8.1 Introduction Section

The purpose of this test plan is to ensure that every software component—from low-level sensor input modules to the central control and telemetry systems—meets its functional, performance, and safety requirements. The efficiency of the Unity version of TrackSide will also be compared to the Go version. This plan is designed to validate that the integrated system operates reliably under real-world conditions while adhering to the rigorous standards demanded by a motorsports application. As emphasized in the FSAE Software Development Plan, extensive testing and debugging are critical for ensuring both performance and safety.

8.2 Unit Test Plan

1. Trackside Unit Tests
    a. Run TrackSide
    b. Use keyboard for inputs
    c. Check to make sure UI is responsive and if telemetry data is being displayed properly
    d. Run TrackSide tester (replicates a car sending data in real time)
    e. Check to see if the data is being updated in real time and whether the data matches the metrics
    f. Ensure TrackSide can run on the targeted hardware (Raspberry Pi 5)
2. Vehicle Electronics Unit Tests
    a. Load in the testing data (examples of inputs with expected outputs)
    b. Run the testing program (will be an executable .exe)
    c. See terminal to see whether tests pass or not
3. Motor Safety Unit Tests
    a. Load in the testing data (examples of inputs with expected outputs)
    b. Run the testing program (will be an executable .exe)
    c. See terminal to see whether tests pass or not
4. Sensor Unit Tests
    a. Load in the testing data (examples of inputs with expected outputs)
    b. Run the testing program (will be an executable .exe)
    c. See terminal to see whether tests pass or not

8.3 Integration Test Plan

Integrate individual components (e.g., Input Observer, Data Management, Traction Control) to verify that sensor data is accurately transmitted through the Central Control Unit (CCU) to the Telemetry GUI. Special attention will be paid to maintaining minimal latency (e.g., under 100 ms for GUI updates and under 50 ms for database transmissions) and ensuring data integrity across all modules.

8.4 Acceptance Test Plan

Pre-recorded telemetry data will be fed to the TrackSide app. All functions of the TrackSide app will be tested and functionality will be assessed.

An Arduino will be given fake sensor data and its output will be printed to a computer and validated.

8.5 Test Configuration Control

Testing will be conducted on target hardware platforms, including Raspberry Pi 5 units (serving as the CCU), multiple Arduino Uno boards (for sensor data simulation), and development machines running both Windows and Linux. The test environment will utilize tools such as the Arduino IDE, GNU C++ Compiler, Visual Studio Code, and PostgreSQL for telemetry data management.

8.6 Items Not Tested

No items will be excluded from testing. All components and functionalities have been included in the test plan to ensure a comprehensive evaluation of the system.

8.7 Test Verification Matrix

A detailed test verification matrix will be maintained to map each test case to its corresponding software requirement. This matrix will serve as a systematic record ensuring that all functionalities, performance metrics, and safety requirements are verified throughout the testing process.

Trackside Unit Tests	5.4.1.1.1 The Telemetry GUI shall display data with less than 100ms latency 5.4.1.1.2 The Telemetry GUI shall support simultaneous display of up to 10 data metrics 5.4.1.2.1 The Database shall handle data transmission with a latency of less than 50ms 5.4.1.2.2 The Database shall store telemetry data at a rate of 10 samples per second 5.4.1.3.1 The Raspberry Pi’s CPU usage while running all pertinent software should not go above 90%
Vehicle Electronics Unit Tests	5.4.5.2.1 Vehicle Stability CSC should be able to calculate when traction is lost within 10 ms 5.4.5.2.2 Vehicle Stability CSC should be able to calculate the motor power distribution within 50ms 5.4.5.2.3 Vehicle Stability CSC should be able to calculate PID within 50 ms.

Motor Safety Unit Tests	5.4.5.5.1 Be able to ensure that all signals are within expected range and there are no errors or unexpected latency
Sensor Unit Tests	5.4.3.1.1 Consistently be able to report steering and pedal inputs to the Central Control Unit 5.4.4.1.1 Consistently display all pertinent information to the driver through an easily navigable UI 5.4.4.2.1 Be able to hold a consistent wifi connection to the Central Control Unit and communicate with it 5.4.4.2.2 Consistently generate display for the Graphics Interface 5.4.5.1.1 The Data Management CSC should be receiving the data within 10 ms 5.4.5.6.1 Consistently be able to report how much power the whole vehicle is using