DETAILED PROJECT OUTLINE — IoT SMART TREE WASHING / IRRIGATION

> A report/template for planning, implementation, and handover. Ready for printing or presentation.

---

0) EXECUTIVE SUMMARY

- **Goal:** Automate tree washing/irrigation based on real conditions and schedules; reduce human effort; save ≥ **30–50%** water versus manual methods; keep foliage clean for better photosynthesis and urban aesthetics.
- **Expected results (KPIs):**
- System uptime ≥ **99%** (excluding planned maintenance).
- Correct alert rate (low tank, dry-run, leakage) ≥ **95%**.
- Water saving ≥ **30%**; complete device/data logs.
- Phased rollout **1 → 10 → 50** nodes with centralized management.
- **Scope v1.0:** 1 cloud MQTT broker + 1–10 **edge nodes** (ESP32 Wi-Fi), 12V pump/solenoid valves, soil moisture/rain/light/flow sensors, and a web/mobile app.

Terminology (quick) — Node vs Gateway vs Zone

- **Node (edge node / device / controller):** one physical control unit (ESP32 in an IP65 box) that reads sensors and drives a pump/valves. One **node** may control **one or multiple zones** (valves/nozzle groups).
- **Gateway/Broker:** the cloud (or local) message hub (e.g., EMQX/Mosquitto). It routes data/commands; it **does not** directly switch water lines.
- **Zone:** a logical/physical watering circuit (one valve or a nozzle group). A node can host `zone_1…zone_N`.

Mini-topology example

```
Cloud Broker ⇄ Backend/Dashboard ⇄ Node#A (ESP32)
                                    ├─ Zone1 → Valve1 → Pot A
                                    ├─ Zone2 → Valve2 → Pot B
                                    └─ Zone3 → Valve3 → Tree C
```

---

1) CONTEXT — PROBLEM — SCOPE

1.1 Context & need

- Dusty urban areas: require **washing** leaves/trunks periodically to improve photosynthesis and appearance.
- Campuses/parks/farms: require **irrigation** optimized by soil moisture and weather.

1.2 Use cases

- **UC1 — Home/Balcony: 1 node** controlling **1–4 plant pots/containers**; mini pump + mist nozzles; app on/off, morning/evening schedule; rain skip.
- **UC2 — Campus/Park:** 10–50 watering points; zoning; per-zone water statistics; alerts for low tank/pump jam.
- **UC3 — Urban washing:** light-pressure wash cycles at night/off-peak; prioritize cleaning leaves/trunks without wasting water.

1.3 In scope vs Out of scope

- **In:** washing/irrigation automation, remote monitoring, scheduling, alerts, OTA updates, dashboard.
- **Out:** computer vision, autonomous tanker vehicles, automated fertilization (placed on roadmap).

---

2) SYSTEM REQUIREMENTS

2.1 Functional requirements (FR)

1. Read sensors: soil moisture, rain, light, flow, ambient temperature/humidity.
2. Control **pump/valves** by thresholds, schedules, and manual commands from the app.
3. Skip washing/irrigation when it rains; stop on dry-run or leakage detection.
4. Scheduling (cron-like), multi-zone control, **manual override** with **safe timeout**.
5. **Logging & telemetry:** record events/metrics and send via MQTT/HTTPS; buffer when offline.
6. **OTA** firmware; remote configuration (Wi-Fi, thresholds, schedules, zones…).
7. **Alerts:** tank low/dry-run/abnormal pressure-flow/disconnect.

2.2 Non-functional requirements (NFR)

- Outdoor durability **IP65**; UV-resistant; waterproof cabling and rust protection.
- Security: TLS, key management, **secure boot** & **flash encryption** (ESP32).
- Control latency < 2 s on LAN; < 5 s via cloud. Scalability to **100+ nodes**.
- Power: battery/solar or 12V adapter; deep sleep, watchdog, auto-recovery.
- Maintainability: replace filters/nozzles; check valves/pumps; hot-swap node.

2.3 Constraints & assumptions

- Stable water source; pressure compatible with selected nozzles.
- Wi-Fi coverage (v1.0); upgradable to LoRa/NB-IoT later.
- Prototype budget **\~1.8–3.5 million VND/node** (indicative; depends on configuration).

---

3) SYSTEM ARCHITECTURE

```
flowchart LR
  subgraph Edge[Washing/Irrigation Node]
    S1[Soil moisture]
    S2[Rain]
    S3[Light]
```

```
    S4[Flow]
    MCU[ESP32]
    DRV[Relay/MOSFET]
    ACT1[Pump 12V]
    ACT2[Solenoid NC]
    PWR[Battery/Adapter]
    S1-->MCU; S2-->MCU; S3-->MCU; S4-->MCU
    MCU-->DRV-->ACT1
    MCU-->DRV-->ACT2
    PWR-->MCU; PWR-->ACT1; PWR-->ACT2
  end
  MCU==MQTT/Wi-Fi==>GW[(Internet/Broker)]
  GW<==HTTPS/MQTT==>APP[Web/Mobile Dashboard]
```

Key components: ESP32 (Wi-Fi), sensors (S1–S4), power drivers (relay/MOSFET + diode), 12V **NC** solenoid valve (fail-safe), power supply (12V adapter or battery+solar), MQTT broker (EMQX/Mosquitto/Cloud), backend (NestJS/Node), time-series DB, dashboard (Grafana/React).

Safety mechanisms:

- **NC valve** closes on power loss (fail-safe).
- **Dry-run**: if pump runs > X s but **flow ≈ 0** → stop + alert.
- **Hysteresis** around moisture threshold to prevent chattering.
- **Watchdog** & auto-recover; **fuse** + **flyback diode** on inductive loads.

---

4) HARDWARE DESIGN

4.1 Bill of materials (BOM — example)

Group	Suggested part	Notes
MCU	ESP32-WROOM / ESP32-S3	Wi-Fi, OTA, TLS
Soil sensor	Capacitive soil moisture (3.3–5V)	More durable than resistive
Rain	Rain sensor (analog/digital) or leaf switch	Skip cycle when raining
Light	BH1750/LDR	Optimize night/day schedule
Flow	YF-S201 (Hall)	L/min, detect dry-run/leak
Temp/RH	SHT3x/DHT22	Context sensor
Actuators	12V diaphragm pump 2–5 L/min; 12V NC solenoid	Washing/irrigation; NC preferred
Driver	Logic-level MOSFET (IRLZ44N) or SSR/Relay 10–30A	Include **flyback diode**
Power	12V-3A adapter **or** 12V LiFePO4 + 20–50W solar	Depends on mode

Group	Suggested part	Notes
Conversion	Buck 12→5V/3.3V (≥2A), INA219 (I/V sensing)	Power telemetry
Enclosure	**IP65** box, cable glands, PE tubing 6 mm, 0.2–0.5 mm nozzles, filter	Outdoor grade

4.2 Connections — ESP32 pin map (example)

- **I2C:** `BH1750 (0x23)`, `SHT3x (0x44)` on `GPIO 21 (SDA)`, `GPIO 22 (SCL)`.
- **Flow YF-S201:** `GPIO 27` (interrupt, pull-up). Formula `Q(L/min) = K × f(Hz)`; K ≈ 7.5 (model-dependent; calibrate).
- **Rain:** `GPIO 34` (ADC) or `GPIO 26` (digital).
- **Pump relay/MOSFET:** `GPIO 25`; **Valve:** `GPIO 33`.
- **Power path** through MOSFET + **Schottky diode** across valve coil.

4.3 Typical calculations

Water per wash (urban): 1 tree, 30 s, 2 L/min → **1 L/cycle**. Once/day → 30 L/month.

Energy budget (example):

- Pump 12V-2A = **24 W**. Runs 2 min/day → 24 × (2/60) = **0.8 Wh/day**.
- ESP32 + sensors (deep sleep; active 10 s/10 min): **\~0.2–0.4 Wh/day**.
- **Total \~1.2 Wh/day** (use **2 Wh/day** for margin).

Battery/solar sizing (sample):

- **Battery** for 2 days autonomy, 80% DoD: `2 Wh × 2 / 0.8 ≈ 5 Wh` → at 12V ≈ **0.42 Ah** (choose **2–4 Ah** to handle inrush & losses).
- **Solar** (4 peak-sun hours, system efficiency 0.6): `2 Wh / (4 × 0.6) ≈ 0.83 W` → choose **10–20 Wp** to cover losses and cloudy days.

  If **longer daily runtime** or multiple nozzles are required, use a **higher-power pump** and rescale Wh/day and battery/solar accordingly.

---

5) FIRMWARE DESIGN

5.1 Task architecture

- **State machine:** `IDLE → MEASURE → DECIDE → ACTUATE → VERIFY(FLOW) → LOG → SLEEP`.
- **FreeRTOS tasks:** SensorTask, ControlTask, CommsTask, OTA/ConfigTask, Watchdog.
- **Configuration** stored in NVS (thresholds, schedules, Wi-Fi, MQTT, zones...).

5.2 Communication & data

MQTT topics (example):

- Telemetry: `iot/treewash/{deviceId}/telemetry`
  Payload: `{ ts, soil, rain, lux, temp, hum, flow_lpm, pump: on/off, valve: on/off, rssi }`
- State: `iot/treewash/{deviceId}/state` (retained)
- Command: `iot/treewash/{deviceId}/cmd`
  Examples: `{ action:"start", duration_s:30, zone:1 }` / `{ action:"set", param:"soil_min", value:35 }`
- Ack/Result: `iot/treewash/{deviceId}/ack`

QoS: Telemetry QoS0/1; State/Config **QoS1 retained**; Commands QoS1.
Offline buffer: queue \~200 records; flush when online.

5.3 Control algorithm (pseudo)

```
if rain_detected or forecast_rain: skip
if soil < threshold_min - H and time_in_window(schedule):
  start pump/valve
  while t < duration_max:
    if flow_lpm < min_flow for > X s: stop & alert (dry-run)
    if manual_stop: break
  stop
else if mode == WASH and time_in_window(night):
  run spray for N×(on/off) cycles to save water
```

- Use **hysteresis H** to avoid oscillation; enforce a **hard timeout** for pump.
- **Dry-run** detection uses `flow_lpm` and **current draw** (INA219) for confirmation.

5.4 Security & OTA

- Mutual-TLS with per-device keys.
- **Secure boot & flash encryption** (ESP32).
- Signed OTA over HTTPS; rollback on failure.

---

6) BACKEND/CLOUD & APPLICATIONS

- **Broker:** EMQX/Mosquitto (Cloud/VM).
- **Ingest API:** NestJS/Node (WebSocket/MQTT bridge) → **Time-series DB** (TimescaleDB/InfluxDB).
- **Dashboard:** Grafana **or** custom React (Next.js): moisture/flow charts, device status, schedules, alert table, manual control.
- **Alerting:** Email/Telegram/Zalo OA; rule engine (flow = 0 while pump ON for > X s → CRITICAL alert).
- **Multi-tenant:** `orgId` in topic/payload; role-based access (Admin/Operator/View).

---

7) TEST PLAN & ACCEPTANCE

7.1 Test matrix (sample)

ID	Scenario	Key steps	Expected result
T01	Low soil moisture → pump ON	Force soil < min; within schedule window	Pump/valve ON; flow > min; stops by timeout
T02	Rain skip	Trigger rain sensor	Cycle skipped; log `rain_skip`
T03	Dry-run	Disconnect water inlet	Stop within X s; send `DRY_RUN` alert
T04	Wi-Fi loss	Turn AP off	Node continues by schedule; buffer logs; sync on reconnect
T05	OTA failure	Push corrupt firmware	Roll back to previous image; device returns online

7.2 Acceptance criteria

- 2-week outdoor continuous run; $\geq$ 95% successful cycles; no leaks; dashboard reflects state within $\leq$ 5 s.

---

8) OPERATIONS & MAINTENANCE

- **Monthly:** clean filters/nozzles; check leaks; tighten fittings.
- **Quarterly:** inspect battery/solar, renew sealant/weatherproofing; review abnormal alerts.
- **Firmware:** minor releases every 1–2 quarters; emergency patches for security.
- **Runbooks:** dry-run, leakage, prolonged offline, low pressure.

---

9) SAFETY & COMPLIANCE

- Isolate power and water paths; grounding; **fuse** on pump line; outdoor-rated cabling.
- **NC valve** default closed; physical emergency stop (E-stop) on cabinet.
- Avoid spraying on electrical equipment/pedestrian areas; schedule at night/off-peak.

---

10) COST ESTIMATES (REFERENCE)

10.1 Hardware per node (VND)

Item	Unit price (est.)	Notes
ESP32 + power module	200–300k	variant-dependent
Sensors (soil, rain, light, flow, temp/RH)	300–700k	bundle of 3–5 sensors
12V pump 2–5 L/min	300–700k	depends on flow/pressure

Item	Unit price (est.)	Notes
12V NC solenoid valve	150–300k	1–2 valves
Drivers (relay/MOSFET, diodes, fuse)	100–200k	
Tubing/nozzles/filter/accessories	150–300k	
IP65 enclosure, glands, sealant, wiring	150–300k	
Subtotal (no solar)	**1.35–2.80 million**	depends on configuration
Battery + Solar (optional)	+800k–1.5 million	10–20 Wp + battery

Prices fluctuate; industrial-grade valves/pumps increase costs.

10.2 Software & cloud (monthly)

Item	1–10 nodes	50 nodes
VM/Broker + DB	150–400k	500k–1.5m
Alerting (SMTP/Telegram)	\~0	\~0
Domain/HTTPS	30–60k	30–60k

10.3 Labor & installation

- Cabinet/node assembly: 2–4 hrs/node.
- Tubing/nozzle installation: 1–3 hrs/tree/point depending on terrain.

---

11) IMPLEMENTATION TIMELINE (8-WEEK SAMPLE)

Week	Main activities	Deliverables/Milestones
1	Finalize requirements, site survey, hydraulic sketch	Requirement doc, nozzle/tubing layout
2	BOM, procurement, PCB/box design	Final BOM, enclosure drawings
3	Firmware skeleton, MQTT schema	Telemetry demo → broker
4	Control + schedule + dry-run + OTA	Device running in lab
5	Backend (NestJS) + DB + Dashboard	Charts, alert table, manual control
6	Outdoor pilot 1–3 nodes	Soak test, tune thresholds
7	Acceptance testing, ops docs	Acceptance checklist
8	Scale to 10 nodes, handover	Final report, operator training

---

12) ROADMAP

- **v1.1:** Weather forecast API → smarter skip/adjust schedule.
- **v1.2:** Anomaly detection from current/voltage/flow (TinyML).
- **v2.0:** LoRa/NB-IoT for wide areas; multi-tenant RBAC; fleet/tanker integration (if applicable).

---

13) APPENDIX

13.1 Sample device configuration (JSON)

```
{
  "wifi": {"ssid": "...", "pass": "..."},
  "mqtt": {"host": "broker.example.com", "port": 8883, "tls": true},
  "thresholds": {"soil_min": 35, "hysteresis": 5, "min_flow_lpm": 0.5},
  "schedule": [{"zone": 1, "start": "22:00", "duration_s": 30, "days": [1,3,
5]}],
  "mode": "AUTO|WASH|MANUAL",
  "safety": {"max_run_s": 120, "dry_run_window_s": 10},
  "ota": {"url": "https://.../fw.bin", "sig": "..."}
}
```

13.2 REST API (condensed)

- `GET /devices/{id}` — device state; `GET /devices/{id}/metrics?from=...&to=...`
- `POST /devices/{id}/cmd` `{ action, zone, duration_s }`
- `PUT /devices/{id}/config` `{ ... }`

13.3 Quick install checklist

1. Mount IP65 box in a high, dry location; avoid direct harsh sunlight.
2. Aim tubing/nozzles at leaves/trunks for **washing** or at root zone for **irrigation**.
3. Install **filter** before pump/valves; test 30–60 s; check for leaks.
4. Configure Wi-Fi/MQTT; test manual → auto; verify logs/alerts.

13.4 Risks & mitigations

Risk	Impact	Mitigation
Dry-run	pump damage	flow sensor + timeout
Leakage	water loss	quality fittings, pressure test, anomaly alerts
Offline	loss of remote control	local schedule + buffering + resync
Soil sensor failure	wrong decisions	calibration, sensor voting, schedule fallback
Clogged nozzle	no wash/irrigation	filter, maintenance schedule, pressure/flow logs

---

14) GLOSSARY (selected)

- **Node (edge node / controller):** physical device with MCU, sensors, and drivers that controls one or more zones.
- **Gateway/Broker:** messaging hub (cloud/local) that routes telemetry/commands.
- **Zone:** a single watering/washing circuit controlled by a valve/nozzle group.
- **Actuator:** pump, solenoid valve, or motor that moves water.
- **Dry-run:** pump running without water flow; dangerous to the pump; must auto-stop.
- **NC valve:** Normally Closed valve; safe default closed on power loss.

Note: For **washing** (urban dust removal) vs **irrigation** (soil moisture), choose appropriate nozzle type (mist vs drip), flow, and decision logic (night schedule vs moisture-based). This outline covers both and can be tailored per scenario.