

AgriVision AI

Edge-AI Smart Agriculture Working Model

Raspberry Pi 4 + Google Coral USB Edge TPU

A publication-ready build and reproducibility guide combining real sensor-controlled irrigation, local plant-health inference, and a transparent model-release workflow.

DEVELOPED BY

Isis Saritha Swapin

Class VIII | PeachBot Author

Contributor: Swapin Vidya



PeachBot | SBC for Biology

Publication Edition v1.0 | 27 August 2026

SOURCE CODE • MODEL TRAINING • EDGE DEPLOYMENT • EXHIBITION GUIDE

PUBLICATION INFORMATION

Authorship, scope and edition status

This edition updates the original 28-page build package to match the public GitHub repository and the PeachBot-owned Hugging Face release path.

Role	Name / Organization	Responsibility
Primary developer / project author	Isis Saritha Swapin	AgriVision AI project, code authorship and student presentation
Contributor	Swapin Vidya	Architecture support, engineering review, documentation and implementation guidance
Publisher / brand	PeachBot AI	Public project and model publishing identity

Public identifiers

Source repository: github.com/swapins/agrivation-ai-pi

Model repository: huggingface.co/peachbotAI/agrivation-mobilenetv2-edge-tpu

Author profile: peachbot.in/people/4

This document is a reproducibility and exhibition guide, not a claim that the system has completed field validation. Wherever validation is pending, the status is stated explicitly.

- Source code license: MIT (repository).
- PlantVillage training data used by the official pipeline: CC0 1.0.
- Hugging Face model repository has been created under PeachBot AI; final weights remain release-gated until measured criteria pass.

OVERVIEW

Abstract and guide map

Abstract

AgriVision AI is a Class VIII smart-agriculture working model that combines a Raspberry Pi 4, Google Coral USB Edge TPU, camera, soil-moisture sensing, environmental sensing, low-voltage irrigation and a local dashboard. The first public AI release is intentionally simplified to a binary Healthy / Problem image classifier, while water-stress behavior is demonstrated independently through real soil sensing and irrigation logic. The project emphasizes local inference, measurable validation, safe low-voltage construction and reproducible publishing.

Part	What it covers
01 - System	Project question, architecture, physical model and hardware
02 - Build	Wiring, calibration, Raspberry Pi, camera and Coral setup
03 - Software	Repository structure, dashboard and operating flow
04 - AI model	Dataset, training protocol, INT8 export and release gate
05 - Verification	Testing, current validation status and Edge TPU deployment
06 - Present	90-second demo, viva, troubleshooting and project report
07 - Reproduce	Code attribution, references, links and final checklist

01 PROJECT DEFINITION

From school idea to transparent Edge AI

One clear question guides the project: can a low-cost local system inspect visible plant health and automate watering without depending on cloud inference?

SEE	THINK	WATER	PROTECT
Camera + sensors	Raspberry Pi + Coral TPU	Safe 5 V irrigation	Confidence + safety limits

- Measure soil moisture, temperature and humidity.
- Capture a leaf at a fixed scanning position.
- Run local INT8 plant-health inference on a Coral-compatible model.
- Switch a separately powered 5 V pump through a MOSFET only when irrigation rules allow it.
- Show results locally on the dashboard and with LEDs / buzzer.
- Return UNCERTAIN when image confidence is below the chosen threshold.

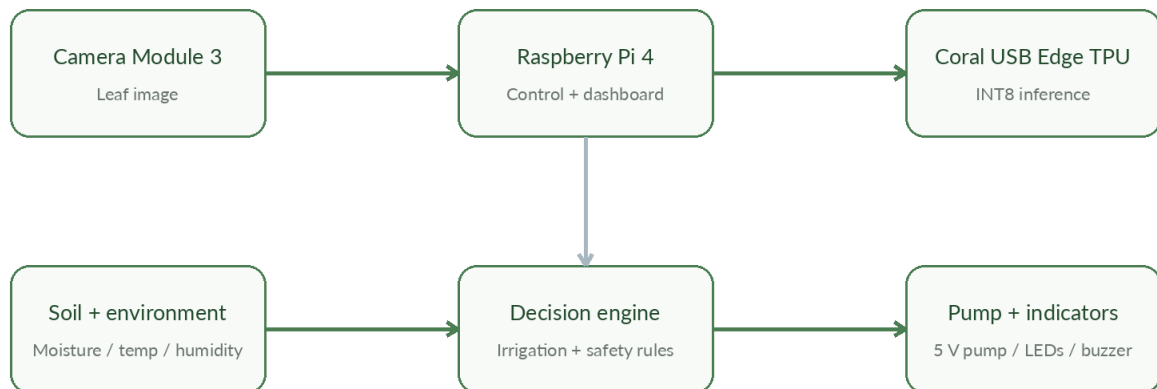
Success criterion

The exhibition must visibly demonstrate two real functions: (1) sensor-controlled watering and (2) genuine local image inference. Everything else improves reliability, reproducibility and presentation quality.

01 SYSTEM

System architecture

AgriVision AI - System Architecture



All core inference and irrigation decisions remain local after software/model deployment.

Figure 1. Core data paths and local decision boundaries.

- Vision path: Camera -> Raspberry Pi -> Coral-compatible model -> Healthy / Problem + confidence -> dashboard / indicators.
- Irrigation path: Soil sensor -> ADS1115 -> Raspberry Pi -> hysteresis and safety rules -> MOSFET -> 5 V pump.
- Environment path: BME280 -> Raspberry Pi -> dashboard and project log.

01 SYSTEM

Core hardware and bill of materials

Item	Qty	Purpose	Priority
Raspberry Pi 4 (4 GB recommended)	1	Main controller + dashboard	Essential
Google Coral USB Accelerator	1	Edge TPU inference	Essential
Raspberry Pi Camera Module 3	1	Leaf capture	Essential
Capacitive soil moisture sensor	1	Soil moisture	Essential
ADS1115 16-bit ADC	1	Analog sensor readout	Essential
BME280	1	Temperature / humidity	Recommended
5 V mini pump	1	Irrigation	Essential
Logic-level MOSFET module	1	Safe pump switching	Essential
Separate 5 V pump supply	1	Pump power	Essential
Green / yellow / red LEDs	1 each	Status indication	Essential
3.3 V-compatible active buzzer	1	Brief warning	Recommended
Breadboard, wires, tube, reservoir	1 set	Physical prototype	Essential

Electrical rule

Never power the pump from a Raspberry Pi GPIO pin. GPIO provides a logic signal only; the pump uses its own 5 V supply through a suitable MOSFET driver.

01 SYSTEM

Physical exhibition model

Use separation and repeatability as design principles: water on one side, electronics on a raised dry platform, and a fixed camera scan area.

Recommended Exhibition Base Layout - 60 cm x 45 cm

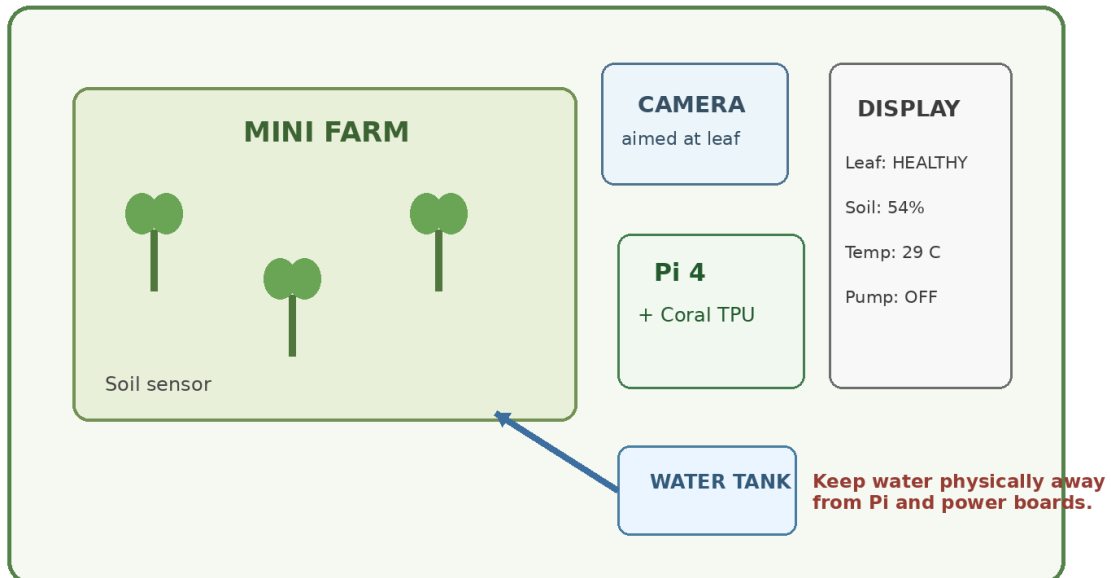


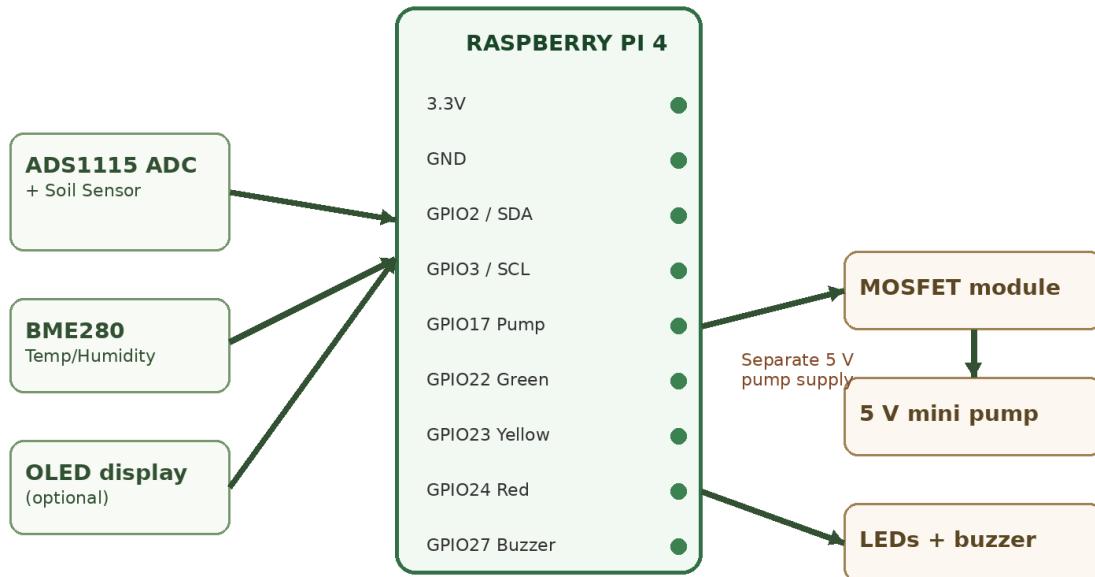
Figure 2. Recommended tabletop arrangement (about 60 cm x 45 cm).

- Keep the water tank below or beside the farm bed - never above the Raspberry Pi.
- Mount the camera about 20-35 cm from the leaf and mark the scanning position.
- Use a plain white or light-grey background and consistent room lighting.
- Label the visible modules so the student can explain the complete signal path.

02 BUILD

Wiring and GPIO plan

Simplified Wiring Map (BCM GPIO numbering)



Important: common ground between Pi logic and MOSFET control. Do NOT power the pump from a GPIO pin

Figure 3. Simplified BCM wiring map; keep a common logic ground where required.

Function	BCM / bus	Notes
Pump MOSFET signal	GPIO17	Logic control only
Green LED	GPIO22	Healthy
Yellow LED	GPIO23	Warning / uncertain / stress context
Red LED	GPIO24	Problem alert
Buzzer	GPIO27	Brief alert only
I2C SDA / SCL	GPIO2 / GPIO3	ADS1115 + BME280
Soil sensor	ADS1115 A0	Calibrate dry and wet endpoints
Coral USB	USB 3	Preferred for throughput

02 BUILD

Safety and good engineering practice

- Use only low-voltage DC equipment on the exhibition table.
- Keep tubing and the reservoir below and physically separated from the Pi, Coral and breadboard.
- Power off before moving the CSI camera ribbon or rewiring the breadboard.
- Use maximum pump runtime and cooldown limits in software even if a sensor fails.
- Use the standard / reduced-frequency Edge TPU runtime for a student-facing display unless a higher-frequency mode is specifically required and thermally protected.
- Cover exposed electronics so visitors cannot pull cables or touch hot components.
- Treat AI output as a prototype classification, not professional agricultural advice.

Water + electronics

The mechanical layout is part of the engineering solution. Safe placement is more important than hiding wires or making the model visually dense.

02 BUILD

Soil calibration and irrigation logic

Capacitive probes vary from unit to unit. Convert raw readings to a percentage only after measuring the actual dry and wet endpoints of the sensor used in the model.

Step	Action
1	Record about 20 readings with the cleaned sensor in air; average -> DRY_RAW.
2	Record about 20 readings in fully wet soil; average -> WET_RAW.
3	Map new readings between these values and clamp the result to 0-100%.
4	Start with pump ON below ~30% and STOP above ~45% for exhibition use.
5	Limit each pump cycle to a few seconds and impose a cooldown.

Why hysteresis matters

Separate ON and OFF thresholds prevent rapid switching when the moisture reading moves around a single boundary.

For a repeatable demonstration, keep a second cup of dry soil and move the probe between normal and dry soil rather than waiting for a pot to dry naturally.

02 BUILD

Raspberry Pi software stack

The deployment target should be chosen for Coral/PyCoral compatibility rather than simply using the newest operating-system image.

Layer	Recommended approach
Operating system	Compatibility-first 64-bit Raspberry Pi OS / Debian environment validated with the chosen Coral runtime.
Camera	Picamera2 / rpikam stack. Do not use the old PiCamera / raspistill legacy stack.
Web UI	Flask dashboard on the local network.
GPIO / sensors	gpiozero + Adafruit CircuitPython libraries for ADS1115 / BME280.
Inference	PyCoral runtime with an Edge TPU-compiled TFLite model.

Compatibility note

Coral software packaging is older than current Raspberry Pi software. Validate the exact OS, Python and Edge TPU runtime combination on the physical Pi before exhibition day.

02 BUILD

Camera and Coral validation

Module	Pre-integration test	Pass condition
Camera Module 3	Use <code>picam</code> / <code>Picamera2</code> to capture a still image.	Sharp image at the marked leaf distance.
Coral USB Accelerator	Install Edge TPU runtime, reconnect accelerator and run an official example model.	Inference result is returned without runtime errors.
USB connection	Prefer a USB 3 port.	Stable detection during repeated scans.

- Validate camera and Coral independently before connecting the soil sensor, pump and dashboard.
- Use `Picamera2` for Python capture; Raspberry Pi documentation names current capture applications `picam-*` on Bookworm and later.
- For a school display, avoid maximum-frequency Edge TPU operation unless heat is managed and the device is kept out of reach.

03 SOFTWARE

Public source repository and package structure

Repository

github.com/swapins/agrivation-ai-pi

Path	Purpose
src/agrivation/	Runtime package: Flask app, controller, hardware, inference, state and logic.
src/agrivation/templates/ + static/	Local dashboard and PeachBot branding.
scripts/	Pi installation, Coral smoke test, soil calibration and hardware self-test.
training/	MobileNetV2 training, INT8 export, Edge TPU compile helper and Hugging Face publishing.
models/	Deployment artifact location and label documentation.
tests/	PC-safe configuration and logic tests.
.github/workflows/	Unit tests, smoke training, full training and guarded Hugging Face publication.

- Primary developer attribution is embedded in README, AUTHORS.md, CITATION.cff, Python package metadata and model-card material.
- Simulation mode is explicit and visibly marked; it is not presented as hardware evidence.

03 SOFTWARE

Runtime and dashboard flow

Operational flow

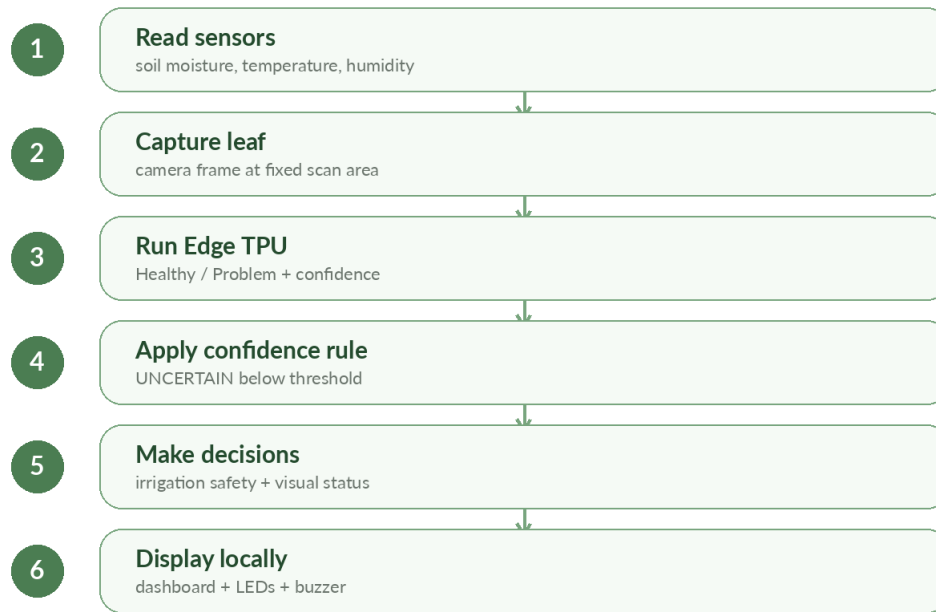


Figure 4. Exhibition operating flow with binary image classification.

Dashboard endpoints

/api/status - live state | /api/scan - on-demand leaf scan | /latest.jpg - most recent image | /health - service status

04 AI MODEL

First release: Healthy / Problem

The publication edition replaces the original proposed three-class image model with a more defensible first release.

Signal	Output	How it is produced
Leaf image	Healthy	Binary MobileNetV2 classifier
Leaf image	Problem	Binary MobileNetV2 classifier - visible abnormality
Low confidence	UNCERTAIN	Confidence threshold in runtime
Dry soil	Water-stress context	Real soil-moisture sensor + irrigation logic
Environment	Temperature / humidity	BME280 readings

Why binary first?

A Healthy / Problem release reduces class ambiguity for a school exhibition and avoids fabricating a visual “stress” category without a separately validated stress-image dataset. Water stress remains visible through the real sensing subsystem.

04 AI MODEL

Dataset and training protocol

Parameter	Official release setting
Dataset	geraldmc/plantvillage-full @ v0.1.0
Dataset size	54,304 controlled-background RGB leaf images
Classes in source	38 host-disease combinations across 14 hosts
AgriVision task	Binary: Healthy / Problem
Architecture	MobileNetV2 alpha 0.35
Input	224 x 224 RGB
Initialization	Scratch for official PeachBot release lineage
Validation	Source held-out test assignment + deterministic validation fold
Export	Full-integer INT8 TFLite, uint8 input/output

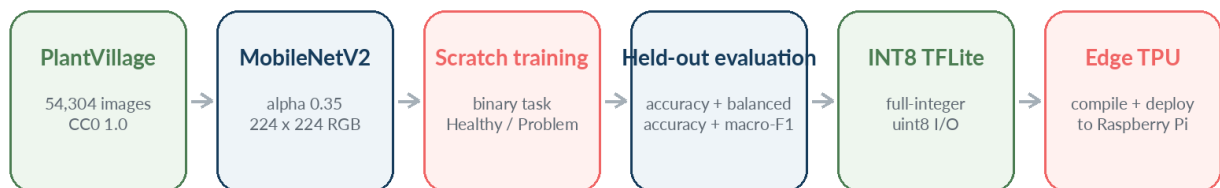
Dataset limitation

PlantVillage is largely photographed against plain backgrounds under controlled lighting. Strong held-out accuracy on this dataset does not establish robustness in real farms.

04 AI MODEL

Training, evaluation and release gate

AI model release pipeline



Release gate: full dataset + official lineage + INT8 held-out accuracy $\geq$ 90% before Hugging Face publication

Figure 5. Official model lineage and publication gate.

- Report float and INT8 held-out accuracy.
- Also record balanced accuracy, macro-F1, confusion matrix and per-class precision / recall.
- Use a class-aware representative set for INT8 calibration.
- Block public publication if the artifact is a smoke/debug model or fails the release threshold.

04 AI MODEL

PeachBot-owned Hugging Face publication

Reproducible publication path

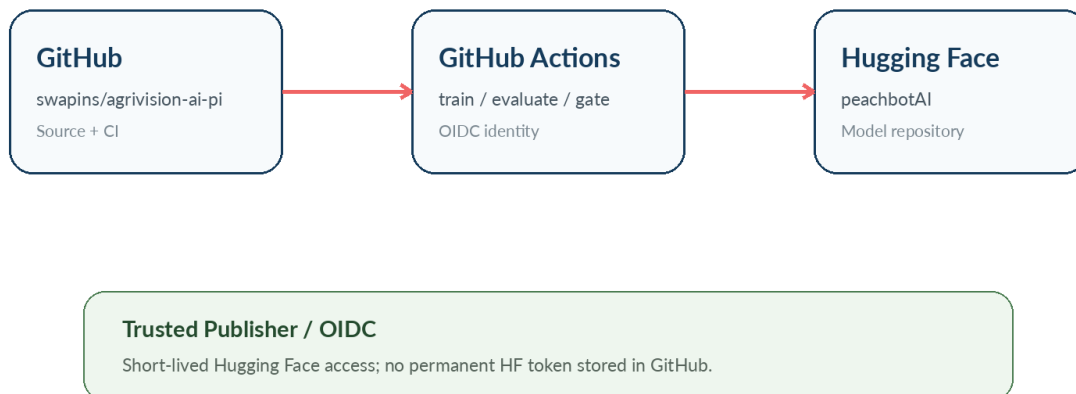


Figure 6. GitHub Actions publishes through a Hugging Face Trusted Publisher.

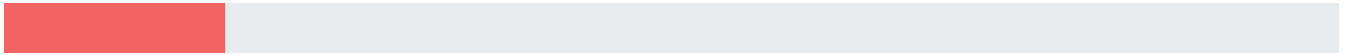
Resource	Identifier
GitHub source	swapins/agrivation-ai-pi
GitHub branch	main
Publishing workflow	train-and-publish-hf.yml
Hugging Face organization	peachbotAI
Model repository	peachbotAI/agrivation-mobilenetv2-edge-tpu

Credential model

The trusted-publisher route uses GitHub Actions OIDC to obtain short-lived, repository-scoped Hugging Face access. A permanent HF token does not need to be stored in GitHub.

05 VERIFICATION

Current validation status



Validation status - publication cut-off

Source repository	PUBLIC	Complete PeachBot-branded code package
Unit tests	PASS	5 / 5 local tests passed
CI smoke pipeline	PASS	Tiny debug dataset: train -> INT8 -> metrics -> artifact
Official full training	IN PROGRESS	Full PlantVillage binary scratch-training workflow launched
Final model publication	GATED	Publishes only after release criteria pass
Raspberry Pi + Coral hardware	PENDING	Must be verified on the physical exhibition unit

Figure 7. Status at the publication cut-off; no uncompleted validation is represented as finished.

The CI smoke run proved that the complete software path can download data, train, quantize to INT8, evaluate, write the model manifest and upload an artifact. The smoke dataset is debug-only and its accuracy is deliberately not reported as project performance.

05 VERIFICATION

Edge TPU compilation and deployment

The Edge TPU does not train the model. Training and quantization happen on a laptop, cloud runner or notebook. A compatible full-integer TFLite model is then compiled for the Edge TPU and copied to the Raspberry Pi.

Artifact	Purpose
plant_health.keras	Training checkpoint / float model
plant_health_int8.tflite	Full-integer TFLite before Edge TPU compilation
labels.txt	Class order: healthy, problem
model_manifest.json	Dataset, lineage, metrics and limitations
plant_health_edgetpu.tflite	Final Coral deployment artifact after successful compilation

Compiler check

Review the Edge TPU compiler report. If important operations remain on the CPU, the model may run but will not provide the intended level of accelerator execution.

05 VERIFICATION

Module-by-module test plan

Stage	Test	Pass condition
1	Pi boot	Stable boot; no repeated power warnings
2	I2C scan	ADS1115 and BME280 detected
3	Soil sensor	Clear dry / wet raw-value separation
4	LEDs + buzzer	Each output switches independently
5	Pump	1-2 second MOSFET-controlled pulse; Pi stays stable
6	Camera	Sharp image at scan distance
7	Coral official example	Inference succeeds
8	Custom INT8 / Edge TPU model	Labels and inference load correctly
9	Dashboard	Status updates; SCAN LEAF works
10	Integrated run	Dry soil waters safely; leaf scan drives correct state

Debug discipline

Make each module pass independently before integrating it. This reduces exhibition-day troubleshooting dramatically.

06 PRESENT

90-second exhibition demonstration

Time	Action / script
0-15 s	"This is AgriVision AI, a smart-agriculture model using Raspberry Pi and a Coral Edge TPU."
15-30 s	Point to the soil sensor and show the live moisture reading.
30-45 s	Move the probe to the prepared dry-soil cup; show the limited 5 V pump cycle.
45-60 s	Place a healthy leaf/card in the scan area and press SCAN LEAF.
60-75 s	Scan a problem leaf/card; show the alert state and confidence.
75-90 s	Point to the Coral: "The compatible neural-network inference runs locally; each image is not sent to a cloud server."

One-line conclusion

AgriVision AI demonstrates how sensing, automation and local machine-learning inference can be combined into a practical low-cost smart-farming prototype.

06 PRESENT

Viva questions and model answers

Q: What is edge AI?

A: AI inference performed close to where data is produced instead of sending every input to a remote cloud service.

Q: Why use a Coral TPU?

A: It accelerates compatible TensorFlow Lite inference so the Raspberry Pi can run a small neural network locally.

Q: Why is an ADC needed?

A: Raspberry Pi GPIO does not directly measure analog voltage; ADS1115 converts the soil-sensor signal into a digital value.

Q: Why use a MOSFET?

A: A GPIO pin cannot safely supply pump current; the MOSFET lets a small logic signal switch a separately powered load.

Q: What is quantization?

A: Representing model values with lower-precision integers so inference can be efficient on hardware such as the Edge TPU.

Q: Why only Healthy / Problem in the first model?

A: It is a clearer first classification target; water stress is demonstrated separately with real moisture sensing.

Q: Can it diagnose crop disease?

A: No. It is a prototype image classifier and needs validated field data and expert confirmation for real diagnosis.

Q: What happens at low confidence?

A: The software returns UNCERTAIN instead of presenting a weak prediction as a fact.

06 PRESENT

Troubleshooting guide

Problem	Likely cause	Action
Coral not detected	Runtime / USB issue	Reconnect accelerator, prefer USB 3 and validate an official example.
Camera not found	Ribbon / camera-stack issue	Power off, reseal cable, list cameras and verify Picamera2 / rpical tools.
Pi reboots when pump starts	Pump current/noise affects Pi	Use separate pump supply, proper MOSFET and common control ground.
Soil % reversed	Calibration endpoints reversed	Verify DRY_RAW and WET_RAW; mapping supports either direction.
Pump chatters	No hysteresis/cooldown	Use separate start/stop thresholds and cooldown.
AI predicts one class	Imbalance / preprocessing / undertraining	Inspect confusion matrix, class weights, input scaling and training duration.
Result changes with background	Dataset background bias	Use fixed scan background and later collect field-like images.
Model fails to load	Wrong model / not compiled	Test the INT8 model separately and confirm Edge TPU compilation and path.

06 PRESENT

Ready-to-use project report content

Aim

To design and demonstrate a smart-agriculture prototype that combines soil monitoring, automatic irrigation and local artificial-intelligence image classification using a Raspberry Pi and Coral Edge TPU.

Principle

The system combines sensing, edge computing and automation. A capacitive sensor measures soil moisture through an ADC; the Raspberry Pi applies irrigation safety rules and controls a low-voltage pump through a MOSFET. A camera captures a leaf image and a quantized TensorFlow Lite classifier performs local plant-health inference. Results are displayed locally.

Innovation / educational value

- Combines physical automation with genuine machine-learning inference.
- Demonstrates why dedicated AI accelerators are useful at the edge.
- Introduces calibration, electrical load switching, computer vision, model confidence and reproducibility in one project.

Limitations

- Image quality and classification depend on dataset, lighting and background.
- The model is not a professional crop-diagnosis device.
- Soil moisture is measured at one point and the irrigation logic is designed for a tabletop model.

06 PRESENT

Future scope

Next step	Why it matters
Field-image dataset	Reduce controlled-background bias and test real leaves in realistic conditions.
Crop-specific multiclass models	Move from generic Problem to named, validated disease categories.
Dedicated stress dataset	Separate image-visible stress only when training evidence supports it.
Multi-zone irrigation	Independent sensors and valves for multiple plant areas.
Long-term logging	Trend charts for moisture, temperature and model outcomes.
Solar / battery enclosure	Demonstrate a field-oriented edge deployment.
LoRa / mesh networking	Connect multiple plots without continuous Internet.
Pi / Coral benchmark	Measure real inference latency, thermal behavior and power on the final hardware.

Research direction

The most valuable next scientific improvement is not a larger model; it is better field data and a clear validation protocol.

07 REPRODUCE

Code attribution and reproducibility

Item	Publication attribution
Project / code	Developed by Isis Saritha Swapin
Contributor	Swapin Vidya
Publishing identity	PeachBot AI
Source code	github.com/swapins/agrivation-ai-pi
Model target	huggingface.co/peachbotAI/agrivation-mobilenetv2-edge-tpu
Formal metadata	AUTHORS.md + CITATION.cff + package metadata + model card

Suggested citation

Citation

Isis Saritha Swapin. AgriVision AI: Raspberry Pi + Coral Edge TPU Smart Agriculture Project. PeachBot, 2026.
Contributor: Swapin Vidya.

- Do not present simulation output as physical-hardware evidence.
- Do not report smoke-run metrics as final model performance.
- Preserve the generated model manifest with every published model artifact.

07 REPRODUCE

References and public links

- [1] Mohanty SP, Hughes DP, Salathé M. Using deep learning for image-based plant disease detection. *Frontiers in Plant Science* 7:1419 (2016). DOI: 10.3389/fpls.2016.01419.
- [2] PlantVillage full curated release: huggingface.co/datasets/geraldmc/plantvillage-full (CC0 1.0).
- [3] Google Coral USB Accelerator documentation and datasheet: coral.ai/docs/accelerator/get-started/ and coral.ai/static/files/Coral-USB-Accelerator-datasheet.pdf.
- [4] Raspberry Pi camera software: raspberrypi.com/documentation/computers/camera_software.html.
- [5] Picamera2 manual: datasheets.raspberrypi.com/camera/picamera2-manual.pdf.
- [6] Hugging Face Trusted Publishers: huggingface.co/docs/hub/trusted-publishers.
- [7] AgriVision AI source: github.com/swapins/agrivision-ai-pi.
- [8] PeachBot model repository: huggingface.co/peachbotAI/agrivision-mobilenetv2-edge-tpu.



GitHub source



Hugging Face model



Author profile

07 REPRODUCE

Final exhibition and publication checklist

- ☐ Pi boots reliably and the dashboard can be opened quickly.
- ☐ Camera capture has been tested at the marked scan distance.
- ☐ Coral official example has passed on the physical unit.
- ☐ Final AgriVision model artifact has passed the release gate and is installed.
- ☐ Dry and wet soil calibration values are recorded for the actual sensor.
- ☐ Pump runtime and cooldown are conservative; tubing cannot spray electronics.
- ☐ Green / yellow / red indicators and buzzer work independently.
- ☐ Healthy and problem sample leaves/cards are ready for repeatable scanning.
- ☐ Student can explain edge AI, inference, ADC, MOSFET, quantization and uncertainty.
- ☐ Backup screenshots/video are stored locally in case of exhibition hardware failure.
- ☐ Public README, model card and PDF use the same developer / contributor attribution.
- ☐ No pending validation is described as complete in publication material.



AgriVision AI

Build locally. Measure honestly. Publish reproducibly.

Developed by Isis Saritha Swapin

Contributor: Swapin Vidya | Published by PeachBot AI



Source code



Model repository

peachbot.in