# System Design Document

## 1. System Overview

The Digital Twin for Disaster Response system consists of modular Python components for data retrieval, processing, conversion, and visualization. It supports climate and geospatial data integration for flood mapping.

## 2. Architecture

The architecture follows a layered design:  
1. Data Layer: ERA5, DEM, and OSM datasets  
2. Processing Layer: Xarray, Dask, and Rasterio for computation and reprojection  
3. Storage Layer: Cloud Optimized GeoTIFFs (COG) for efficient tile serving  
4. API Layer: FastAPI-based microservice serving raster and vector data  
5. Visualization Layer: Folium client rendering precipitation and infrastructure maps

## 3. Module Design

• data\_pipeline.py: Handles data ingestion (ERA5, DEM, OSM) and processing  
• serve\_api.py: Implements FastAPI endpoints for raster tiles and GeoJSON  
• flood\_accum\_cog.tif, dem.tif, era5\_flood.nc: Example processed datasets  
• Folium Client: Displays layers interactively for visualization and time analysis

## 4. Data Flow

1. The system downloads ERA5 NetCDF and reads it with Xarray.  
2. Data is processed into daily/hourly precipitation layers.  
3. DEM and OSM layers are aligned and reprojected.  
4. Processed rasters are saved as Cloud Optimized GeoTIFFs.  
5. FastAPI serves tiles from COGs via titiler.core.  
6. Folium/Jupyter visualizes the map and time-series precipitation.

## 5. API Design

Endpoints:  
• /tiles/precip/{z}/{x}/{y} - Returns precipitation raster tiles  
• /osm/buildings - Returns GeoJSON of buildings  
• /osm/roads - Returns GeoJSON of roads  
• /cog/info - Metadata about raster layers

## 6. Tools & Technologies

• Programming: Python 3.11+  
• Libraries: Xarray, Rasterio, Rio-Cogeo, OSMnx, Folium, FastAPI, Titiler  
• Data Formats: NetCDF, GeoTIFF, Cloud Optimized GeoTIFF (COG)  
• Visualization: Folium, Leaflet Time Slider Plugin