



TURF

Turf Upkeep and Robotic Fieldcare

Group 20
Aryan Tulsi
Samuel Eisert
Andrew Koch

The Team



Aryan Tulsi

Computer Engineering

Hardware and Firmware
Design, Main Board and
Regulator



Samuel Eisert

Computer Engineering

Software and Firmware
Design, Mobile App
Development



Andrew Koch

Title

Add a line about this team
member and their role.



Motivation

Lawn care takes time and is costly

Mowing your lawn can waste hours of time
Paying a lawn care service is costly

The Alternative is Expensive

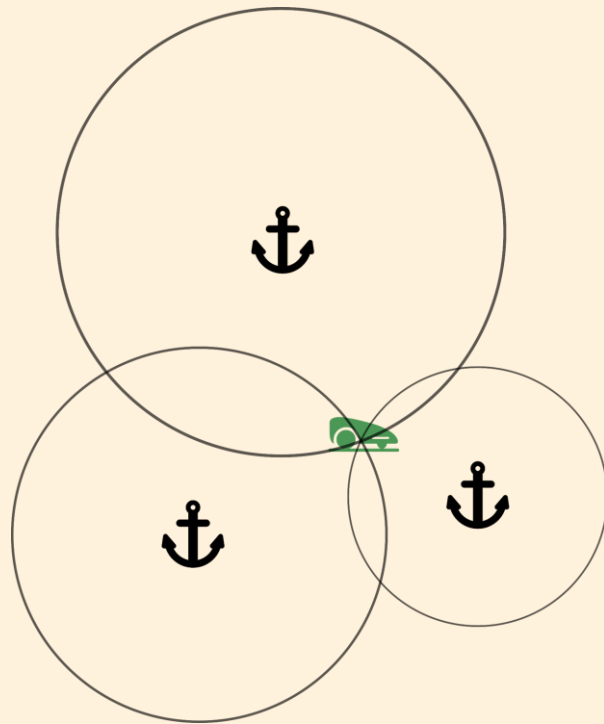
Current autonomous lawn mowers are expensive to purchase and
boundary wire systems incur additional setup costs



The Solution

Our solution is to build a robot using an affordable form of localization. Ultra Wideband Localization is a cost effective and precise alternative to GNSS or LIDAR.

UWB localization works by measuring the roundtrip time of a message between the robot and several bluetooth anchors at known locations. Once we know how far the robot is from at least three coordinates, we can calculate its current 2D position.





Goals



Base Goals

- Develop an Affordable Autonomous Lawn Mower Capable of navigating a pre-programmed coverage area
- Implement Obstacle Detection and Emergency Stop
- Develop a Companion App

Advanced Goals

- Implement automatic docking and Charging
- Modularize the design to allow for interchangeable tools

Stretch Goals

- Enable smart scheduling and weather-aware operation (pause/resume when raining).
- Add computer vision for object detection and dynamic avoidance of pets, people, or foreign object



Objectives



Base Objectives

- Implement localization using UWB TOF (Ultra-Wide Band Time-of-Flight).
- Implement control system for propulsion, steering, and mowing motor
- Demonstrate at least 10 minutes of untethered autonomous mowing within a defined test plot.
- Use ultrasonic rangefinder to stop before obstacles
- Design and fabricate a durable, weather-resistant housing for electronics and cutting mechanism.

Advanced Objectives

- Build and demonstrate an automatic docking/charging station for the mower.
- Implement a mowing schedule (i.e. once a week the mower mows the backyard on its own)

Stretch Objectives

- Incorporate rain and weather detection to suspend mowing in unsafe conditions.
- Use computer vision (OpenCV) to detect people, pets, or foreign objects in the mowing area, and automatically modify path.



Requirements Specifications



Marketing Requirements	Engineering Requirements	Justification
The mower should be able to autonomously cover an entire home lawn.	Must implement a row-by-row coverage algorithm (boustrophedon pattern), with ≤ 10 cm overlap between passes	Ensures full and efficient lawn coverage without leaving uncut strips.
The mower should be able to navigate accurately across the lawn.	Position estimation error ≤ 20 cm relative to field anchors using UWB TOF localization	Accurate localization is necessary to keep mowing rows straight and consistent.
The mower should be energy efficient to maximize runtime.	Single charge operation time ≥ 30 minutes; system enters sleep mode when idle.	Allows TURF to complete a standard suburban lawn in one session. Sleep mode conserves energy when idle.
The mower should be safe to operate around people and obstacles.	Must detect obstacles ≥ 1 m away and stop within 0.5 m of obstacle.	Safety for humans, animals, and equipment is critical for adoption.
The mower should be easy to recharge and maintain.	Automated docking station for charging; recharge time ≤ 10 hours.	Simplifies operation and maintenance for groundskeepers.
The mower should be affordable for small organizations (schools, clubs), and households.	Estimated BOM $< \$500$ for prototype.	Ensures feasibility for small institutions and households with limited budgets.

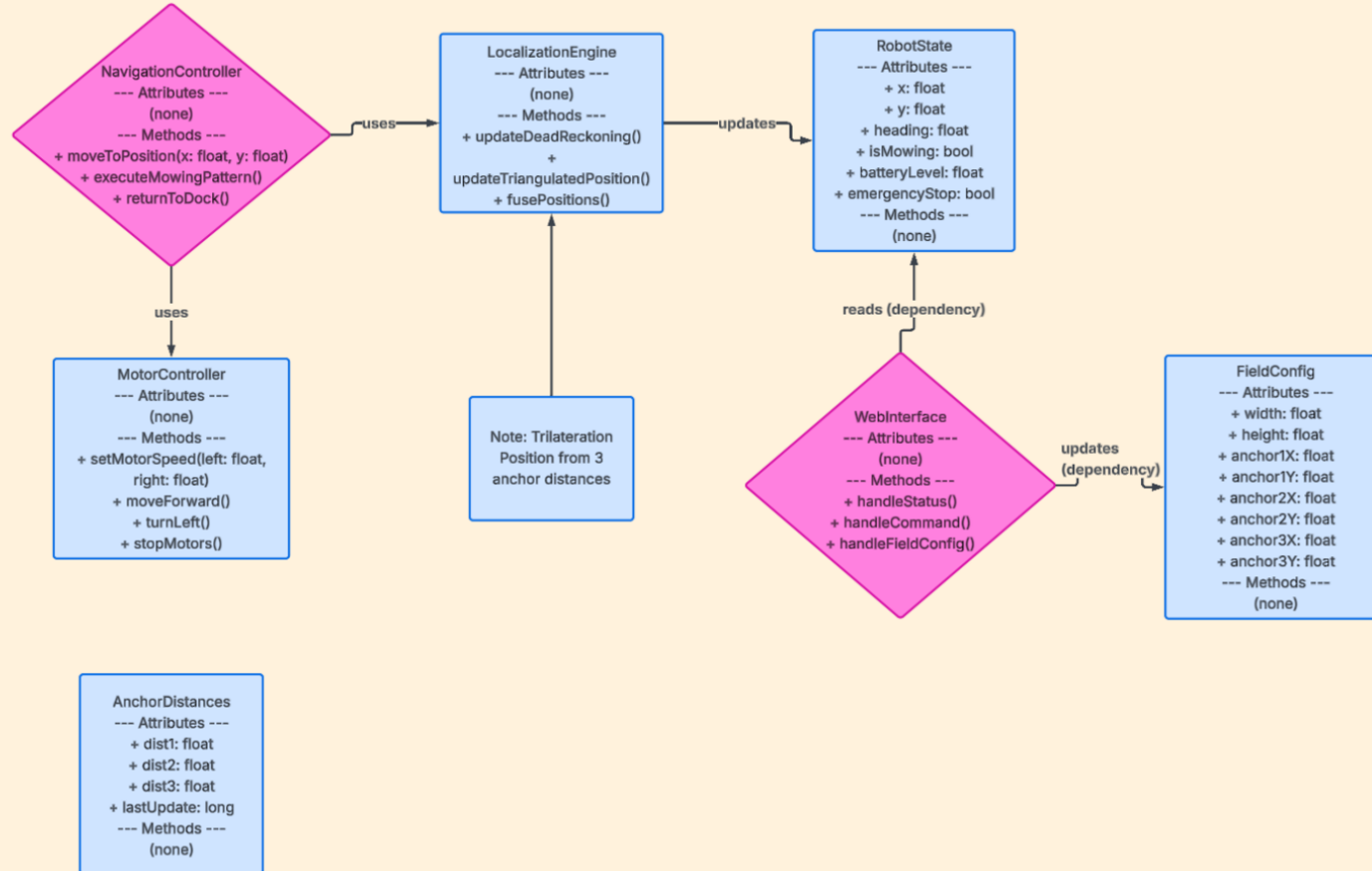


System Overview

Component	Technology Used	Functionality	Hardware
Mobile App	React Native ExpoGO EAS Build	User Interface Live Monitoring Area Visualization	Android
Firmware	C++	Motor Control Navigation Safety	ESP32 Microcontroller
Communication	Rest API (JSON)	Data Exchange	WiFi Hotspot
Positioning	Triangulation Algorithm Dead Reckoning	Locating Error Correction	GPS Anchors
Testing	Node.js	Software Simulation API Testing	Windows

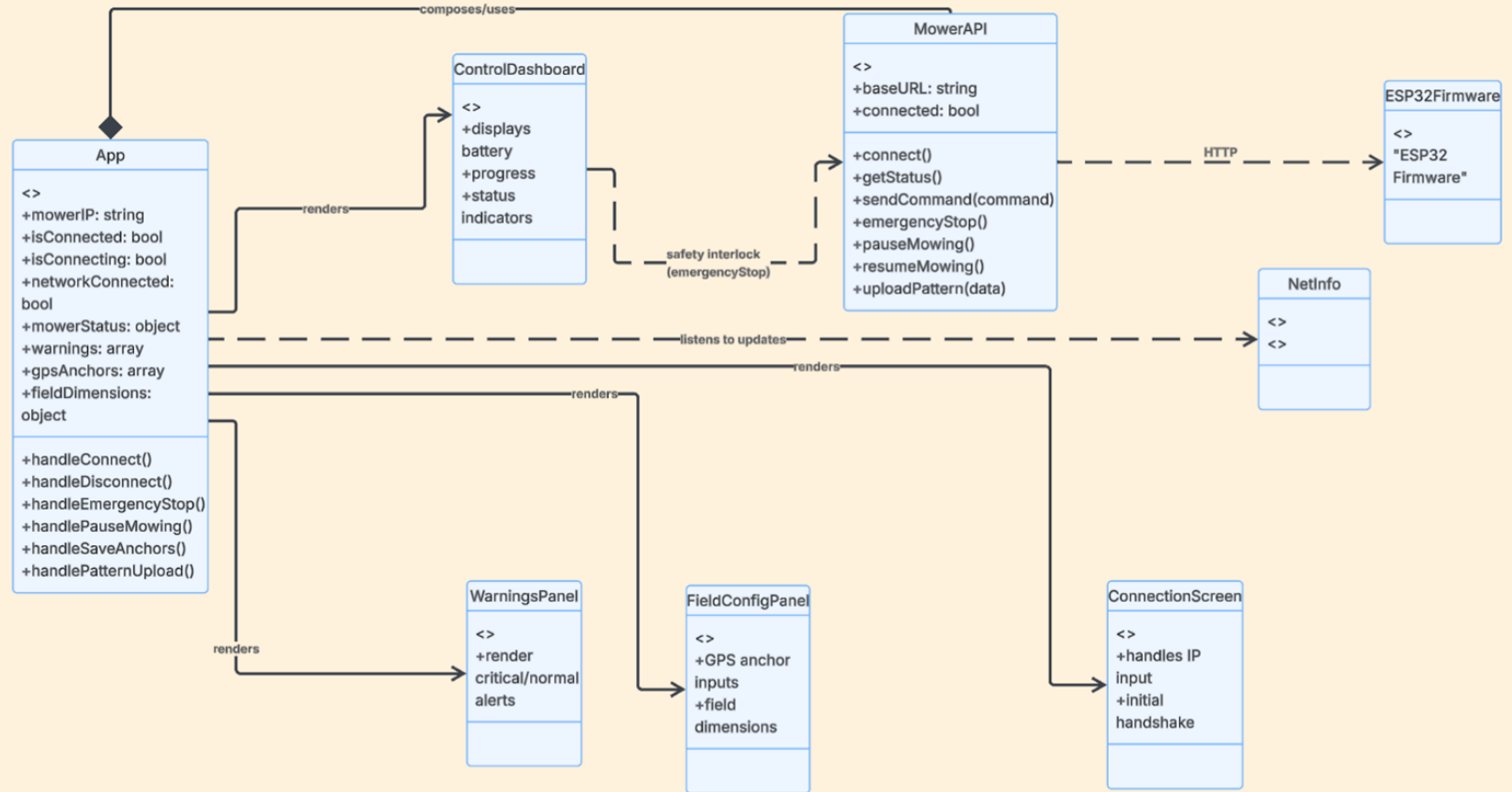


Class Diagram (Firmware)



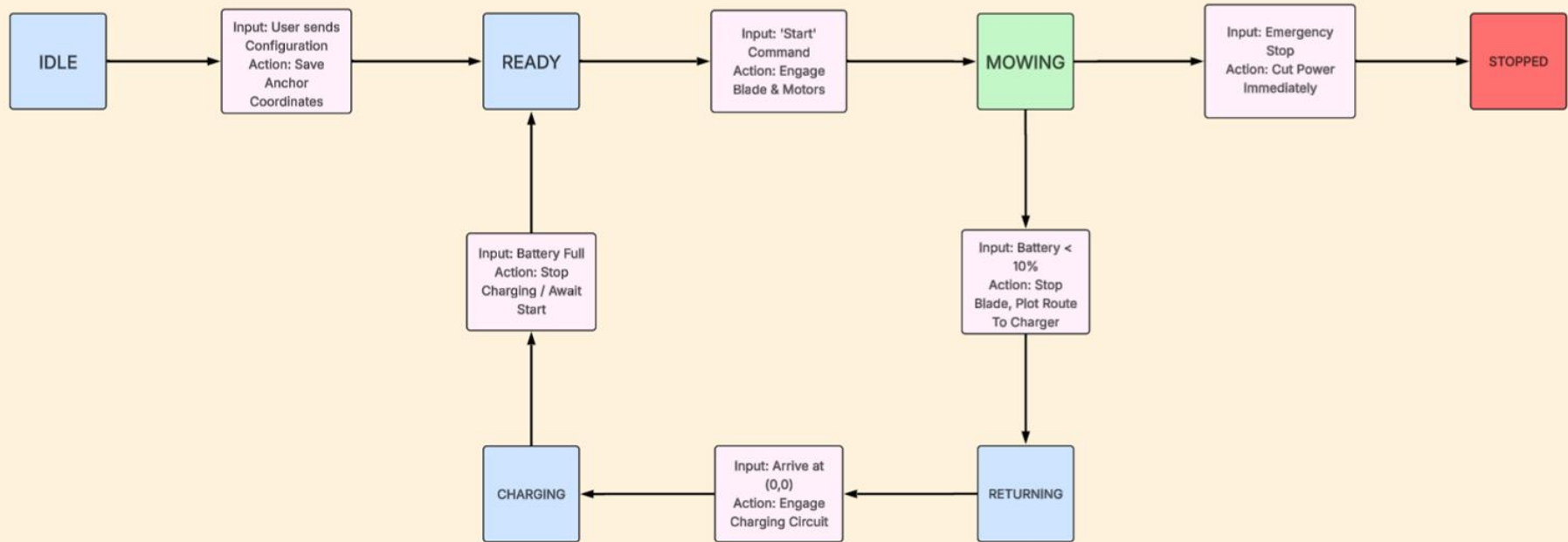


Class Diagram (Mobile App)



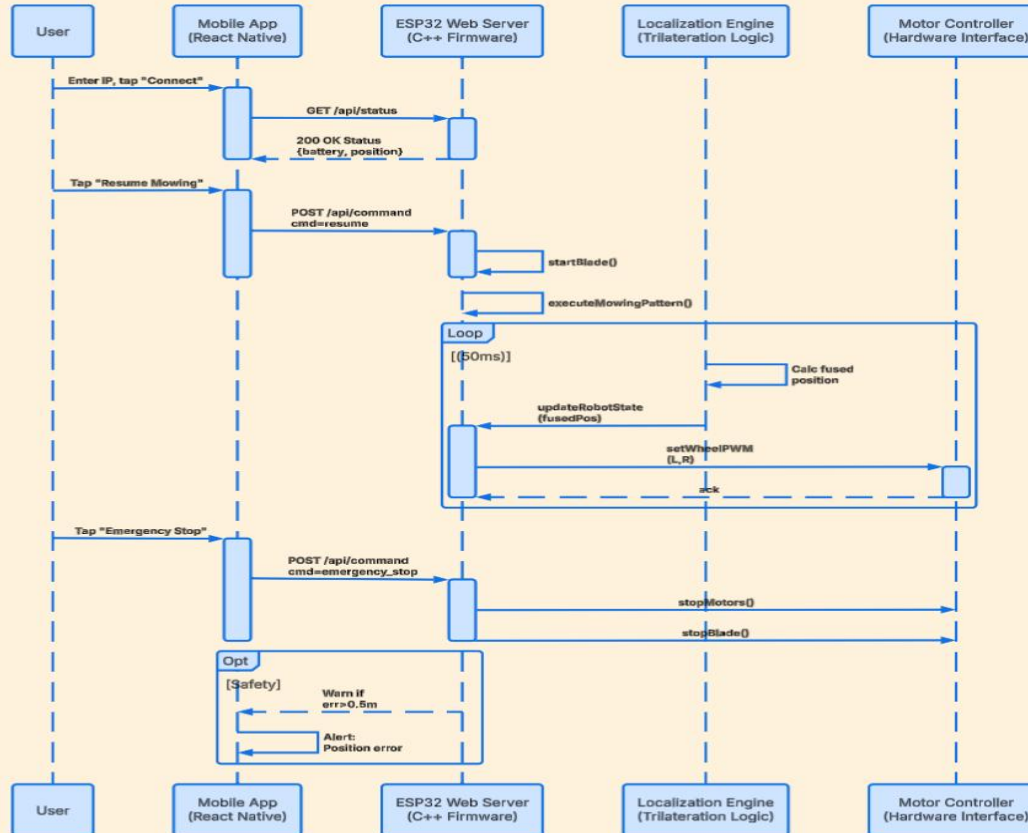


Software Design





Sequence Diagram





Software Test Results

Test Case	Criteria	Result	Notes
API Connection	Loading time < 2 seconds	PASS	Connection to Test Server Confirmed
Emergency Stop	Latency < 50 ms	PASS	Motor Control Stop Confirmed
Position Accuracy	Drift < 0.5 Meters	PASS	Used Dead Reckoning to Confirm Position
APK Install	Installed and Operated on Android Device	PASS	APK Downloaded and Installed
Hotspot Connection	Data Transfer Success	PASS	Tested With Windows Hotspot to Simulate ESP32



Software Selection (Firmware)

Software	Pros	Cons	Selection
ESP-IDF	Native Development Language for ESP32 Advanced RTOS features	Complex Setup Compared to Arduino	Not Selected
C++/Arduino	Direct Hardware Access via GPIO Pins JSON Library Integration Small Memory Footprint	Manual Memory Management Low Level Syntax	Selected
MicroPython	Simple Syntax Rapid Prototype Development	Slow Execution Compared to C++ High Memory Usage Limited Library for ESP32	Not Selected

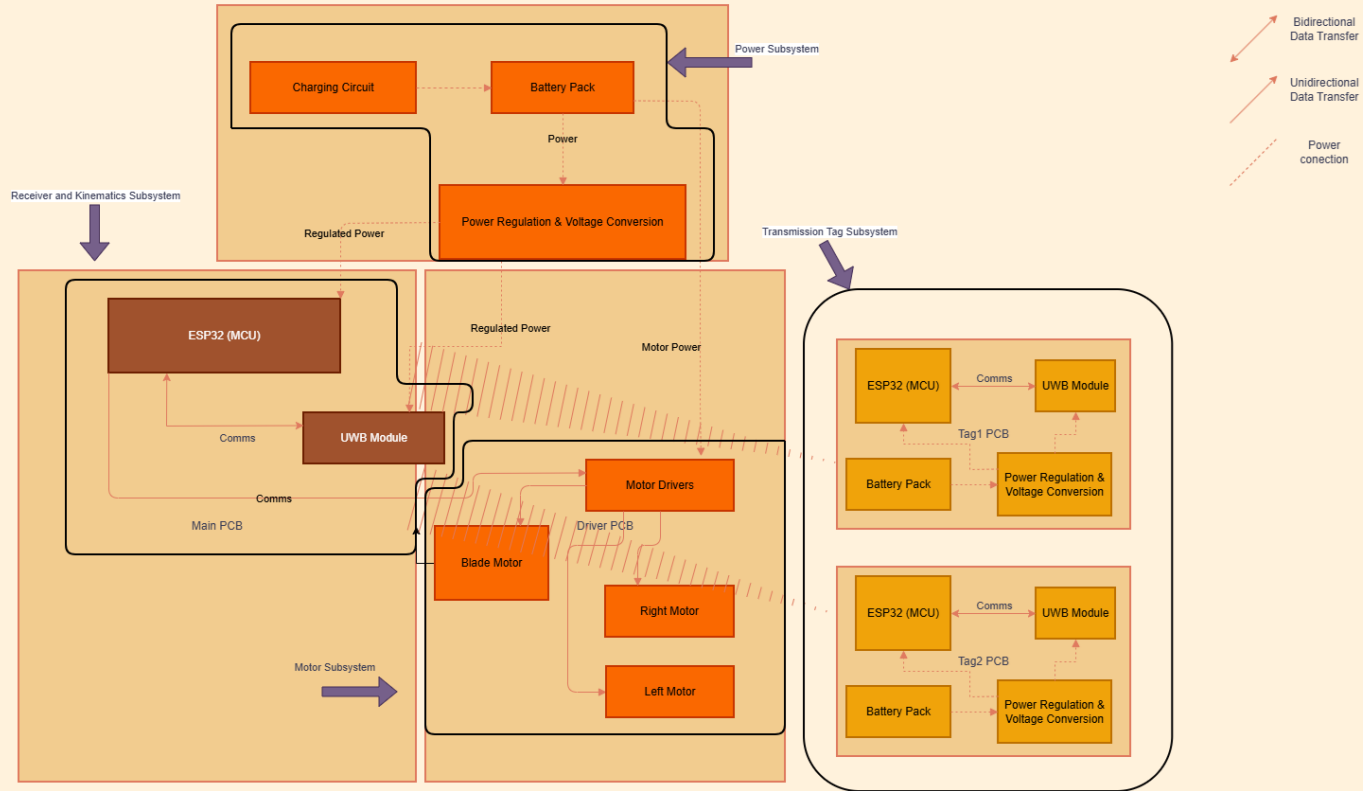


Software Selection (Mobile App)

Software	Pros	Cons	Selection
React Native (ExpoGO)	Single Codebase for iOS and Android Development EAS Build Cloud Development Previous Experience	Dependent on ExpoGO Tools	Selected
Flutter	Better UI Performance Single Codebase for iOS and Android	Larger APK Sizes No Cloud Building	Not Selected



Hardware Design





Logic Processor Considerations



Technology Type	Single-Board Computers (SBCs)	Microcontrollers (MCUs)
Example Device Family	Raspberry Pi	ESP32
Processing Capability	Very High (>1 GHz multi-core CPU)	Moderate (100–300 MHz)
Power Consumption	High (5–10 W typical)	Low (<1 W)
Operating System Support	Full OS (Linux, Windows IoT)	Bare-metal / RTOS (FreeRTOS)
Hardware Integration Complexity	High – requires external RAM, storage, and I/O support	Low – single IC integration
Cost Range (USD)	\$35 – \$100	\$3 – \$15
Real-Time Performance	Non-deterministic scheduling	Deterministic scheduling
Suitability for TURF	Powerful but complex; excessive for simple real-time tasks	Selected – meets cost and real-time requirements



Microcontroller Selection



MCU	ESP32-WROVER-E-N16R8	ESP32-WROOM-32	ATmega328P
CPU / Clock Speed	Dual-core Xtensa @ 240 MHz	Dual-core Xtensa @ 240 MHz	8-bit AVR @ 16 MHz
Flash / RAM	16 MB Flash / 8 MB PSRAM	4 MB Flash / 520 kB SRAM	32 KB Flash / 2 KB SRAM
Wireless Connectivity	Wi-Fi + BLE	Wi-Fi + BLE	None (requires external)
RTOS Support	Native FreeRTOS	Native FreeRTOS	Limited
Approx. Cost (USD)	\$7 - \$8	\$5–6	\$2–3
Strengths	High performance, large memory, multitasking	Proven platform, inexpensive	Low cost, easy to program, mature ecosystem
Limitations	Slightly higher cost	Limited RAM; unsuitable for heavy concurrent tasks	Insufficient performance, no wireless
Suitability for TURF	Selected – meets all performance and integration needs	Acceptable fallback option	Unsuitable



Localization Method Considerations

Parameter	Dead Reckoning	GNSS / GPS	Ultra-Wideband (UWB)
Principle of Operation	Integrates encoder and IMU motion data	Satellite based	Time-of-Flight ranging between fixed anchors and mobile tag
Accuracy	Low (drift accumulates over time)	Moderate (1–3 m typical)	High (<10 cm typical)
Infrastructure Requirement	None	None Requires satellite access	Requires ≥ 3 anchors within range for triangulation
Cost Level	Low	Medium	Medium–High (cost increases with larger areas and more anchors)
Strengths	Simple implementation; low hardware overhead	Wide coverage; global positioning	Centimeter-level precision; reliable short-range positioning
Limitations	Cumulative error; unreliable for long durations	Not precise enough for lawn boundaries	Limited range; requires fixed anchor network
Suitability for TURF	Supplementary only (for short-term estimation)	Unsuitable for fine-grained navigation	Selected – provides required precision for confined fields



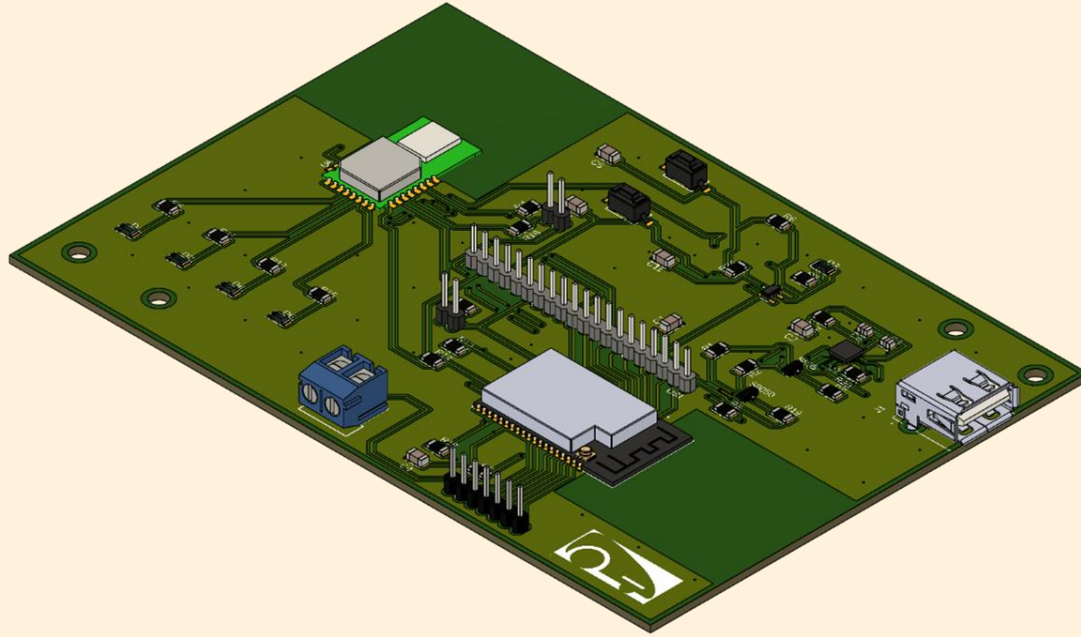
UWB IC Selection

IC	Standard	Accuracy	Power	Cost	Ecosystem Maturity	Suitability
DW1000	IEEE 802.15.4-2011 UWB	10cm	Low	\$9	High	Selected – best cost and community support
DW3000	IEEE 802.15.4z UWB	10cm (greater interference resistance)	Very Low	\$10	Low	Potential for future system upgrades / iterations





Main PCB Design



The Main Board is used for both the robot and for anchors

- Reduces development and Testing time
- Marginally increases costs.





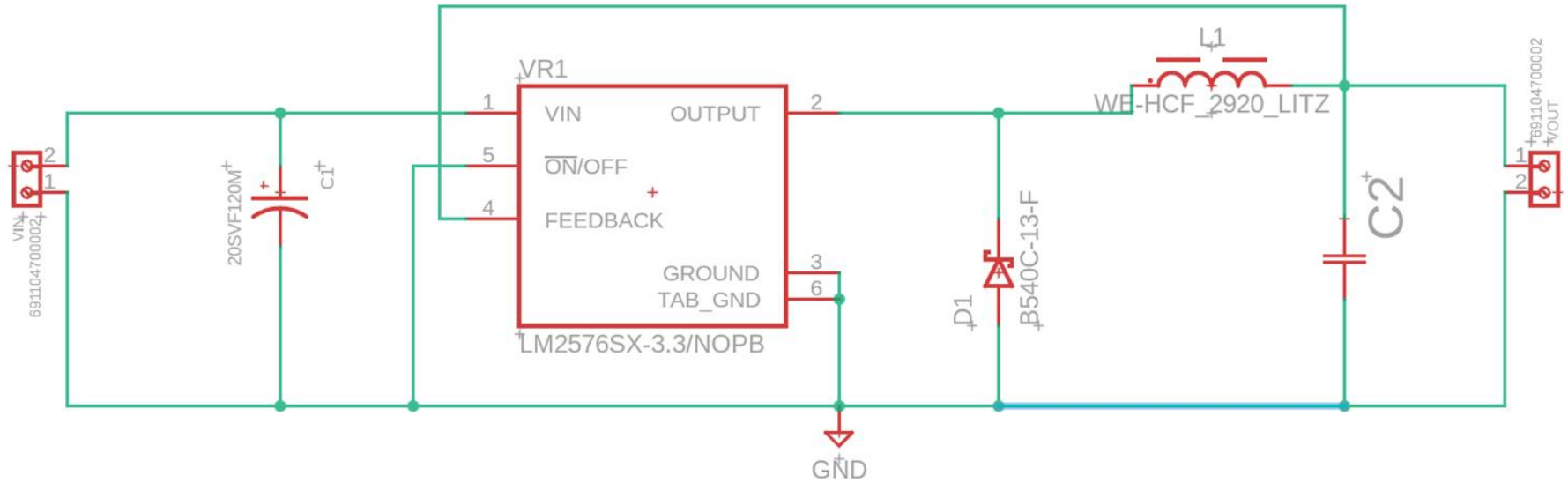
3.3V Regulator Selection

IC	Efficiency	Frequency	Suitability
LMR60440	88.1%	2.15 MHz	Not suitable. Switching Frequency too high.
LM2576	80.5%	52 kHz	Selected for lower switching frequency and better stability



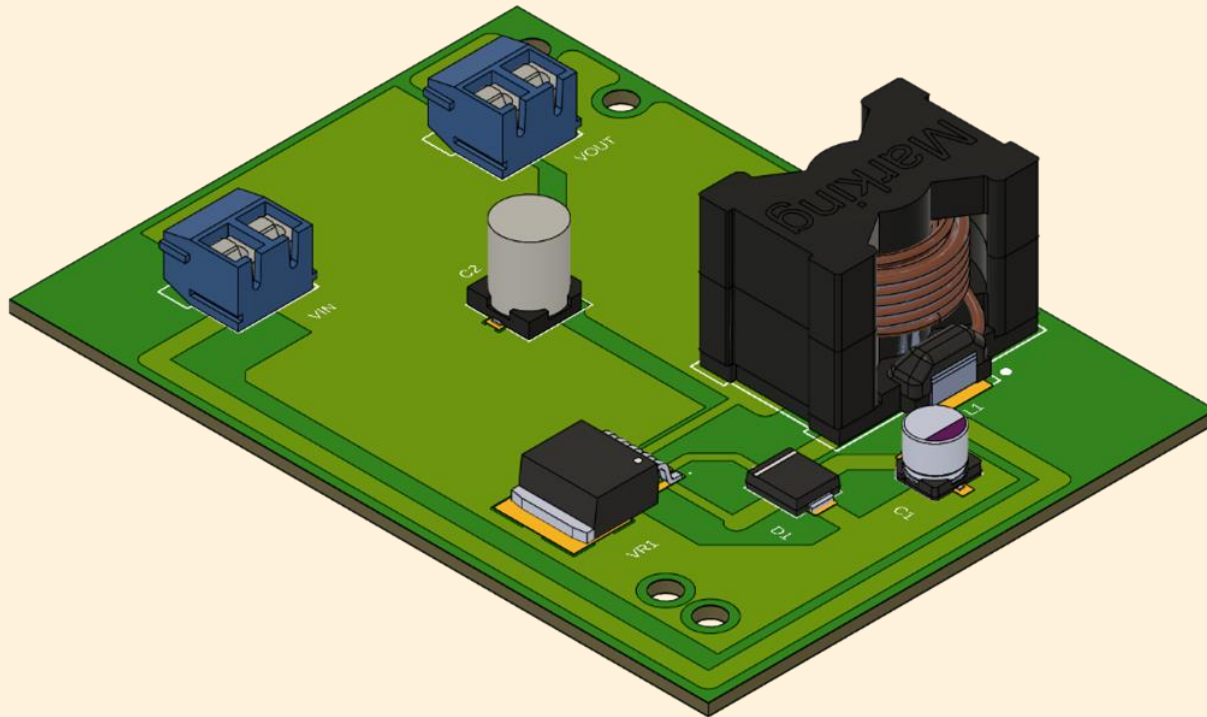


3.3V Regulator Selection





3.3V Regulator Selection





Motor Considerations



Motor Type	Brushed DC Motor	Brushed DC Motor with Hall Sensors	Brushless DC (BLDC) Motor	Stepper Motor
Control Method	PWM (open-loop)	PWM + feedback	Electronic commutation	Step/direction (open-loop)
Feedback Capability	None	Velocity & direction	Integrated (electronic)	Implicit step counting
Strengths	Simple, low cost, high torque	Enables closed-loop control	Efficient, long lifespan	Precise positioning no encoder needed
Limitations	No speed feedback; brush wear over time	Slightly higher cost and wiring complexity	Complex drivers, excessive for low-speed use	Low efficiency, higher current draw
Suitability for TURF	Good baseline but imprecise	Selected. Cost-effective, precision	Suitable for high-end or future expansion	Unsuitable for battery-powered mobile platforms



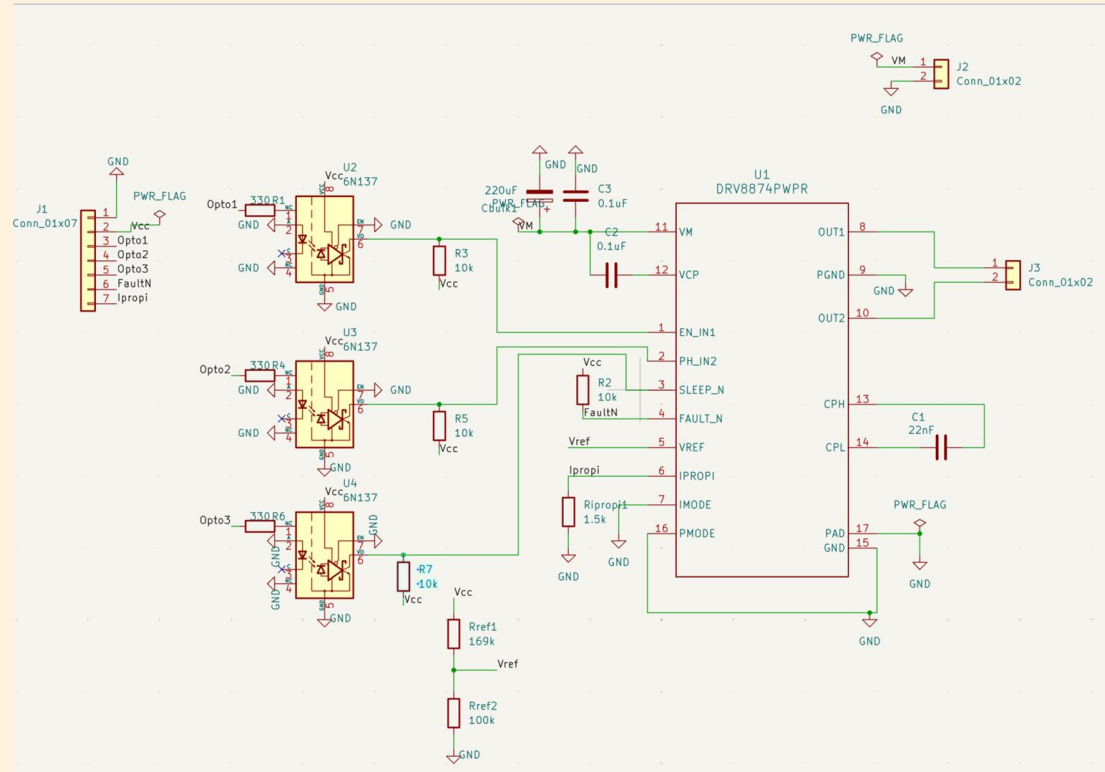
Motor Driver Selection



Feature	DRV8874	L6206
Supply Voltage	4.5–37 V	8–52 V
Control Interface	PH/EN or PWM	IN1/IN2 logic inputs
Protections	UVLO, OCP, TSD, charge-pump UV	OCP, TSD, cross-conduction protection
Efficiency	High (low RDS(on))	Moderate (higher RDS(on))
Integration Level	High	Moderate
Ideal Use	Selected – 9 – 12 V robotics, efficient drive	Higher-voltage

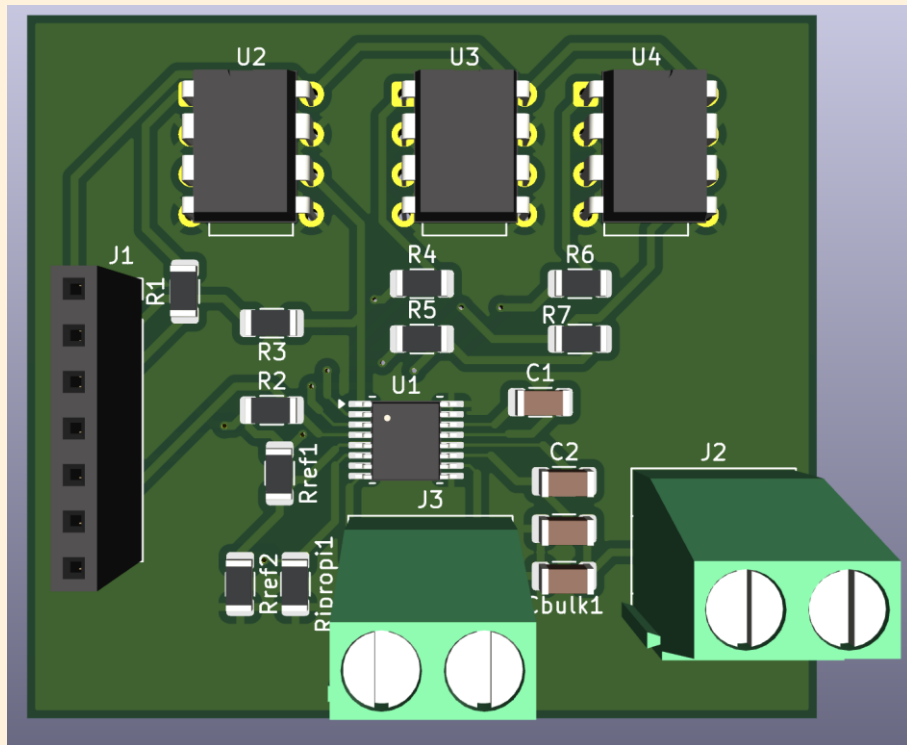


Motor Driver Schematic





Motor Driver PCB Design





Hardware Testing and Performance Evaluation

- PCBs have not been received yet, so testing will be done in the coming weeks.
- Preliminary proof of concept work was done in the previous semester with development boards to test localization feasibility.
- The robot will be evaluated based on its ability to cover a 10m x 10m area without user control.



Budget and Costs

Item	Unit Price	Quantity	Total Cost
ESP32-WROVER-E	\$6.13	4	\$24.52
DWM1000	\$23.62	4	\$94.48
Chassis	\$19.17	1	\$19.17
PCBs (with Shipping)	NA	NA	\$149.85
Miscellaneous Parts	NA	NA	\$186.82
Sub Total (Before Tax and Tariffs)			\$474.84
Total after Taxes and Tariffs			\$512.78

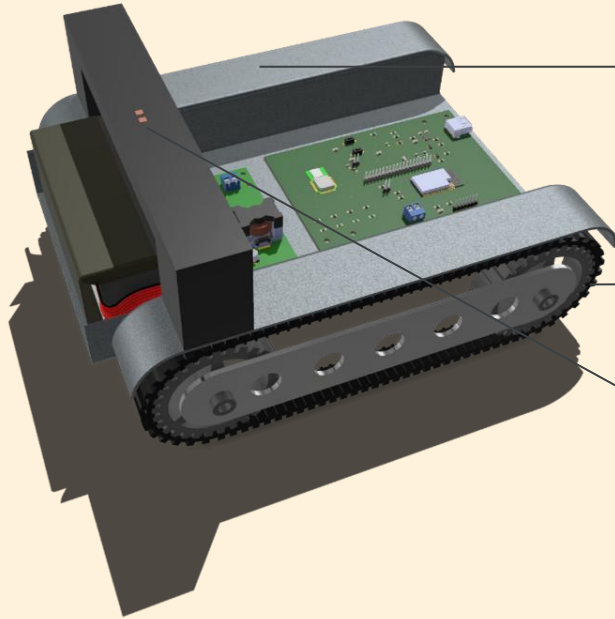
Our Budget is \$1000, with the goal of a final Prototype costing less than \$500.

Current Total Cost reflects the purchase of many spare parts.



Work Distribution

Task	Team Member
Mobile App Development	Samuel Eisert
Microcontroller Firmware	Samuel Eisert Aryan Tulsi
Main Board Development	Aryan Tulsi
Regulator Board Development	Aryan Tulsi
Motor Driver Board Development	Andrew Koch



Premade Chassis

Saves time to build prototype

Tracked Wheels

Improves traction on soil and grass

Charging Contacts

Allows for charging with self docking

Progress and Milestones

