TRAQUA TEST PLAN

Rieke Platthaus | Maria Włodarczyk | Inès Margand | Guillem Vázquez Rolduá | Maximilian Salmi | Bernardo Alves
`

Bernardo Correia Alves
[Email address]

Contents

No table of contents entries found.

Introduction

1.1 Purpose

This document describes the testing strategies and methodologies used by the TRAQUA group to evaluate smart water bottle system.

The purpose of this document is to ensure the proper operation of the system, as well as to identify and fix any defects or problems that may arise during the development and deployment process.
In addition, this document provides a detailed description of the testing methods and strategies used throughout the testing process.

Chapter 2: Criteria

This chapter establishes the standards required for proper test execution and examines conditions that may undermine the reliability of test outcomes.

2.1 Entry Criteria

Use Case Documentation Availability

- The various steps, scenarios and expected outcomes are available and approved by stakeholders

Test Environment Readiness

The environment required to execute the tests is prepared and configured to mirror the production environment. This includes:

• The latest prototype/version of the code in the "main" branch of the GitLab repository “backend”

• A personal computer with windows 10 or 11 or MacOS operating system and a functioning keyboard, mouse, display, and internet connection

• React native 0.80

• Expo

Test Data Preparation

• Valid and invalid data sets are prepared as per use case specifications

Test Case Development

• Test cases outline the steps, expected outcomes, and the preconditions for test execution

Test Team Preparedness

• Test team is prepared and capable of executing test case

2.2 Exit Criteria

Test Case Execution Completion

◦ All identified test cases that cover the relevant testing methods have been done successfully.

Functional Correctness Validation

◦ The system should be able to correctly execute the steps outlined in the test cases and outcomes should match the expected results.

Defect Resolution

◦ Any defects or bugs should be recorded and resolved

◦ Ready for User Acceptance Testing (UAT)

◦ Test Report Generation

◦ Test reports summarize the test execution and should include coverage and deviation from expected behaviour.

2.3 Suspension Criteria

- Scope Change
- If testing moves forward faster than the continuous development of the product.
- Critical defects
- If the base product is unable to run on a testing device, or the state of the product
- would impact the validity of the test results.
- External dependencies are unavailable
- Resources are unavailable
- If the necessary resources for the test environment, test data, or personnel are
- unavailable the testing may need to be suspended until resources are available
- If most critical tests fail

Chapter 3: Resources

This chapter lists all the pols and resources that are employed by the project team to manage, keep track of, and execute tests.

- Test Case Creation: Github
- Test Case Tracking: Github
- Test Case Execution: Manual
- Test Case Management: Github
- Defect Management: Github
- Test Reporting: PDF/Microsoft Word, Postman
- Checklist Creation: Github
- Unit Testing: PyCharm, piplines
- Code running: Device itself

Chapter 4 Test strategies

This chapter outlines the overall approach and guiding principles for testing the TRAQUA bottle. It defines the scope and objectives of testing, the specific types of testing the configuration of the test environment, and how the severity of bugs is assessed. The goal is to ensure system reliability, safety, and alignment with operational requirements before deployment.

4.1 Scope

This only applies to testing the application. The main scope is testing the functionality of the system.

4.2 Objectives

Verifying the functionality, precision, and dependability of the smart bottle. The testing procedure is intended to be comprehensive and scenario-driven.

Functional Testing:

• Objective:

  o Validate the overall functionality of the system

• Rationale:

  o Make sure the system works as expected

4.3 Types of testing

The testing strategy for TRAQUA is pretty simple. It focuses on state tests .

ate Transition Testing — Traqua Smart Bottle

This method tests the system's behavior based on transitions between defined water-safety states: **Safe**, **Warning**, **Unsafe**, and **Unknown**. It ensures that the bottle correctly classifies water quality based on confidence levels derived from pH, TDS, and temperature readings, and that appropriate system actions (e.g., triggering on-device LED and OLED alerts, updating the mobile app's safety chip, blocking UVC activation, and logging hydration events) are taken depending on the state.

The verification covers the full sensing → classification → actuation chain, including the BLE link between the ESP32 firmware and the React Native app. The Unknown state is exercised by simulating poor sensor quality (stale readings, BLE disconnects, malformed JSON payloads) to confirm the system fails safe rather than reporting a stale Safe verdict.

4.5 Scope

The scope of testing for the Traqua smart-bottle system covers the full sensing-to-display chain: ESP32 firmware, the BLE link, and the React Native mobile app. Each item below is included to ensure water-quality classification is accurate, hydration tracking is consistent, and on-device safety interlocks behave as specified.

- Testing will cover the classification accuracy of the water-quality logic (Safe, Warning, Unsafe, Unknown) against synthetic and real sensor inputs

- Testing will include the handling of poor-quality or invalid sensor data such as BLE packet loss, malformed JSON payloads, and stale readings
- Testing will cover state transitions driven by pH, TDS, and temperature thresholds, and the alert mechanism that follows them (LED, OLED, in-app safety chip)
- Testing will verify the UVC safety interlock — the UVC light must never energise unless the magnetic reed switch is closed
- Testing will examine the interaction between the ESP32 firmware, the BLE characteristic, and the useBluetoothConnection hook in the app, including reconnection after disconnect
- Testing will validate hydration-tracking logic: drink detection, daily-goal progress, and the 12-hour idle reset
- Testing will be performed on Android (primary target) with parity checks on iOS where applicable
- Testing will include both unit tests for individual modules (water-safety scoring, BLE payload parser) and integration tests across the firmware ↔ app boundary

Test Principles

- Tests are delivered at the end of each sprint for stakeholder review
- Phased testing keeps code validity under control as features are added
- Every test has a documented rationale so peers and stakeholders can follow it
- Testing environments emulate real bottle use as closely as possible (real ESP32, real BLE, physical sensors)
- Testing is repeatable, quantifiable, and measurable
- Tests are mapped to the sprint content they cover
- Each test has explicit entry and exit criteria

Bug severity is explained in our wiki documentation.

Test cases

Test case ID	**Initial State**	**Input condition**	**Expected State**	**Expected Action**
TC01	Any	Connect to the ESP32	Log result	Connect successful
TC02	Any	Wrong id in connection	Log result	Connection not succefull
TC03	Any	Get TDS value	Log result	Tds value to be within 50 to 200 value
TC04	Any	Get Temperature value	Log result	Read temp value in the app
TC05	Any	Gravity sensor	Log result	Read value of if the bottle is tilted or not
TC06	Any	Led light changes color	Log result	Led either green or red
TC07	Any	Pressure sensor estimated the water	Log result	Read total water in bottle
TC08	Any	Ufv light to run	Clean water	Water to be cleaned
TC09	Any	Red switch	UVc to turn on or off	When the switch is connected the light should not run