🎯 Phase 1: High-Level Concept

🔧 Problem

Organizations managing physical infrastructure (HVAC, utilities, public works) struggle with:

Incomplete visibility into asset health and usage

High maintenance costs due to reactive workflows

Siloed data between IoT, operations, and planning teams

🚀 Solution

Build a Digital Twin platform that provides:

Real-time asset monitoring via IoT telemetry

Predictive maintenance using ML models

Interactive 3D visualization of assets and systems

AI-driven recommendations to optimize asset performance and lifecycle

🧱 Phase 2: Architecture Overview

css

Copy

Edit

[Edge Devices/Sensors] --> [MQTT Broker / AWS IoT] --> [Data Pipeline (Kafka/Flink)]

↘ ↘

[Edge Inference] [Cloud ML Model Inference]

↘

[Digital Twin Engine] <-- [Asset DB]

↘

[3D Visualization (WebGL)] <-- [API Gateway / GraphQL / REST] --> [User Portal]

🔨 Phase 3: Tech Stack

Layer Technology

IoT Edge ESP32/Raspberry Pi, MQTT, AWS IoT Core

Data Pipeline Apache Kafka or AWS Kinesis

ML/AI PyTorch, TensorFlow, Scikit-learn, AWS SageMaker

Backend API Python (FastAPI), Node.js (alt), MQTT bridge

DB PostgreSQL + TimescaleDB (time series), Redis

3D Visuals Three.js or Unity WebGL

Frontend React + TailwindCSS or Next.js

DevOps Docker, Kubernetes, ArgoCD, Terraform

Monitoring Prometheus + Grafana

Auth Keycloak or Auth0

💡 Phase 4: Feature Set

MVP Features

🔌 IoT data ingestion via MQTT

🔍 Real-time dashboard showing asset metrics (temp, vibration, voltage)

🧠 Predictive failure detection using ML model

🗺️ 3D visualization of assets with alerts (Three.js)

📤 Notification engine (SMS, Email, Slack webhook)

Stretch Goals

🤖 Edge AI models for low-latency predictions

🧩 Digital twin library for HVAC types, valves, sensors, etc.

🗃️ CMMS (Computerized Maintenance Management System) integration

🧠 Agentic AI planner: "Optimize load balancing for HVAC during peak hours"

📊 Business intelligence dashboard (Power BI or Plotly Dash)

📋 Phase 5: User Personas & Stories

👷 Field Technician

As a technician, I want to receive real-time alerts and diagnostics from the system so I can respond to issues before failure occurs.

🧑‍💼 Ops Manager

As an operations manager, I want to visualize asset performance and schedule predictive maintenance to reduce downtime.

🧠 AI/Data Engineer

As an engineer, I want to deploy updated ML models and track telemetry pipelines to ensure data accuracy and continuous learning.

🛠️ Phase 6: Next Steps (Suggested Sprint Plan)

Sprint 1:

Define asset models (schema, DB, 3D)

Build FastAPI backend

Set up MQTT broker and ingest telemetry

Mock sensor simulator in Python or Node

Sprint 2:

Implement dashboard (React or Streamlit)

Integrate PostgreSQL + TimescaleDB

Train simple ML model (e.g., linear regression or XGBoost)

Sprint 3:

Integrate 3D digital twin visualizer with real-time updates

Set up alerting system (email/SMS)

Dockerize everything

Sprint 4:

Add role-based login with Keycloak

Deploy to Kubernetes cluster (EKS or Minikube)

Set up Grafana monitoring

Sprint 5+:

Implement edge inference

Add AI agent interface ("What’s the most critical system today?")

Start working toward a public demo version

📁 GitHub Repo Structure (Preview)

cpp

Copy

Edit

digital-twin-platform/

├── backend/

│ ├── app.py

│ ├── mqtt\_client.py

│ ├── models/

│ └── db/

├── frontend/

│ ├── public/

│ └── src/

│ ├── components/

│ └── pages/

├── digital\_twin/

│ ├── asset\_models/

│ └── visualizer/

├── ml/

│ ├── models/

│ ├── training/

│ └── inference/

├── infra/

│ ├── docker/

│ └── terraform/

└── README.md