```
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()
```