import numpy as np

import matplotlib.pyplot as plt

from scipy.ndimage import gaussian\_filter

from sklearn.ensemble import RandomForestClassifier

import pickle

import random

# 1. Terrain & Rainfall Data Loader (Simulated)

def load\_terrain\_data():

elevation = np.random.rand(100, 100) \* 100 # Simulated elevation map (DEM)

return elevation

def load\_rainfall\_data():

rainfall = np.random.rand(100, 100) \* 10 # Simulated rainfall in mm

return rainfall

# 2. Drainage Flow Simulation

def simulate\_water\_flow(elevation, rainfall\_intensity):

smoothed = gaussian\_filter(elevation, sigma=1)

water\_accum = np.maximum(rainfall\_intensity - smoothed \* 0.01, 0)

return water\_accum

def visualize\_water\_flow(water\_accum, filename="water\_flow\_simulation.png"):

plt.figure(figsize=(6, 5))

plt.imshow(water\_accum, cmap='Blues')

plt.title("Simulated Water Accumulation")

plt.colorbar(label="Water Depth (cm)")

plt.tight\_layout()

plt.savefig(filename)

plt.close()

print(f"Water flow visualization saved as {filename}")

# 3. AI-Based Flood Risk Predictor

def train\_flood\_model():

features = np.random.rand(100, 5)

labels = np.random.choice(["Low", "High"], 100)

model = RandomForestClassifier(n\_estimators=100)

model.fit(features, labels)

with open("flood\_model.pkl", "wb") as f:

pickle.dump(model, f)

return model

def predict\_flood\_risk(model):

test\_features = np.random.rand(1, 5)

prediction = model.predict(test\_features)

return prediction[0]

# 4. Mock IoT Integration

def get\_sensor\_data():

return {

"Drain\_Location\_1": round(random.uniform(0.1, 0.5), 2),

"Drain\_Location\_2": round(random.uniform(0.2, 0.8), 2),

"Rainfall\_mm": round(random.uniform(5, 50), 2)

}

# 5. Dashboard

def dashboard(sensor\_data, risk):

print("\n--- Urban Drainage Dashboard ---")

for loc, val in sensor\_data.items():

print(f"{loc}: {val} units")

print(f"Predicted Flood Risk: {risk}")

if risk == "High":

print("ALERT: High flood risk detected!")

# Run the full system

def run\_drainage\_system():

elevation = load\_terrain\_data()

rainfall = load\_rainfall\_data()

water = simulate\_water\_flow(elevation, rainfall)

visualize\_water\_flow(water)

model = train\_flood\_model()

risk\_prediction = predict\_flood\_risk(model)

sensors = get\_sensor\_data()

dashboard(sensors, risk\_prediction)

# Run the program

if \_\_name\_\_ == "\_\_main\_\_":

run\_drainage\_system()