

# SMART WATER MANAGEMENT

Smart water management systems are crucial for conserving this precious resource. To create a Python code for smart water management, you can use various sensors (like flow sensors) and actuators (like valves) connected to a microcontroller (such as Raspberry Pi or Arduino). Here's a basic outline of how you might structure your Python code:

```
import time
```

```
import RPi.GPIO as GPIO # If you're using Raspberry Pi GPIO pins
```

Initialize Sensors and Actuators:

```
python
```

```
# Initialize GPIO pins for sensors and actuators
```

```
GPIO.setmode(GPIO.BOARD)
```

```
flow_sensor_pin = 17 # Example pin number for the flow sensor
```

```
valve_pin = 18 # Example pin number for the water valve
```

```
GPIO.setup(flow_sensor_pin, GPIO.IN, pull_up_down=GPIO.PUD_UP)
```

```
GPIO.setup(valve_pin, GPIO.OUT)
```

Read Sensor Data (Flow Sensor):

```
def read_flow_sensor():
```

```
    # Code to read data from the flow sensor
```

```
    # Return flow rate or water consumption data
```

```
    pass
```

Control Water Flow (Actuator - Valve):

```
def control_valve(is_open):
```

```
    if is_open:
```

```
GPIO.output(valve_pin, GPIO.HIGH) # Open the valve
```

```
else:
```

```
GPIO.output(valve_pin, GPIO.LOW) # Close the valve
```

Main Program Loop:

```
def main():
```

```
    try:
```

```
        while True:
```

```
            # Read data from the flow sensor
```

```
            flow_rate = read_flow_sensor()
```

```
            # Implement smart logic based on flow rate data
```

```
            # For example, if flow rate > threshold, close the valve
```

```
            if flow_rate > threshold:
```

```
                control_valve (False) # Close the valve
```

```
            else:
```

```
                control_valve (True) # Open the valve
```

```
            time.sleep(1) # Delay to control the frequency of sensor readings
```

```
except KeyboardInterrupt:
```

```
    GPIO.cleanup() # Cleanup GPIO pins on keyboard interrupt
```

```
if __name__ == "__main__":
```

```
    main()
```