaciones de nuevas predicciones o alertas.

3.3 Capa de datos (MongoDB)

Un único clúster de MongoDB aloja: (i) colecciones documentales de catálogo (usuarios, plantas, paneles, baterías, mantenimientos) y (ii) colecciones de series temporales para clima y predicciones (generación, mantenimiento, desgaste, estado de carga). Las series temporales emplean políticas de retención (TTL) y compresión, y se indexan por planta/panel y marca temporal.

3.4 Capa de inteligencia (Modelos de predicción)

Modelos de nowcasting y forecast entrenados con histórico meteorológico y parámetros de fabricante. Las salidas se almacenan como series temporales para facilitar agregaciones por ventana y visualización inmediata.

3.5 Infraestructura y despliegue

Cada componente se empaqueta en contenedores Docker. Se prevé un entorno con réplica de MongoDB para alta disponibilidad y, cuando sea necesario, sharding por planta y tiempo. La observabilidad abarcará métricas de API, latencias de consulta, colas de tareas y salud de modelos.

4. Modelo de datos en MongoDB (descripción textual)

El diseño sigue una desnormalización controlada para acelerar lecturas en dashboards. Las referencias entre colecciones se realizan por identificadores y, cuando aporta valor, se duplican campos de uso frecuente (p. ej., nombre de planta).

Tabla 2 detalla las colecciones principales, su tipo y consideraciones de diseño.

Colección	Tipo	Campos clave (texto)	Índices sugeridos	Retención	Notas
usuarios	Catálogo	nombre, email (único), hash_password, rol (ADMIN/ OPERADOR), estado, createdAt	email (único), rol	N/A	Control de acceso por rol; auditoría de actividad recomendada.
plantas	Catálogo	nombre, ubicación (lat/lon), zona horaria, capacidad_kWp, descripción, tags	índice geoespacial por ubicación; nombre	N/A	Útil para consultas geográficas y asignación de clima.
paneles	Catálogo	panelId, fabricante, modelo, potencia_instalada_Wp, orientación, inclinación, área, eficiencia, fecha_instalación, estado	panelId, estado; índice compuesto panelId+estado	N/A	Metadatos de fabricante y parámetros físicos para modelado.
baterías	Catálogo	panelId, fabricante, modelo, capacidad_kWh, potencia_nominal_kW, eficiencia, DoD, ciclos_estimados, fecha_instalación, estado	panelId, estado	N/A	Capacidad y límites operativos para predicción de SoC.

mantenimientos	Catálogo operacional	panelId/ bateriaId, fecha_programada, tipo, descripción, estado, origen, score_prioridad	panelId, fecha_programada, estado	N/A	Generados por modelo o manualmente; ciclo de vida gestionado en UI.
clima Mediciones	Series temporales	ts (marca temporal), meta (plantaId, lat, lon, fuente, resolución), GHI, DNI, DHI, GTI, temp_2m, humedad, nubes, viento, precipitación	meta.plantaId + ts	Recomendado: 24 meses	Ingesta periódica desde Open-Meteo.
predicciones_generacion	Series temporales	ts, meta (panelId/ plantaId, horizonte_min, modelo, versión), valor (kW/kWh), intervalo_confianza	meta.panelId / meta.plantaId + meta.horizonte_min + ts	Recomendado: 36 meses	Nowcasting y forecast de generación por panel o planta.
predicciones_mantenimiento	Series temporales	ts, meta (panelId, tipo), score, explicación, horizonte_dias	meta.panelId + ts	Según política (12-24 meses)	Se traduce a tareas programables en 'mantenimientos'.
predicciones_desgaste_panel	Series temporales	ts, meta (panelId), degradación_rel, tasa_anual_est, drivers	meta.panelId + ts	Largo plazo (≥36 meses)	Seguimiento de envejecimiento y salud del módulo.

predicciones_soc_bateria	Series temporales	ts, meta (bateriaId, horizonte_min), soc_pred, p_carga_kW, p_descarga_kW	meta.bateriaId + ts	36 meses	Planificación de operación de almacenamiento.
--------------------------	-------------------	--	---------------------	----------	---

Estrategia de indexación: índices compuestos centrados en los meta-campos (plantaId, panelId, bateriaId) y la marca temporal. Para grandes volúmenes, sharding por planta y tiempo. Se aplican políticas de retención (TTL) en colecciones de ingesta y predicción.

5. Ingesta meteorológica con Open-Meteo

Open-Meteo es una API abierta que provee variables clave para fotovoltaica, incluyendo radiación (GHI, DNI, DHI, GTI), temperatura a 2 m, humedad relativa, cobertura nubosa, viento y precipitación. Permite consultar datos actuales, series horarias y, cuando aplica, resolución de 15 minutos, además de pronósticos con horizontes de varios días y acceso a histórico. Estas variables alimentan directamente los modelos de generación y mantenimiento.

Tabla 3 resume las variables meteorológicas utilizadas y su aporte al modelo.

Variable	Utilidad en el proyecto	Frecuencia típica
GHI / DNI / DHI / GTI	Cálculo de potencia FV e irradiancia sobre plano del módulo.	15 min / 1 hora / diario
Temperatura (2 m)	Rendimiento térmico y envejecimiento.	1 hora / diario
Cobertura nubosa	Atenuación y variabilidad intra-día.	1 hora
Viento (vel./dir.)	Refrigeración y riesgo de ensuciamiento (arena/polvo).	1 hora
Humedad relativa	Efectos atmosféricos y soiling.	1 hora
Precipitación	Limpieza natural de módulos, señal para mantenimiento.	1 hora / diario

Estrategia de orquestación: tareas programadas consultan Open-Meteo por planta en función de sus coordenadas; los datos se persisten como series temporales y, a continuación,

se ejecutan los modelos para producir nowcasting y forecast. Se recomiendan políticas de idempotencia (claves por ts+planta) y backfilling para entrenamiento.

6. Casos de uso fundamentales

Los casos de uso se alinean con los objetivos del proyecto y se apoyan en el catálogo, la serie meteorológica y las predicciones.

Tabla 4: resumen de entradas, salidas y métricas.

Caso de uso	Entradas	Salidas	Modelo / Lógica
Generación actual y futura	Clima actual/ pronóstico, metadatos de panel/ planta	Curvas de potencia/ energía con intervalos de confianza	Nowcasting/forecast supervisado con features meteorológicas
Mantenimiento	Series de clima, historial de predicción, tareas previas	Sugerencias con tipo, fecha y prioridad	Detección de soiling y reglas de umbral con señales meteorológicas
Desgaste de paneles	Edad y características, clima acumulado	Trayectoria de degradación y tasa anual	Modelos de tendencia con factores térmicos/ UV
Estado de carga de baterías (SoC)	Generación estimada, características de batería	Trayectoria de SoC e indicaciones de carga/descarga	Balance energético con restricciones (DoD, eficiencia)

7. Seguridad y cumplimiento

Autenticación mediante JWT o sesiones, autorización por rol (Administrador/Operador), cifrado de extremo a extremo (HTTPS). Validación de datos en la API, políticas de contraseñas fuertes y hash seguro. En persistencia NoSQL, se controlan operadores en filtros de usuario, uso de proyecciones y pipelines validados. MongoDB con mínimos privilegios, auditoría y copias de seguridad.

8. Despliegue, operación y rendimiento

Despliegue por contenedores; integración y entrega continua. Para series temporales, políticas de retención (p. ej., 24–36 meses) y compresión; índices compuestos por meta-campos y tiempo. Sharding por planta y ventana temporal a alto volumen. Observabilidad: métricas de API, colas, latencias de lectura, uso de CPU/memoria y salud de modelos.

9. Conclusiones

La arquitectura propuesta integra React, FastAPI con GraphQL, MongoDB y Open-Meteo en un diseño coherente para un gemelo digital sin telegestión de hardware. La unificación de catálogo y series temporales en MongoDB reduce complejidad operativa, mientras que la fachada GraphQL simplifica la obtención de datos compuestos para dashboards.

Open-Meteo aporta la base meteorológica necesaria para habilitar los casos de uso de generación, mantenimiento, desgaste y carga de baterías. El resultado es una solución modular, escalable y portable, adecuada para distintas regiones y escenarios de despliegue.