Functional Requirements

- **User Input:** The system shall accept user parameters (target range, strike angle, restitution coefficient) via the GUI.
- **Pressure Calculation:** Compute and recommend the pneumatic pressure needed for a given target distance and angle.
- **Simulation Control:** Upon user command, initiate a Gazebo physics simulation of the ball strike with specified angle and pressure.
- **Physics Prediction:** Calculate the expected projectile distance using a physics model (taking into account gravity and restitution).
- **Result Display:** Show both predicted and actual landing distances (and other metrics) to the user.
- **Data Logging:** Record all launch parameters and results (predicted vs actual distances, timestamps) for later review.
- **Configuration Persistence:** Load and save experiment or calibration data (e.g. prior runs) as needed by PressureRecommender or DataLogger.
- **Error Handling:** Validate inputs and warn the user if parameters are out of range or invalid.

Non-Functional Requirements

- **Modularity:** Each node/component must be loosely coupled and focused on a single responsibility, as per ROS2 best practices. This ensures easy maintenance and testing.
- **Real-Time Performance:** The system should respond to user inputs and display simulation results with minimal latency; the simulation itself should run in (simulated) real time.
- **Usability:** The GUI must be clear and intuitive, allowing users to enter parameters and understand results easily.
- **Testability:** Key components (e.g. PhysicsCalculator, PressureRecommender) should be unit-testable with known inputs/outputs. The design allows each node to be tested in isolation.

- **Scalability/Compatibility:** The system should run on standard ROS2 platforms (compatible with Gazebo), and be extensible (e.g. adding new ball models or sensor feedback).
- **Reliability:** The system must handle invalid inputs gracefully and log all operations for troubleshooting.