ADAPTIVE AGROTECH

EXT 2026 / STUDENT EDITION

YOUR PROJECT

FieldScout
LiDAR navigation

Design the architecture. Build the interface.
Demonstrate autonomous movement.

PHYSICAL PROTOTYPE / SUPPLIED COMPANY PHOTOGRAPH

Your goal: give the existing robot a Jetson-based navigation system that senses its surroundings, estimates its position and reaches waypoints through the ESP32 motor controller.

Begin with two deliverables: 1. Bill of materials. 2. Signal and power block diagram.
Use the browser lab to understand commands before integrating real hardware.

01 / TRY THE LEARNING LAB

Use the Simulation Studio

Extract the full ZIP and open **index.html**. Explore the original 3D robot on that page, then open the Simulation Studio to test motion, LiDAR and controller messages.

SIMULATION ONLY

The browser does not connect to the physical robot. Scans, pose and motion are simulated; message previews are not transmitted to motor hardware.

FASTEST START

Click Start example mission. The Studio loads a route, enables the simulated controller and begins. **Follow robot** and zoom make the vehicle easier to see without enlarging it relative to the map.

DRIVE, STOP AND INSPECT

Action	What to do
Manual motion	Click **Start manual driving**. Hold W/A/S/D, arrow keys or a direction button. Release to stop.
Stop or disable	Use Space or **Stop simulation**. **Stop & disable** stops and disables motion.
Emergency stop	Click **Emergency stop** or press X. After **Reset emergency stop**, start manual driving or a mission deliberately.
Your own route	Click clear ground to add waypoints, then **Run my waypoints**. Inspect wheel commands and Modbus messages as it moves.

EXERCISE / A DETOUR OR A BLOCKED ROUTE?

Detour: start a mission and click **Add obstacle ahead**. Turn on **Original route** to compare paths. **No route:** click **Test blocked route**; a barrier appears during its example mission. The robot stops. Use **Remove added obstacles**, then **Resume mission** to continue the remaining goals.

Model scale: the schematic robot occupies 0.46 x 0.34 m including wheels; the collision envelope has radius 0.30 m. These are simulation settings, not measured hardware dimensions. The detail inset is not at map scale.

Original 3D: index.html#explorer opens the assembly on the same page. Use zoom and Fit arena in the Studio to inspect the arena without changing robot scale.

02 / YOUR FIRST DELIVERABLE

01. Bill of materials

List what is already available, what must be purchased, and what still needs verification. A parts list becomes useful when it includes interfaces, power and software support.

Item	What students must decide
Existing drive base	Identify chassis, motors, two motor drivers, ESP32 and cables. Retain this working drive architecture.
Jetson + storage + cooling	Photograph the exact board label. The project document specifies **Orin Nano**; original Nano is a different platform.
2D LiDAR + adapter	Compare exact SKUs for near blind range, scan rate, dark objects, sunlight, field of view and ROS 2 support.
IMU / measured feedback	Verify sensor availability and how encoder/driver feedback reaches the Jetson. Commands alone are not odometry.
Power and protection	Record battery chemistry, full-charge range, BMS and measured loads. Select fused, regulated branches accordingly.
Integration parts	Include mounts, fasteners, connectors, data cables, strain relief, cooling and test equipment.

Required columns: quantity, exact part/revision, available/buy/TBC, interface, voltage/current, driver/software, supplier quote, tax/shipping, lead time and owner. Unknown prices stay blank or TBC; do not invent availability.

CHOOSE A COMPATIBLE COMPUTE STACK

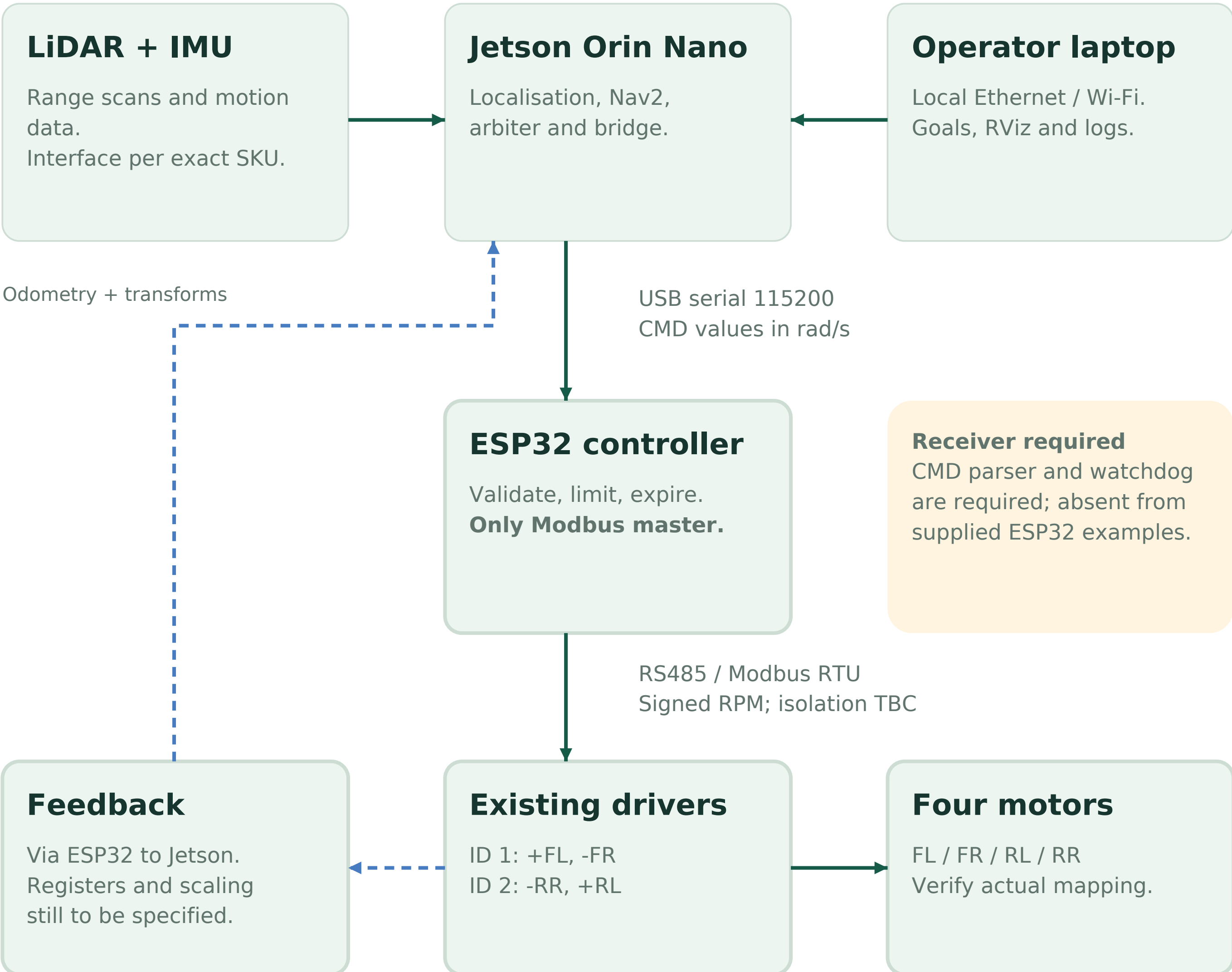
Board	Proposed software path
Orin Nano	Conservative baseline: JetPack 6.2.2 / Ubuntu 22.04 / ROS 2 Humble. Alternative: JetPack 7.2.1 / Ubuntu 24.04 / ROS 2 Jazzy after driver testing.
Original Nano	Legacy JetPack 4.x family. Do not assume that current Humble/Jazzy packages install natively; agree a tested plan or change the compute board.

Exit gate: reviewed BOM, provisional power budget, inventory photographs and a short list of unresolved choices. Obtain the project budget and purchase approval through the normal course/company process.

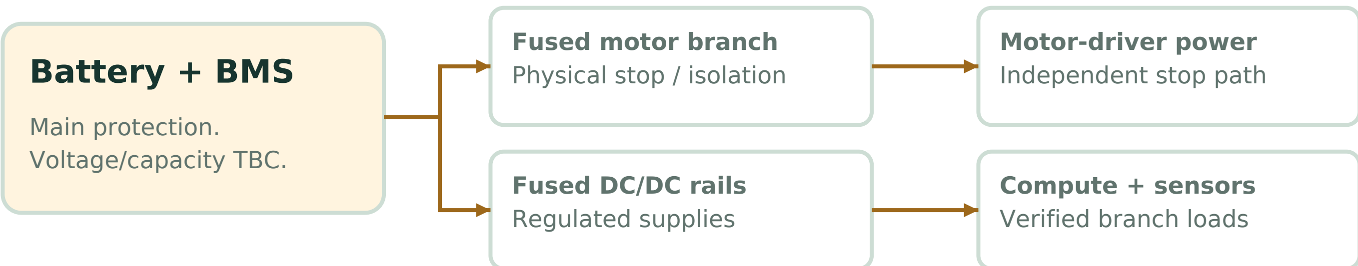
03 / YOUR SECOND DELIVERABLE

02. Draw the block diagram

Show both signal and power paths. Use this proposed architecture, then replace provisional labels with verified connectors, supplies and component choices.



POWER / VERIFY VOLTAGES AND LOADS



Submit: signal/power diagram plus connector pin table. Dashed blue is the proposed feedback path. Document grounding and isolation. A block diagram does not replace an electrical schematic.

04 / BUILD IN STAGES

03-10. Integrate, then navigate

The Studio uses known-map A* and ideal pose. Synthetic LiDAR reveals the full ideal geometry of new boxes and triggers A* replanning. It is not SLAM, DWA or a physical validation result.

03 Measure the baseline

Record loaded wheel radius, effective track, footprint, mass and component labels. Confirm wheel IDs and polarity with the robot supported.

04 Bring up sensors and power

Run LiDAR in RViz; verify /scan axes, ranges and timestamps. Log voltage, temperature, CPU load and sensor rate under realistic load.

05 Build the command receiver

Implement or obtain the ESP32 serial receiver, validation, arming and stale-command timeout. Prove it without motor power, then wheels-up.

06 Provide odometry and transforms

Use measured wheel/IMU data or a documented alternative. Establish map -> odom -> base_link and measured sensor transforms.

07 Map, then localise

Build and save an indoor 2D map. Restart and recover position at measured reference points before issuing autonomous goals.

08 Integrate Nav2 and the bridge

Configure footprint, obstacle inflation and speed limits. Replace the browser model with real localisation and one supervised command stream.

09 Run missions and fault tests

Visit three waypoints; distinguish successful obstacle detours from blocked-route stops. Test scan loss, USB loss, reboot and deliberate re-enabling.

10 Validate and hand over

Repeat trials and report failures. Deliver pinned software, calibration, wiring, maps, logs and a reproducible operator/build guide.

05 / FROM NAVIGATION TO WHEELS

One command, two interfaces

The Jetson requests logical wheel velocities. The ESP32 validates the request, converts units and wiring polarity, then writes the existing motor drivers.

A / JETSON TO ESP32

USB serial, 115200 baud; newline-terminated ASCII. The expected wheel order is **FL, FR, RL, RR**, in **rad/s**. Positive logical speed means robot-forward.

```
CMD,9.425,9.425,9.425,9.425\n
```

Illustrative forward request: approximately 90 RPM per wheel.

Conversion: RPM = rad/s x 60 / (2 x pi). Define rounding and limits explicitly. Apply hardware polarity exactly once. ROS positive yaw turns left; the Lua joystick's positive steering turns right.

B / ESP32 TO DRIVERS

Wheel	Driver / register	Signed driver RPM
Front left	ID 1 / 0x2088	+FL
Front right	ID 1 / 0x2089	-FR
Rear right	ID 2 / 0x2088	-RR
Rear left	ID 2 / 0x2089	+RL

The ESP32 is the **sole Modbus RTU master**. RS485 uses 115200, 8N1. Function 0x10 writes two signed 16-bit RPM registers starting at 0x2088; word bytes are high-first and CRC bytes low-first.

INTEGRATION STILL REQUIRED

The supplied ESP32 examples do not parse HELLO/CMD wheel packets. One prints a heartbeat; the other runs a timed drive sequence. A compatible receiver and its watchdog must be implemented or obtained before physical navigation.

A driver ACK confirms a register transaction, not actual wheel speed, pose or stopping. The existing demo STATUS counters are not measured robot telemetry.

06 / PROVE THE SYSTEM

Validation gates

These are proposed indoor prototype targets for agreement with the company and supervisor. They are not measured FieldScout performance or a certification claim.

Gate	Evidence required
Before motor power	Wiring/power review; finite-number and packet-length checks; wrong order/units rejected; boot begins stopped and disarmed.
Wheels supported	Correct wheel identity and forward signs; command limits; failure handling; USB disconnect; restart requires deliberate enable.
First floor movement	Begin at proposed 0.10 m/s; only raise the initial limit to 0.20 m/s after measured stopping tests pass. Use an approved clear area.
Autonomous trial	Three goals, ten runs; proposed >=9 complete without intervention; zero contacts in all runs; goal error <=0.20 m and heading <=15 degrees.
Obstacle response	Stop or replan without contact. Proposed >=0.25 m clearance from the measured footprint; enlarge the margin if testing requires it.
Fault response	Record last valid input, zero-target time and actual stopping time. Scan loss, command loss and reboot must not cause automatic restart.

TIMING IS A MEASUREMENT

The Lua design mentions a 400 ms ESP32 timeout, but that receiver is absent. Treat 400 ms as a proposed command-expiry target and use a monotonic controller clock. Blocking bus retries and deceleration can extend physical stopping time.

RECORD MORE THAN SUCCESS

Log pose error against independent marks, goal completion, interventions, minimum clearance, sensor age, command timing, power and failures. Do not score accuracy using only the estimator's own pose.

Operating envelope: begin on a dry, level, controlled indoor course with inert obstacles. Browser replanning is a learning example, not proof of real obstacle avoidance. A 2D scan can miss low/high objects, glass and drop-offs. Verify an independent physical motor-power stop.

07 / KEEP THE WORK FOCUSED

Questions, decisions, handover

The Student Q&A webpage contains all 77 original questions with full answers. This page highlights the decisions that affect your first weeks.

Do we redesign the motor drivers? The current brief retains the existing ESP32, drivers and drive platform. Design the navigation subsystem and the interface to it.

Which environment comes first? Propose a controlled indoor course. Greenhouse, outdoor, livestock, mowing, solar and fleet work require separate scope and validation.

Are brochure figures fixed requirements? They are reference claims, not supplied acceptance results. Measure the delivered unit; do not infer battery voltage, IP rating or runtime.

What must the company confirm? Hardware handover, exact inventory, actual firmware, native CAD/wiring, budget, support channel, test site and IP/publication terms.

PROPOSED REVIEW SEQUENCE

When	Bring evidence
First review	BOM, block diagram, inventory and problem formulation.
October	Sensor/power logs; receiver tests; wheels-up interface and odometry.
November	Saved map, repeatable localisation and first waypoint mission.
December	Repeated acceptance runs, rebuild rehearsal and final handover. The stated 23 December endpoint needs supervisor/company agreement.

Every Monday: one-page status, one reproducible result, blockers and the next test. **Final handover:** BOM, diagrams, CAD/mounts, pinned source/configuration, protocol, calibration, maps, logs, known issues and demo video.

OPEN NEXT

Build plan: build.html
Message reference: protocol.html
All 77 student questions: questions.html
Student toolkit: resources.html

OFFICIAL REFERENCES / REVIEWED 21 SEPTEMBER 2026

NVIDIA JetPack: developer.nvidia.com/embedded/jetpack/downloads
ROS platform support: ros.org/reps/rep-2000.html
Navigation setup: docs.nav2.org
Manufacturer LiDAR driver: github.com/Slamtec/sllidar_ros2