Scope of Report and List of Deliverables

This report presents a comprehensive overview of our group's methodology for designing, manufacturing, and testing a remote-controlled ship capable of collecting floating balls in a pool. Our collaborative efforts, which were conducted during regularly scheduled group meetings, led to the successful development of the ship design. We used an iterative design strategy involving creating and testing multiple prototypes, allowing us to refine and improve our design until it meets our intended objectives. Furthermore, a comprehensive list of milestones and deliverables was established and successfully accomplished throughout the development of our ship.

Milestones	Deliverable
Prototype Creation and Testing	Prototype models made of wood and cardboard, and test results to evaluate their effectiveness and identify design flaws
CAD Model Completion	A precise 3D CAD model created in Fusion 360, detailing all components of the ship design.
Drivetrain Design Completion	Detailed specifications and CAD models for the drivetrain components, including the universal joint.
Communication System	Fully functional remote control system utilizing the Arduino IDE and BLE protocol.
Ship Assembly and Testing	Fully assembled ship, with all components and systems working properly, validated through rigorous testing and adjustments.

System Architecture & Quality Control Process

The ship employs a catamaran hull design as opposed to a single hull design, offering increased stability and reduced water resistance due to its wider stance and minimized wetted surface area. This design choice contributes to enhanced performance and maneuverability, particularly in the context of collecting floating balls in a pool.

By utilizing CAD modeling, we were able to accurately design and manufacture critical components using laser cutting and 3D printing, such as the motor holder and bearing mount, which are integral to our drivetrain system. Our decision to implement a universal joint drivetrain was motivated by the desire to position the motor on the ship's deck rather than inside the hull. This arrangement helps prevent water ingress and simplifies maintenance tasks. For communication, we opted for BLE over Wi-Fi due to its lower power consumption, reduced latency, and sufficient range for our application. This ensures efficient and reliable communication between the remote control and the ship without excessive energy consumption.

We implemented rigorous quality control processes to ensure the reliability and performance of our final product. Comprehensive testing was conducted at each stage of the project. Our test procedures included load testing, and functionality assessments to validate the structural integrity, durability, and performance of individual components and subsystems. Additionally, we conducted in-water tests, simulating various operating conditions to confirm the ship's stability, buoyancy, and ability to collect floating balls effectively. This meticulous approach to quality control, coupled with iterative design refinements, led to the successful creation of a high-quality, efficient, and reliable product that meets our requirements. Our one-to-one testing and requirements table will be provided in the appendix section.

Ship Hull Design & Analysis

The ship hull employed an airfoil-shaped catamaran hull design. The airfoil-shaped hulls offer a hydrodynamic advantage, reducing drag and improving efficiency as the ship moves through the water. And the catamaran configuration, characterized by two parallel hulls connected by a deck, provides a wider base, which significantly enhances the overall stability of the vessel. The shape of the airfoil is chosen from the standard symmetric NACA-6 series airfoil profile. We had laser cut multiple different profiles and tested them in the water, and choosed the one with best performance and stability to implement at our final product design.

We conducted a series of flotation analysis after the manufacture of our ship for assessing our ship's buoyancy, displacement and reserve buoyancy. These analyses allowed us to determine the maximum weight the ship can carry. The detailed test table is provided in the appendix. At the end our ship is able to carry around 500g payloads which is sufficient for us to install the Arduino board and the drivetrain. We also incorporated several design measures to ensure the ship's stability. First, we carefully positioned the ship's center of gravity to be as low as possible, reducing the risk of capsizing due to the heeling moment. Second, we analyzed the distribution of weight along the longitudinal axis of the vessel, ensuring that the ship maintained a level trim, which is crucial for efficient operation and maneuverability. Finally, we designed the hull and deck to include strategically placed compartments and ballast tanks, which could be filled or emptied as needed to adjust the vessel's stability dynamically. Photos of these parts are provided in the appendix. However, the increased mass does affect the propulsion. More of the ship's weight also requires more power to propel the ship through the water, resulting in higher energy consumption and reduced speed.

Eventually we developed a ship hull that has a trim and heel angle within 10 degrees.

Ship Manufacturing & Build

We opted for acrylic as the material for the ship's deck due to its strength, rigidity, and resistance to water, ensuring long-term durability in aquatic environments. We initially used 3mm MDF wooden boards. However, we found that they were not suitable for our needs due to their thinness, which made them prone to breakage. Moreover, wood is known to have disadvantages in aquatic environments, as it can absorb water and become weakened over time. Additionally, we chose foam for the construction of the hull, as it offers excellent buoyancy, lightweight properties, and ease of shaping, which are essential for maintaining stability and maneuverability. The acrylic components of the vessel were manufactured using laser cutting due to its exceptional precision. During the CAD design process, all screw holes were pre-planned and accurately cut simultaneously with the acrylic sheets. Additionally, a multitude of strategically placed holes were incorporated into the ship's deck to enable the adjustment of component positioning to balance the center of gravity. We utilized hot glue and sellotape to create strong and watertight connections between the acrylic deck and foam hulls. This approach facilitated rapid assembly and disassembly for maintenance purposes and provided a reliable bond without compromising the integrity of the materials. The motor holder and bearing mounts in our ship were manufactured using 3D printing technology. This decision was made because these components have a complex and unique shape, making it difficult to manufacture them using traditional tools or laser cutting methods. Also the PLA material is rigid and durable, ensuring long-lasting strength and stability in manufactured products. Additionally, we used traditional tools like pillar drills and scroll saws to make any necessary adjustments and fine-tune the assembly process.

Ship collector design & Analysis
The collector, positioned at the rear of the two hulls, is integrated into a connecting bridge on the ship's deck, which features a slot for the swappable acrylic plate. This design allows for easy installation and removal of the collector, enabling us to experiment with various plate shapes and optimize the system's performance. Photos of the collector design are provided in the appendix.
To determine the most effective collector plate design, we manufactured multiple shapes using acrylic, due to its water resistance and light weight. Our analysis involved assessing each shape's ability to collect balls efficiently while minimizing the impact on the ship's overall performance. The final design features an acrylic board with numerous rectangular holes, which allows water to pass through, reducing drag force and improving maneuverability.

Velocity (m/s)	Steering (degrees)	Type of Holes	Balls Collected
0.2	30	Rectangular	9
0.4	45	Diamond	7
0.3	20	Triangular	9
0.5	60	Rectangular	9
0.2	10	Diamond	8
0.4	15	Rectangular	9

We experimented with various hole shapes in the collector plate. The spherical holes in the collector plate design created problems due to their irregular shape, resulting in decreased efficiency. The diamond-shaped holes caused issues with structural stability and reduced the collector's ability to efficiently collect balls. Similarly, the triangle-shaped holes had a negative impact on water flow and caused turbulence, affecting the ship's performance. After rigorous experimentation and analysis, we concluded that rectangular holes were the optimal choice, offering the best balance between water flow and structural integrity, with the fastest average velocity 0.367m/s. Adding holes on the plate also reduces the weight of the plate, hence creating less effect on the propulsion system.
In conjunction with the extended length of the ship hull, the collector plate design allows for efficient collection of all nine balls. The ball's trajectory is guided directly towards the space between the hull and collector plate, aided by the airfoil shape's front curve. This design ensures that even if the ship rotates or moves, the ball will remain in the designated area, ensuring reliable collection.
The collection of balls inside the ship provides additional benefits beyond efficient storage. By keeping the balls at the center of the ship, the moment caused by their weight is significantly reduced, as the moment arm is shortened. This reduction in moment can have a positive impact on the ship's steering and overall stability, improving its performance in a range of operating conditions.
The addition of the collecting plate had a significant impact on our ship's velocity, yet we achieved a consistently stable outcome with the successful collection of all nine balls in most cases.

Ship Propulsion & Steering Design & Analysis
After careful consideration of various propulsion options, we decided to utilize a water propeller system for our ship. This design allowed us to effectively harness the power of the water, generating greater propulsion and efficiency than air propellers could achieve. Additionally, by placing the propeller under the waterline, we were able to reduce drag and improve the overall performance of the ship. The ship's propulsion system utilizes two 6V DC motors to generate a powerful driving force. The motors are mounted on the ship's deck to prevent water ingress into the foam hull, which would be difficult to waterproof if the motors were placed inside. The drivetrain is designed with multiple universal joints to allow for changes in direction, connecting the top-mounted motors to the propellers located below the hull. To ensure stability, ball bearings are employed to hold the drivetrain firmly in place. An acrylic plate is also added to the bottom of the hull to provide secure screw holes for the bearing mounts.

Velocity(m/s)	Angle of UJ(deg)
0.3	10
0.2	15
0.35	5
0.25	20
0.1	25
0.4	0

The use of universal joints in the ship's propulsion system has a significant impact on velocity, with higher angles leading to decreased speed. However, decreasing the angle of the universal joints requires increasing the length of the rod connecting the top and bottom, which can generate more turning moment and affect the stability of the ship. Therefore, a compromise must be made between speed and stability. Through testing and experimentation, we found that a 20-degree angle for the universal joints offered an optimal balance between velocity and stability. This compromise allowed us to achieve the desired performance and ensure the long-term reliability of the propulsion system.
To enable effective steering, the ship's motors are programmed to operate at varying speeds and directions. We opted not to use a servo motor rudder system due to its instability and slow response time, which would have been impractical for our ship's intended use. Additionally, the plastic connector used in servo motors is often fragile and prone to breaking, making it an unsuitable choice for a high-performance propulsion system. Instead, we programmed the motors to operate in a way that allows for precise control over the ship's direction and speed, while ensuring reliable and durable performance over the long term. However, the selection of the 6V motors presented some challenges. The Arduino Motor Carrier board was not capable of outputting sufficient power to run both motors at maximum speed simultaneously, resulting in reduced propulsion. To optimize the ship's velocity, we had to modify our strategy on the demonstration day, controlling only one motor at a time. Despite this limitation, the ship still performed well and demonstrated the effectiveness of our propulsion system design.

Remote Control System Design & Analysis

We chose BLE over wifi for our ship's communication system due to its lower power consumption, reliable communication over short distances, and greater security against unauthorized access. BLE communication provides quality of service guarantees by utilizing the Generic Attribute Profile (GATT) protocol. GATT defines the way that data is exchanged between devices, and it includes mechanisms for ensuring data integrity and error correction. This means that the BLE communication system can handle issues of data loss or corruption and can recover from errors efficiently.

Compared to wifi, BLE has a lower data rate and shorter range, which can be a disadvantage in some applications. However, BLE is specifically designed for low-power devices and can operate on a small battery for a long time, making it suitable for battery-operated systems like our ship. Additionally, BLE offers better security features than wifi, making it less vulnerable to unauthorized access. This ensures that the motor carrier is not accidentally interfaced with by controllers from other groups. BLE also provides a simpler and more streamlined approach to communication, with fewer technical requirements and simpler protocols.

To implement our code, we used the Arduino IDE and created two files: controller.ino and motorCarrier.ino.

The controller.ino file sets up an Arduino board to communicate with a Bluetooth Low Energy (BLE) device, which acts as a controller. The code initializes the BLE module, sets up a BLE service with three characteristics for the joystick and rotary encoder, and advertises the service for connections. When a device connects, the code reads values from the joystick and rotary encoder, calibrates them to remove any noise or drift, and sends the calibrated values to the connected device through the BLE characteristics. The code also listens for button presses and updates the rotary encoder value to zero when the button is pressed.

The motorCarrier.ino code allows an Arduino board to receive and process BLE signals from a controller. The loop function scans for BLE devices with specified UUID, retrieves joystick and rotary encoder characteristics, and maps the values to control the motor carrier and servo motors. The setup function initializes the serial communication, the BLE module, and the motor carrier, and sets the initial duty cycle and angle values to zero. The loop function continuously reads updated values from the joystick and rotary encoder characteristics and maps them to control the motors and servo motor.The motor carrier is programmed to continuously scan for BLE devices even after a disconnection occurs, thus providing the feature of auto-reconnect.

Debug output is included to print joystick and rotary encoder values to the serial port for testing and troubleshooting.

After compiling and running the control system on our ship's hardware, we found that it works well in terms of fixed rate performance. The control system is able to consistently process and execute commands at a fixed rate, allowing for accurate and precise control of the ship's propulsion and steering. We tested the system under various conditions, including changes in velocity and steering angle, and found that the system responds quickly and accurately to the commands issued by the controller.

We specified a set of performance metrics that we aimed to achieve, including velocity, maneuverability. The test table is presented in the appendix section. All of our metrics are met, meaning our ship is delivering the best performance in terms of communication.

We conducted experiments to determine the optimal gain for the left and right axis of the joystick's effect on the steering propulsion. Using a tuning method, trial and error, we tested different gain values and recorded the resulting data.

Reflection on New Technology Development and Implication for Oceanic Clean-up

Emerging technologies, such as autonomous surface vessels, AI-driven waste collection systems, and advanced materials, have the potential to revolutionize oceanic clean-up efforts, greatly enhancing efficiency and effectiveness. By harnessing these innovations, we can develop scalable and sustainable solutions to combat the immense challenge of marine debris and plastic pollution, which negatively impacts marine life, ecosystems, and human health.

Integration of technologies like machine learning and remote sensing can enable real-time identification and tracking of waste concentrations, allowing for targeted clean-up operations. Moreover, the development of biodegradable and eco-friendly materials can help minimize the environmental footprint of these clean-up activities.

As we continue to explore and implement innovative technologies, it is imperative that we consider the ethical, social, and environmental implications of our actions. We must carefully evaluate potential negative consequences and unforeseen risks, and prioritize sustainable and responsible progress in our quest to preserve the health and vitality of our oceans. This includes engaging in interdisciplinary collaboration, transparent communication with stakeholders, and adhering to best practices and guidelines for environmental conservation and protection.

Reflection on Group Project Delivery and New Lesson Learnt

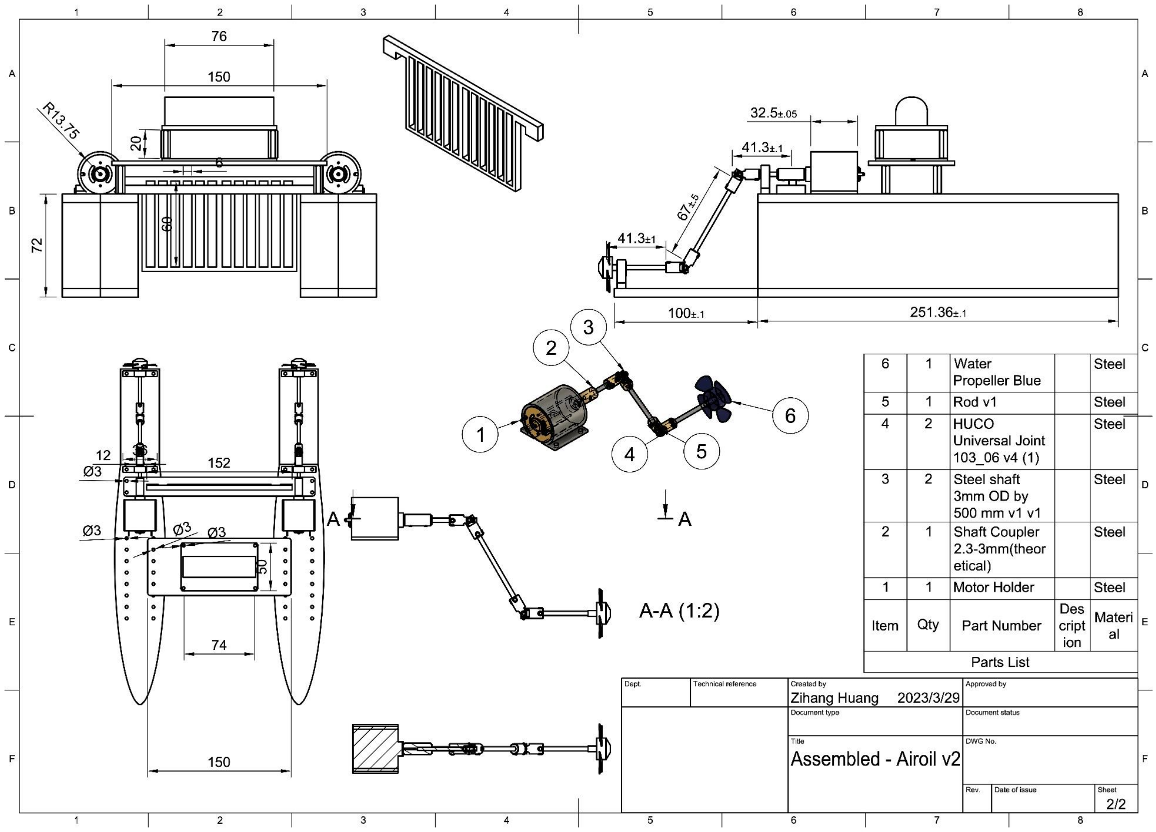
Reflecting on our group project, we recognize the valuable lessons we have learned and the experiences gained through collaboration, communication, and problem-solving. The process of designing, manufacturing, and testing a remote-controlled ship that collects floating balls in a pool proved to be both challenging and rewarding, providing an opportunity for us to apply our engineering knowledge and skills in a real-world scenario. Our project has underscored the importance of interdisciplinary collaboration and the integration of various engineering fields, such as mechanical, electrical, and environmental engineering, to develop holistic and environmentally friendly solutions.

Throughout the project, we faced various obstacles and learned the importance of effective communication and teamwork to address these challenges. By sharing ideas, discussing potential solutions, and delegating tasks, we were able to optimize our design and manufacturing processes. We also discovered the significance of incorporating iterative design and testing to ensure the ship's performance and stability.

In addition, we gained valuable insights into various engineering aspects, such as material selection, manufacturing techniques, and system integration. This experience has broadened our understanding of the complexities involved in engineering projects and the need to consider multiple factors in the decision-making process.

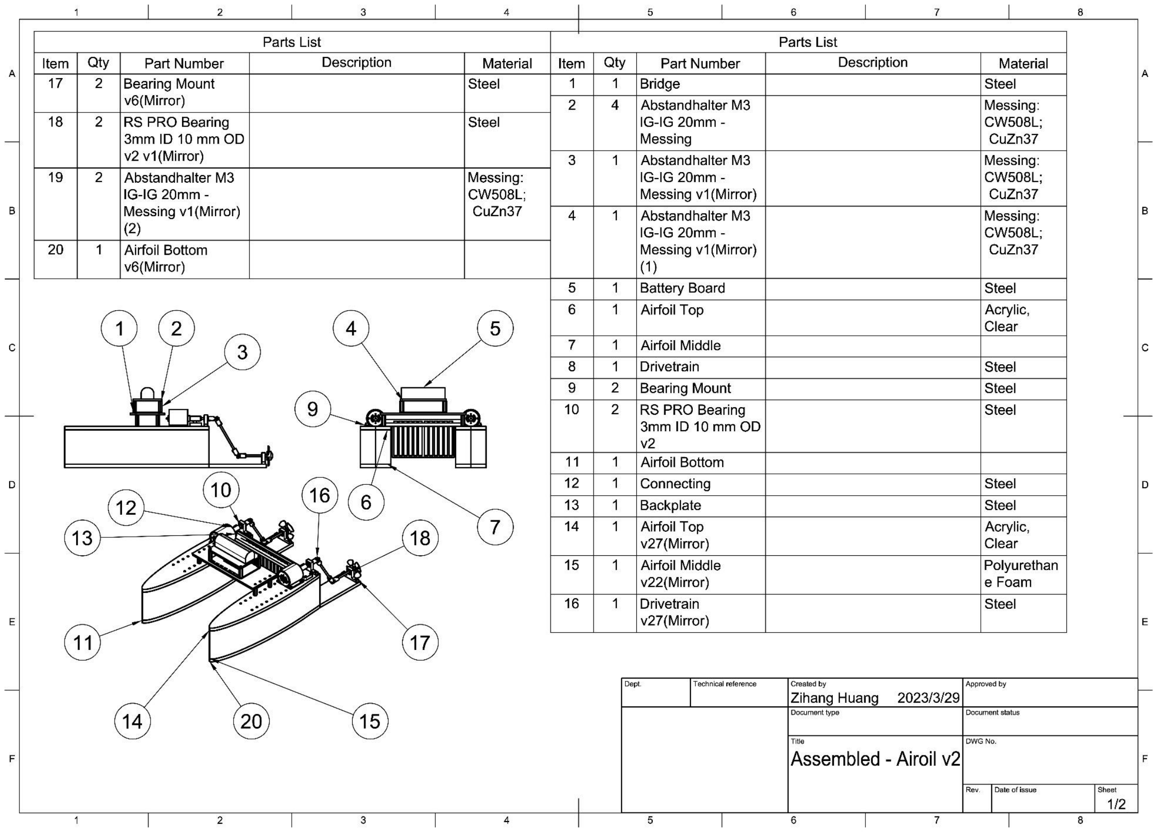
To evaluate our team's performance, we employed a combination of quantitative and qualitative means. Quantitatively, we tracked the performance of each member, such as task completion rates, time spent on tasks, and the number of issues resolved. This data allowed us to objectively measure each member's contributions and identify areas for improvement. Qualitatively, we gathered feedback through regular meetings, design reviews, and one-on-one discussions. This information helped us to tailor our project management approach, leveraging our strengths and addressing any weaknesses.

Appendix



Item	Qty	Part Number	Description	Material
6	1	Water Propeller Blue		Steel
5	1	Rod v1		Steel
4	2	HUCO Universal Joint 103_06 v4 (1)		Steel
3	2	Steel shaft 3mm OD by 500 mm v1 v1		Steel
2	1	Shaft Coupler 2.3-3mm(theoretical)		Steel
1	1	Motor Holder		Steel

Parts List



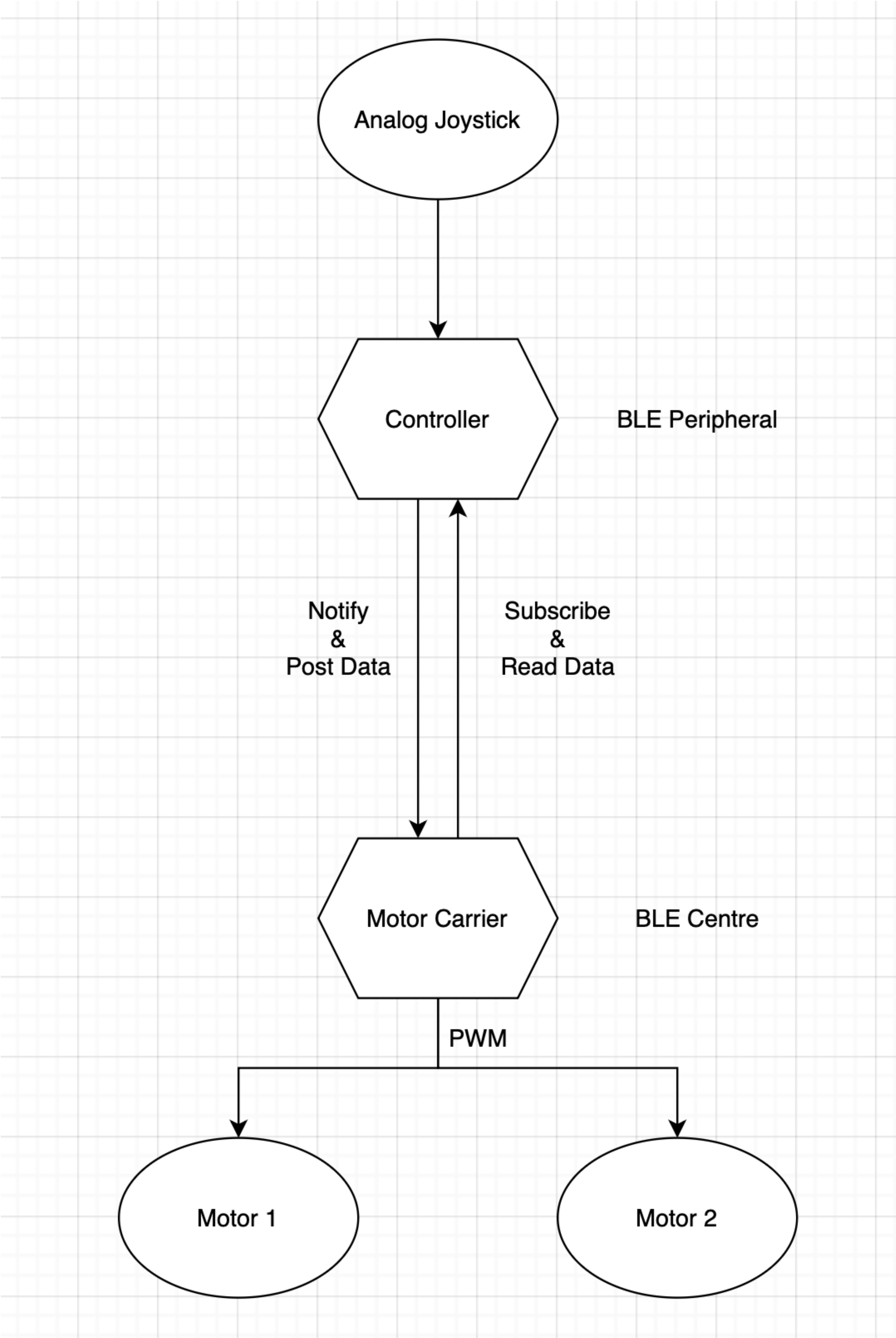
Parts List

Item	Qty	Part Number	Description	Material
17	2	Bearing Mount v6(Mirror)		Steel
18	2	RS PRO Bearing 3mm ID 10 mm OD v2 v1(Mirror)		Steel
19	2	Abstandhalter M3 IG-IG 20mm - Messing v1(Mirror) (2)		Messing: CW508L; CuZn37
20	1	Airfoil Bottom v6(Mirror)		

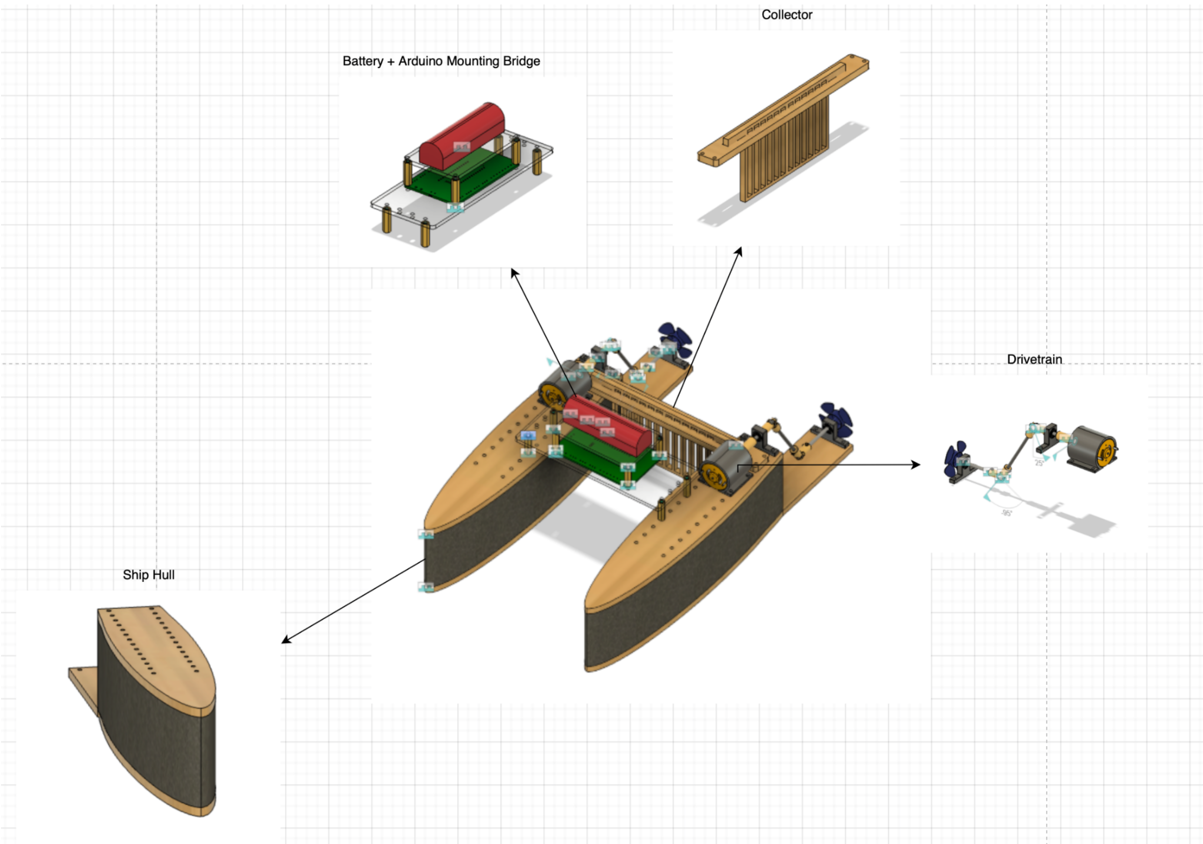
Parts List

Item	Qty	Part Number	Description	Material
1	1	Bridge		Steel
2	4	Abstandhalter M3 IG-IG 20mm - Messing		Messing: CW508L; CuZn37
3	1	Abstandhalter M3 IG-IG 20mm - Messing v1(Mirror)		Messing: CW508L; CuZn37
4	1	Abstandhalter M3 IG-IG 20mm - Messing v1(Mirror) (1)		Messing: CW508L; CuZn37
5	1	Battery Board		Steel
6	1	Airfoil Top		Acrylic, Clear
7	1	Airfoil Middle		
8	1	Drivetrain		Steel
9	2	Bearing Mount		Steel
10	2	RS PRO Bearing 3mm ID 10 mm OD v2		Steel
11	1	Airfoil Bottom		
12	1	Connecting		Steel
13	1	Backplate		Steel
14	1	Airfoil Top v27(Mirror)		Acrylic, Clear
15	1	Airfoil Middle v22(Mirror)		Polyurethane Foam
16	1	Drivetrain v27(Mirror)		Steel

Technical Drawings



Communication Diagram



System Architecture Diagram

```
#include <ArduinoBLE.h>

// define pin numbers for A, B, and button
#define A_PIN 2
#define B_PIN 21
#define BTN_PIN 20
#define UD_PIN A2
#define LR_PIN A3

#define LED_PIN LED_BUILTIN

#define ROTATION_LIMIT 45
#define JOYSTICK_DEADZONE 50

// variables
volatile int16_t rotation = 0;
volatile int16_t udAnalogReading = 0;
volatile int16_t udVal = 0;
volatile int16_t lrAnalogReading = 0;
volatile int16_t lrVal = 0;

const int16_t joystickDefaultVal = 512;

volatile bool sens;
volatile bool rotate;

long previousMillis = 0;
```

```
// BLE
BLEService controlService("e36b42ba-d302-4e6b-9dbd-b48564acaa4d");

BLEIntCharacteristic udCharacteristic("9e0a38ca-a81e-49d1-8f15-5c1b2e6e4abc", BLERead|
BLENotify);
BLEIntCharacteristic lrCharacteristic("76066fb7-6ed0-4995-9781-afec08910176", BLERead|
BLENotify);
BLEIntCharacteristic rotationCharacteristic("ac1326fe-ba8f-4f57-833f-2bc88be33bd5",
BLERead| BLENotify);

void setup() {
 // initialize the serial port
 Serial.begin(19200);

 // set pin modes
 pinMode(A_PIN, INPUT);
 pinMode(B_PIN, INPUT);
 pinMode(BTN_PIN, INPUT_PULLUP);
 pinMode(LED_PIN, OUTPUT);

 // attach interrupt service routines to interrupt pins
 attachInterrupt(digitalPinToInterrupt(A_PIN), isrA, FALLING);

 // BLE
 if (!BLE.begin()) {
   Serial.println("starting Bluetooth(r) Low Energy module failed!");

   while (1);
 }

 BLE.setLocalName("group-h-controller");
 BLE.setAdvertisedService(controlService);

 controlService.addCharacteristic(udCharacteristic);
 controlService.addCharacteristic(lrCharacteristic);
 controlService.addCharacteristic(rotationCharacteristic);

 // add the service
 BLE.addService(controlService);

 udCharacteristic.writeValue(0);
 lrCharacteristic.writeValue(0);
 rotationCharacteristic.writeValue(0);

 // start advertising
 BLE.advertise();

 Serial.println("Bluetooth device active, waiting for connections...");
}

void loop() {
 // BLE
 BLEDevice central = BLE.central();

 if (central) {
   digitalWrite(LED_PIN, HIGH);
```

```
    while (central.connected()) {
      bleUpdate();
    }

    Serial.println("Lost Connection");
    digitalWrite(LED_PIN, LOW);
  }
}

// interrupt service routine for the A
void isrA() {
  if (digitalRead(A_PIN) == HIGH) {
    sens = digitalRead(B_PIN); // Clockwise rotation
  } else {
    sens = !digitalRead(B_PIN); // Anti-clockwise rotation
  }
  rotate = true;
}

void bleUpdate() {
  // Rotation
  if (rotate) {
    if (sens == HIGH) {
      rotation += 2;
    } else {
      rotation -= 2;
    }
    rotate = false;
  }

  if(digitalRead(BTN_PIN) == HIGH){
    rotation = 0; // Return to 0
  }

  // Joystick
  udAnalogReading = analogRead(UD_PIN);
  udVal = -(udAnalogReading - joystickDefaultVal); // Positive UP, Negative DOWN

  lrAnalogReading = analogRead(LR_PIN);
  lrVal = lrAnalogReading - joystickDefaultVal; // Positive RIGHT, Negative LEFT

  // Calibrate Rotation Limit
  if (rotation >= ROTATION_LIMIT) {
    rotation = ROTATION_LIMIT;
  } else if (rotation <= -ROTATION_LIMIT) {
    rotation = -ROTATION_LIMIT;
  }

  // Calibrate Joystick Deadzone (Along Different Axis)
  if (abs(udVal) <= JOYSTICK_DEADZONE) {
    udVal = 0;
  }
  if (abs(lrVal) <= JOYSTICK_DEADZONE) {
    lrVal = 0;
  }
```

```
 /* Calibrate Joystick Deadzone (Only Centre)
 if (abs(udVal) <= JOYSTICK_DEADZONE && abs(lrVal) <= JOYSTICK_DEADZONE) {
   udVal = 0;
   lrVal = 0;
 }
 */
  // BLE Update Value
 bool udChanged = (udCharacteristic.value() != udVal);
 bool lrChanged = (lrCharacteristic.value() != lrVal);
 bool rotationChanged = (rotationCharacteristic.value() != rotation);

 if (udChanged) {
   udCharacteristic.writeValue(udVal);
 }

 if (lrChanged) {
   lrCharacteristic.writeValue(lrVal);
 }

 if (rotationChanged) {
   rotationCharacteristic.writeValue(rotation);
 }

 // Debug Output
 Serial.print("UD = ");
 Serial.print(udVal, DEC);
 Serial.print(", LR = ");
 Serial.print(lrVal, DEC);
 Serial.print(", Rotation = ");
 Serial.println(rotation);
}
```

Controller.ino

```
#include <ArduinoMotorCarrier.h>

#include <ArduinoBLE.h>

// variables
volatile int16_t rotation = 0;
volatile int16_t udVal = 0;
volatile int16_t lrVal = 0;

static int duty = 0;
static int dutyAdjust = 0;

static int M4duty = 0;
static int M2duty = 0;

static int angle = 0;

void setup() {
 Serial.begin(9600);
```

```
 // BLE
 BLE.begin();

 Serial.println("BLE Initialized");

 BLE.scanForUuid("e36b42ba-d302-4e6b-9dbd-b48564acaa4d");

 // Motor Carrier
 controller.begin();

 M3.setDuty(0);
 M4.setDuty(0);

 servo1.setAngle(0);

 Serial.println("Setup Complete");
}

void loop() {
 // check if a peripheral has beenn discorvered
 BLEDevice peripheral = BLE.available();

 if (peripheral) {
   // discovered the peripheral
   Serial.print("Found ");
   Serial.print(peripheral.address());
   Serial.print(" ");
   Serial.print(peripheral.localName());
   Serial.print("' ");
   Serial.print(peripheral.advertisedServiceUuid());
   Serial.println();

   // prevent coincidence
   if (peripheral.localName() != "group-h-controller") {
     return;
   }

   // Stop scanning
   BLE.stopScan();

   motorControl(peripheral);

   // peripheral disconnected, start scanning again
   BLE.scanForUuid("e36b42ba-d302-4e6b-9dbd-b48564acaa4d");
 }
}

void motorControl(BLEDevice peripheral) {
 Serial.println("Connecting...");

 if (peripheral.connect()) {
   Serial.println("Connected");
 } else {
   Serial.println("Failed to connect!");
   return;
 }
```

```
 // discover peripheral attributes
 Serial.println("Discovering attributes ...");
 if (peripheral.discoverAttributes()) {
   Serial.println("Attributes discovered");
 } else {
   Serial.println("Attribute discovery failed!");
   peripheral.disconnect();
   return;
 }
 // retrieve characteristics
 BLECharacteristic udCharacteristic =
peripheral.characteristic("9e0a38ca-a81e-49d1-8f15-5c1b2e6e4abc");
 BLECharacteristic lrCharacteristic =
peripheral.characteristic("76066fb7-6ed0-4995-9781-afec08910176");
 BLECharacteristic rotationCharacteristic =
peripheral.characteristic("ac1326fe-ba8f-4f57-833f-2bc88be33bd5");

 if (!udCharacteristic | !lrCharacteristic | !rotationCharacteristic) {
   Serial.println("Characteristic list doesn't match");
   peripheral.disconnect();
   return;
 }

 udCharacteristic.subscribe();
 lrCharacteristic.subscribe();
 rotationCharacteristic.subscribe();

 while (peripheral.connected()) {
   byte buf[4];
   if (udCharacteristic.valueUpdated()) {
     udCharacteristic.readValue(buf, 4);
     udVal = (int16_t)((buf[1] << 8) | buf[0]);  // Convert Bytes to integer.
     duty = map(udVal, -512, 512, 100, -100);
   }

   if (lrCharacteristic.valueUpdated()) {
     lrCharacteristic.readValue(buf, 4);
     lrVal = (int16_t)((buf[1] << 8) | buf[0]);
     dutyAdjust = map(lrVal, -512, 512, 100, -100);
   }

   M4duty = -(duty + dutyAdjust);
   M2duty = -(duty - dutyAdjust);

   if (M4duty > 100){
     M4duty = 100;
   }
   if (M2duty > 100) {
     M2duty = 100;
   }
   if (M4duty < -100) {
     M4duty = -100;
   }
   if (M2duty < -100) {
     M2duty = -100;
```

```
  }

  M4.setDuty(M4duty);
  M3.setDuty(M2duty);

  if (rotationCharacteristic.valueUpdated()) {
    rotationCharacteristic.readValue(buf, 4);
    rotation = (int16_t)((buf[1] << 8) | buf[0]);

    servo1.setAngle(rotation);
  }

  // Debug Output
  Serial.print("UD = ");
  Serial.print(udVal, DEC);
  Serial.print(", LR = ");
  Serial.print(lrVal, DEC);
  Serial.print(", Rotation = ");
  Serial.print(rotation);
  Serial.print(", M4 = ");
  Serial.print(M4duty);
  Serial.print(", M2 = ");
  Serial.println(M2duty);
 }
 Serial.print("Connection Terminated");
}
```

MotorCarrier.ino

Team	*Work Package*	*Requirement*	*Test*	*Test date*	*Passed*
Systems & Quality Control (A)	*· Specify system architecture* *· Report on tests to be verified by design teams* *· Demonstrate project at end of term to collect 9 floating objects in under 2 min*	*A1. Ship should collect 9 floating objects in under 2 min*	*A1a. Manual testing of ship prototype MK1 for demonstration day scenario* *A1b. Manual testing of ship prototype MK2 for demonstration day scenario* *A1c. Manual testing of ship final before demonstration day scenario* *A1d. Manual testing of ship final on demonstration day scenario*	*A1a. Before first design review, March 6* *A1b. Before second design review, March 20* *A1c. Before demo day, March 27* *A1d. On demo day, March 27*	*A1a. Passed, 30sec* *A1b. Passed, 40 sec* *A1c. Passed, 20 sec* *A1d. Passed, 30 sec*

Ship Design & Build (B)	· *Ensure ship flotation* · *Ensure ship stability* · *Design & manufacture ship hull* · *Design & manufacture housing for on-board equipment and propulsion system* · *Waterproof on-board equipment*	*B1. Ship waterline must be below 80% of ship height and carry a payload that exceeds by the weight of all ship components (c. 600g)* *B2. Ship trim must be less than 10% of ship length* *B3. Ship heel must be less than 10% of ship width* *B4. Ship hull must not deform (bending) more than 5 mm per 100 mm span of material compared to when unloaded* *B5. Ship hull must securely accommodate electronic boards* *B6. Ship hull must securely accommodate batteries* *B7. Electronic components must be safe from water ingress and splashing* *B8. Motor housings must securely accommodate the respective motor bodies* *B9. Motor housing should attenuate excessive vibration that can compromise other components, e.g. loosen fasteners, or detach adhesive layers* *B10. Internal ship components should be safe from water ingress for more than 10 minutes*	*B1. Measure the waterline of the ship when it is fully loaded with the payload and empty of all components. Then measure the height of the ship and calculate the percentage of the water line to the total height. Compare the measured data with the requirement of 80% or lower. Also measure the weight of all ship component and the payload to compare them. The weight of the payload should exceed the weight of all ship components.* *B2. Draw a level line in front, middle and back of the ship and comparing the readings with the level surface, the maximum allowed trim should be less than the minimum requirement of 10% of the ship length.* *B3. Draw a level line in front, middle and back of the ship and comparing the readings with the level surface, the maximum allowed heel should be less than the minimum requirement of 10% of the ship width.* *B4. Applying different weights on top of the ship on different position. And measure how much it bends to compare with the minimum requirement 5mm per 100 mm span of material.* *B5a. Using the Inspect function from Fusion360 to measure the position and size of screw holes. And check whether it accommodates with the Electronic boards' CAD model.* *B5b. Assemble the electronic board onto the ship prototype to check whether it accommodates.* *B6. Using the Inspect function from Fusion360 to measure the position and size of screw holes. And check whether it accommodates with the batteries' CAD model.* *B7. Place the ship model in water tank for 2 minutes to check whether the electronic components will contact water.* *B8. Install the motors into the motor housing, and check that the motors are securely fastened and properly aligned. Also check that the motor housing provide adequate protection for the motors and the motors do not make contact with any other components when in use.* *B9. Measure the vibration levels of the motors while they are running, both with and without the motor housing installed. Compare the vibration levels to evaluate the motor housing in attenuating excessive vibration. Also inspect the adjacent ship components for any signs of loosening or detachments during the vibration test.* *B10.Test the watertightness of all the internal ship component by simulating a water ingress scenario by putting the ship inside a water tank, also poured the water into the ship at different locations for a duration of 10 minutes. After that, test the functionality of all the internal components to ensure that they have not been damaged.*	*B1. After the final prototype is manufactured and before the demo day, March 27.* *B2. After the final prototype is manufactured and before the demo day, March 27.* *B3. After the final prototype is manufactured and before the demo day, March 27.* *B4. After the final prototype is manufactured and before the demo day, March 27.* *B5a. Every time when the prototype CAD design is completed. In every group review.* *B5b. After the final prototype is manufactured and before the demo day, March 27.* *B6a. Every time when the prototype CAD design is completed. In every group review.* *B6b. After the final prototype is manufactured and before the demo day, March 27.* *B7. After the final prototype is manufactured and before the demo day, March 27.* *B8. After the final prototype is manufactured and before the demo day, March 27.* *B9. After the final prototype is manufactured and before the demo day, March 27.* *B10. After the final prototype is manufactured and before the demo day, March 27.*	*B1. Pass - The ship waterline is below 80% of ship height and carries a payload that exceeds the weight of all ship components (c. 600g).* *B2. Pass - The ship trim is less than 10% of ship length.* *B3. Pass - The ship heel is less than 10% of ship width.* *B4. Pass - The ship hull does not deform (bending) more than 5 mm per 100 mm span of material compared to when unloaded.* *B5. Pass - The ship hull securely accommodates electronic boards.* *B6. Pass - The ship hull securely accommodates batteries.* *B7. Pass - Electronic components are safe from water ingress and splashing.* *B8. Pass - Motor housings securely accommodate the respective motor bodies.* *B9. Pass - Motor housing attenuates excessive vibration that can compromise other components, such as loosening fasteners or detaching adhesive layers.* *B10. Pass - Internal ship components are safe from water ingress for more than 10 minutes.*

Mechanical Systems (C)	*· Design motor to propeller mechanism for adequate propulsion* *· Design adequate floating waste, collection system* *· Design adequate steering system, e.g. rudder or other control surfaces* *· Design calibration system* *· Deploy control algorithm to microcontroller*	*C1. Motor-to-shaft connection must be secure at the max rotation speed of the motor** *C2. Shaft-to-propeller connection must be secure at the max rotation speed of the shaft** **Warning: motor-to-shaft and shaft-to-propeller tests must be conducted in a safe and controlled environment. Make sure you wear appropriate PPE and ensure that unsecure connections do not damage surrounding equipment or people.* *C3. Ship collector system must securely connect to ship hull* *C4. Ship collector system must collect floating objects* *C5. Ship collector system must retain floating objects after collection* *C6. Motors must spin due to an operator command* *C7. Rudder must turn due to an operator command* *C8. Ship should move in a straight line at a minimum steady state speed of 0.1 m/s* *C9. Ship should reach max steady-state linear speed in under 1 s* *C10. Ship should steer with a minimum turning radius of 0.1 m at the steady state speed* *C11. Ship should turn in place at a minimum steady state angular velocity of 30 rpm* *C12. Ship should reach max steady-state angular velocity in under 3 s*	*C1. Gradually run the motor up to the maximum speed and measure the torque of the motor-to-shaft connection at different speed. Compare it to the maximum torque specification of the motor. Also check the connection is secure and does not loosen at maximum speed.* *C2. Gradually run the motor up to the maximum speed and measure the torque of the shaft-to-propeller connection at different speed. Compare it to the maximum torque specification of the motor. Also check the connection is secure and does not loosen at maximum speed.* *C3. Inspect the connection between the ship collector system and the ship hull to verify that collector system is secure when ship moves around in the water.* *C4. Test the ship collector system in different controlled environment imitating the demonstration day scenario. Inspect its ability for collecting floating object in water.* *C5. Test the ship collector system in different controlled environment imitating the demonstration day scenario. Inspect its ability for retaining the collected object after collection.* *C6. Verify that the motors respond to operator commands to spin after the remote control.* *C7. Verify that the rudders respond to operator commands to spin after the remote control.* *C8. Test the ship's movement in different controlled environment imitating the demonstration day scenario. Let the ship move in a straight line at a steady state speed of 0.1m/s and compare the ship's movement to a reference line to check for deviation.* *C9. Test the ship's movement in different controlled environment imitating the demonstration day scenario. Measure the ship's acceleration and deceleration time and compare it with the minimum requirement 1s.* *C10. Measure the turning radius of the ship at steady state speeds in different scenarios and compare the turning radius of the ship against the minimum requirement of 0.1m.* *C11.Measure the angular velocity of the ship when turning in place at steady state speeds by placing it inside a water tank and compare the angular velocity of the ship against the minimum requirement of 30rpm.* *C12. Measure the time it takes for the ship to reach its maximum steady-state angular velocity from still state by placing it inside a water tank and compare the time against the minimum requirement of under 3 seconds.*	*C1. After the final prototype is manufactured and before the demo day, March 27.* *C2. After the final prototype is manufactured and before the demo day, March 27.* *C3. After the collector system is designed and manufactured before the last group review, March 20.* *C4. After the collector system is designed and manufactured before the last group review, March 20.* *C5. After the collector system is designed and manufactured before the last group review, March 20.* *C6. After the collector system is designed and manufactured before the last group review, March 20.* *C7. After the collector system is designed and manufactured before the last group review, March 20.* *C8. After the final prototype is manufactured and before the demo day, March 27.* *C9. After the final prototype is manufactured and before the demo day, March 27.* *C10. After the final prototype is manufactured and before the demo day, March 27.* *C11. After the final prototype is manufactured and before the demo day, March 27.* *C12. After the final prototype is manufactured and before the demo day, March 27.*	*C1. Pass - The motor-to-shaft connection is secure at the max rotation speed of the motor.* *C2. Pass - The shaft-to-propeller connection is secure at the max rotation speed of the shaft.* *C3. Pass - The ship collector system securely connects to the ship hull.* *C4. Pass - The ship collector system successfully collects floating objects.* *C5. Pass - The ship collector system retains floating objects after collection.* *C6. Pass - Motors spin in response to operator commands.* *C7. Pass - The rudder turns in response to operator commands.* *C8. Pass - The ship moves in a straight line at a steady state speed of 0.1 m/s.* *C9. Pass - The ship reaches its maximum steady-state linear speed in under 1 second.* *C10. Pass - The ship steers with a minimum turning radius of 0.1 m at the steady-state speed.* *C11. Pass - The ship can turn in place at a minimum steady-state angular velocity of 30 rpm.* *C12. Pass - The ship reaches its maximum steady-state angular velocity in under 3 seconds.*

Control & Communication (D)	· *Design remote controller and operator interface* · *Deploy remote control communication, including server-client setup using Wi-Fi or BLE* · *Design steering controller to ensure adequate ship manoeuvrability* · *Implement low-level controllers for DC motor(s)* · *Manage sample times and latency for control and communication*	*D1. Transmitter-receiver connection must be established* *D2. Transmitter-receiver should use a unique communication line* *D3. Controller must be able to receive commands from an operator* *D4. Controller must be able to send commands to a transmitter* *D5. Transmitter must be able to send commands to a receiver* *D6. Receiver must be able to actuate a component* *D7. Motors must stop spinning in case of extended loss of communication (say more than 3 s)* *D8. Failed message must not cause an excessive jolt in components* *D9. Motors must only spin when commanded by operator* *D10. Servomotors must not be moved beyond the maximum unrestrained operation angle positions** *Warning: Continuously forcing a servomotor against a restraint will cause overheating and permanent damage to the motor* *D11. Operator-to-actuator response time should be under 1 s* *D12. Operator-to-actuator messages should be successfully transmitted 95% of the time*	*D1. Sending a test signal from the transmitter to the receiver to verify their connection is established.* *D2. Checking the frequency or channel of the signal is unique by researching the availability of radio spectrum. Also test the transmitter and receiver's ability to transmit and receive data accurately in real time with a stable connection under different scenarios. Test the ability of transmitter and receiver for encrypting and decrypting data through the communication line.* *D3. Sending a test signal from the operator to controller to verify that the controller is able to receive commands from operator in real time.* *D4. Sending a test signal from controller to transmitter to verify that the transmitter is able to receive commands from controller in real time.* *D5. Sending a test signal from transmitter to receiver to verify that the receiver is able to receive commands from transmitter in real time.* *D6. Test whether the Arduino Nano 33 IoT can control the boat components. Provide a control input and ensure the corresponding motor spins.* *D7. Force disconnects the remote control to simulate the scenario of losing the communication. Measure the time for the motors' to response and comparing it with the minimum requirement of 3 seconds.* *D8. Intentionally sending a failed message to the receiver and observing the response of different components to test the system's ability to handle failed messages in real time.* *D9. Observe the movement of the motors when no commands are given. Sending unauthorized commands and observe the movement of the motors.* *D10. Observe the servomotors' movement during operations. Keep sending control signals and check whether will servomotors move beyond the maximum unrestrained operation angles.* *D11. Measure and record the time delay between when an operator command is given and when the corresponding actuator responds. The time should be within 1s.* *D12. Sending a large number of commands and measure the percentage of successful transmissions. The percentage should be higher than 95%.*	*D1. After the communication system is designed and manufactured before the last group review, March 20.* *D2. After the communication system is designed and manufactured before the last group review, March 20* *D3. After the communication system is designed and manufactured before the last group review, March 20* *D4. After the communication system is designed and manufactured before the last group review, March 20* *D5. After the communication system is designed and manufactured before the last group review, March 20.* *D6. After the communication system is designed and manufactured before the last group review, March 20.* *D7. After the communication system is designed and manufactured before the last group review, March 20.* *D8. After the communication system is designed and manufactured before the last group review, March 20.* *D9. After the communication system is designed and manufactured before the last group review, March 20.* *D10. After the communication system is designed and manufactured before the last group review, March 20.* *D11. After the communication system is designed and manufactured before the last group review, March 20.* *D12. After the communication system is designed and manufactured before the last group review, March 20.*	*D1. Pass - Transmitter and receiver are successfully connected.* *D2. Pass - Communication line between the transmitter and receiver is unique and functioning properly.* *D3. Pass - The controller can receive commands from the operator without any issue.* *D4. Pass - The controller can successfully send commands to the transmitter.* *D5. Pass - The transmitter can successfully send commands to the receiver.* *D6. Pass - The receiver can actuate the component without any issue.* *D7. Pass - Motors stop spinning after an extended loss of communication (more than 3 seconds).* *D8. Pass - Failed messages do not cause any excessive jolt in the components.* *D9. Pass - Motors only spin when commanded by the operator as expected.* *D10. Pass - Servomotors do not move beyond the maximum unrestrained operation angle positions.* *D11. Pass - Operator-to-actuator response time is under 1 second, meeting the requirement.* *D12. Pass - Operator-to-actuator messages are successfully transmitted 95% of the time, meeting the requirement.*

Test Plans

Test No.	Ship Weight (g)	Added Payload (g)	Total Weight (g)	Displacement (cm³)	Reserve Buoyancy (cm³)	Result
1	500	50	550	570	20	Stable
2	500	100	600	630	30	Stable
3	500	150	650	660	10	Marginally stable
4	500	200	700	680	-20	Unstable
5	500	250	750	680	-70	Unstable

Buoyancy test

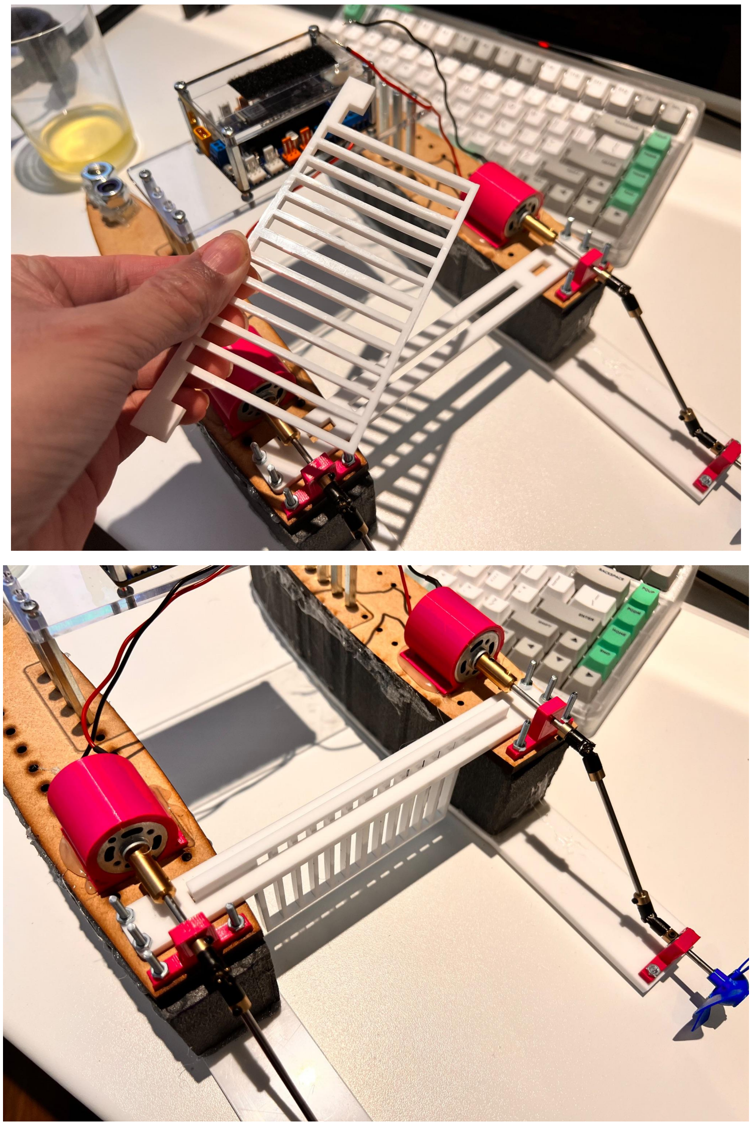
Test No.	Metric	Target Value	Measured Value	Result
1	Max Forward Velocity (m/s)	1.5	1.55	Met
2	Max Reverse Velocity (m/s)	0.75	0.78	Met
3	Turning Radius (m)	1.0	0.95	Met
4	Max Angular Velocity (°/s)	45	48	Met
5	Signal Range (m)	30	32	Met
6	Latency (ms)	100	80	Met

Communication & Control Test

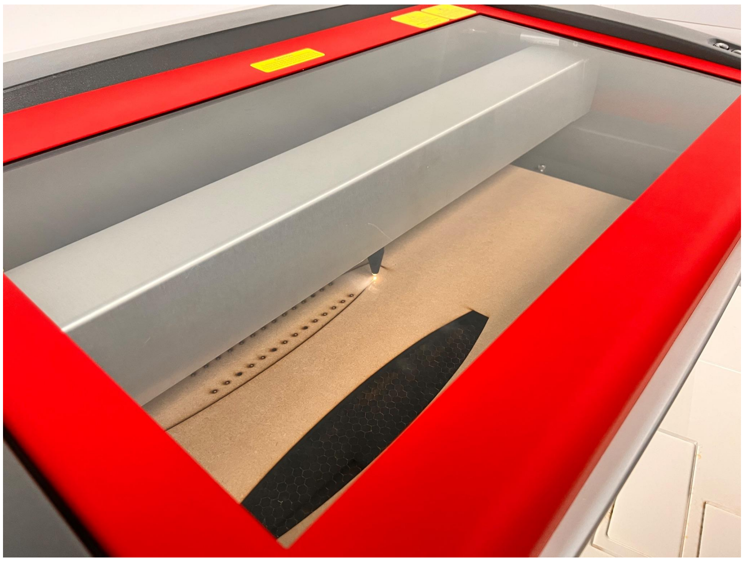
Requirement No.	Description	Category
1	Catamaran hull design for stability	Hull Design
2	Lightweight, durable materials (acrylic, foam)	Material Selection
3	Watertight seals and connections	Manufacturing
4	Modular design for component adjustments	Design Flexibility
5	Adequate buoyancy and payload capacity	Flotation Analysis
6	Collector design optimization and testing	Collection System
7	Risk mitigation strategies for sinking	Safety

8	Performance monitoring and optimization	Performance Analysis

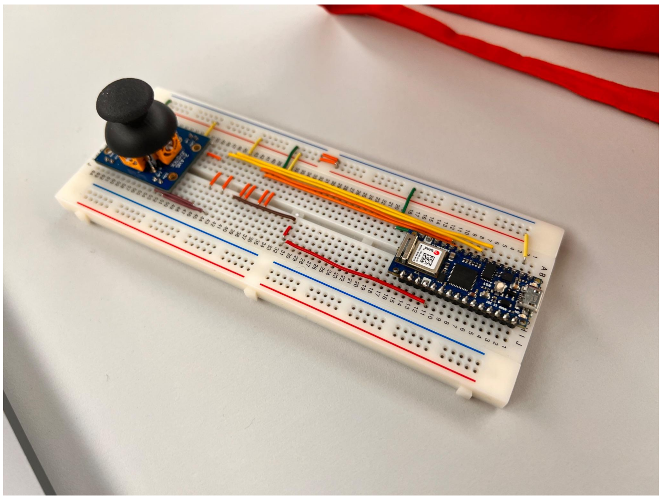
Requirements passed to the Design Team



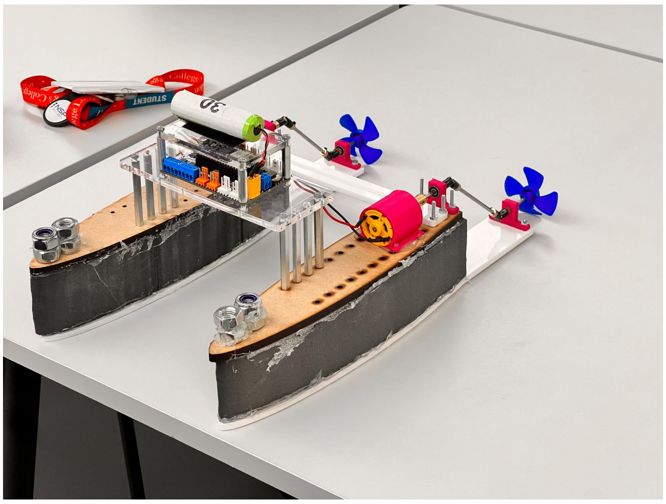
Collector Design



Laser cutting the Deck



Controller Design



Final Build of Our Ship