DEVELOPMENT PART -1 SMART WATER SYSTEM

Developing a Python codebase for a smart water system involves integrating various components and functionalities. Below is a simplified framework for a smart water system:

python

# Import necessary libraries Import time

# Define sensor and actuator pins Water\_sensor\_pin = 14

Pump\_relay\_pin = 4

# Initialize sensor and actuator

# (You need to set up the actual hardware components accordingly) # Example for Raspberry Pi and RPi.GPIO library

Import RPi.GPIO as GPIO GPIO.setmode(GPIO.BCM)

GPIO.setup(water\_sensor\_pin, GPIO.IN) GPIO.setup(pump\_relay\_pin, GPIO.OUT)

# Main loop for system operation Try:

While True:

# Read sensor data

Water\_level = GPIO.input(water\_sensor\_pin)

# Implement control logic

If water\_level == 0: # Low water level detected

GPIO.output(pump\_relay\_pin, GPIO.HIGH) # Turn on the pump Print(“Water level is low. Pump is turned on.”)

Else:

GPIO.output(pump\_relay\_pin, GPIO.LOW) # Turn off the pump Print(“Water level is sufficient. Pump is turned off.”)

# Add appropriate delays or use interrupts as needed Time.sleep(5)

Except KeyboardInterrupt:

GPIO.cleanup()

```

This is a simple example that simulates a smart water system. In practice, you’ll need to integrate real sensors and actuators, implement more sophisticated control logic, and potentially connect the system to a user interface for monitoring and control.

Ensure to replace the GPIO pin numbers, setup code, and control logic with the appropriate configurations for your hardware. Additionally, consider adding error handling, data logging, and other necessary functionalities as per the specific requirements of your smart water system.