System Test Plan

For

Bluetooth Protocol Analytical Research

Team Members: Matthew Irvin, Carolina Terre, Jonah Rowell,
Connal Grace and Gianna Scarangelli

Version/Author	Date
v1	10/18

Table of Contents

1. Introduction

1.1 Purpose

This document is a test plan for Bluetooth Protocol Analytical Research System Testing, produced by the System Testing team. It describes the testing strategy and approach to testing the team will use to verify that the application meets the established requirements of the business prior to release.

1.2 Objectives

- Meets the requirements, specifications, and Business rules.
- Supports the intended business functions and achieves the required standards.
- Satisfies the Entrance Criteria for User Acceptance Testing.

2. Functional Scope

The Modules in the scope of testing for the Bluetooth Protocol Analytical Research System Testing are mentioned in the document attached in the following path:

1. The System Requirements Specification Document
2. Section 3.1 of this Document

3. Overall Strategy and Approach

3.1 Testing Strategy

The Bluetooth Protocol Analytical Research System Testing will include testing of all functionalities that are identified in scope (Refer to Functional Scope Section). System testing activities will include the testing of new functionalities, modified functionalities, screen level validations, workflows, functionality access, and testing of internal & external interfaces.

3.2 System Testing Entrance Criteria

In order to start system testing, certain requirements must be met for testing readiness. The readiness can be classified into: Usability Testing and Functional Testing.

3.3 Testing Types

3.3.1 Usability Testing

User interface attributes, cosmetic presentation and content will be tested for accuracy and general usability. The goal of Usability Testing is to ensure that the User Interface is comfortable to use and provides the user with consistent and appropriate access and navigation through the functions of the application (e.g., access keys, consistent tab order, readable fonts etc.)

System Requirements Specification. 3.1.1 Arduino IDE: “Arduino IDE will be used to push code to the ESP32.”

System Requirements Specification. 3.1.2 Zigbee Software: “Zigbee Development software will be used to control the Zigbee Boards in order to send a signal.”

System Requirements Specification. 3.1.3 “HackRF Tools: HackRF Tools will be used to control the HackRF One.”

System Requirements Specification. 3.1.4 “GNU Radio: GNU Radio will be used to send RF signals via the HackRF One. Users can configure what frequency and range the signals are being

sent as well as view Bluetooth packets and signals.

3.3.2 Functional Testing

The objective of this test is to ensure that each element of the component meets the functional requirements of the business as outlined in the:

System Requirements Specification. 3.4.2 “The ESP32 modules will communicate via Bluetooth.”

System Requirements Specification. 3.4.3 “The Zigbee modules will communicate via Zigbee.”

System Requirements Specification. 3.3.1 “Five Raspberry Pis will be loaded with Kali Linux (or other software) in order to run their respective hardware.”

System Requirements Specification. 3.3.2 “HackRF Tools will be used in order to establish a connection with the HackRF One.”

System Requirements Specification. 3.3.3 “GNU Radio will be used in order to generate and send RF signals using the HackRF.”

System Requirements Specification. 3.3.4 “Data for the ESP32 and Zigbee Boards will be sent via serial from the Raspberry Pis.”

3.4 Suspension Criteria and Resumption Requirements

3.4.1 Suspension Criteria

Testing will be suspended if the incidents found will not allow further testing of the system/application under test. These incidents may include disruption of power, Bluetooth, data and other functionalities that we require between each device in use. Any anomaly in function will be treated as a reason for suspension. If testing is halted, and changes are made to the hardware, software, or database, it is up to the Testing Manager to determine whether the test plan will be re-executed or part of the plan will be re-executed.

3.4.2 Resumption Requirements

Resumption of testing will be possible when the functionality that caused the suspension of testing has been retested successfully. All test cases must be successfully completed again to confirm the test bed has regained full functionality. This includes ensuring all bluetooth communication, RF signals, and data transfer amongst system components are now functional.

4. Execution Plan

4.1 Execution Plan

The execution plan will detail the test cases to be executed. The Execution plan will be put together to ensure that all the requirements are covered. The execution plan will be designed to accommodate some changes if necessary, if testing is incomplete on any day. All the test cases of the projects under test in this release are arranged in a logical order depending upon their inter dependency.

Requirement (From System Requirements Specification)	Test Case Identifier	Input	Expected Behavior	Pass/Fail
3.1.1 Arduino IDE: “Arduino IDE will be used to push code to the ESP32.	1.1	Code written in Arduino IDE	ESP32 transmits/receives data according to the bit stream generated by the Arduino code	
3.1.2 Zigbee Software: “Zigbee Development software will be used to control the Zigbee Boards in order to send a signal.”	2.1	Code written in Zigbee Development software	Zigbee signal is transmitted	
3.1.3 “HackRF Tools: HackRF Tools will be used to control the HackRF One.”	3.1	Code written using HackRF Tools	HackRF receives transmitted data	
3.1.4 “GNU Radio: GNU Radio will be used to send RF signals via the HackRF One.	4.1	Data given to GNU Radio	HackRF receives the data	
3.1.4 GNU Radio can configure what frequency and range the signals of the data being transmitted	4.2	Desired frequency and range of transmission	Data transmitted at the desired frequency and range	

3.1.4 GNU can vie Bluetooth packets as signals	4.3	Bluetooth packets being received	Received data being read as signals	
3.4.2 “The ESP32 modules will communicate via Bluetooth.”	5.1	Data transmitted from one Bluetooth module	Another Bluetooth module receives the data	
3.4.3 “The Zigbee modules will communicate via Zigbee.”	6.1	Data transmitted from one Zigbee module	Another Zigbee module receives the data	
3.3.1 “Five Raspberry Pis will be loaded with Kali Linux (or other software) in order to run their respective hardware.”	7.1			
3.3.2 “HackRF Tools will be used in order to establish a connection with the HackRF One.”	8.1	Code written using HackRF Tools	Connection established with HackRF One	
3.3.3 “GNU Radio will be used in order to generate and send RF signals using the HackRF.”	9.1	Desired data to be transmitted	Desired data transmitted using the HackRF	
3.3.4 “Data for the ESP32 and Zigbee Boards will be sent via serial from the Raspberry Pis.”	10.1	Data from the Raspberry Pi	The ESP32/Zigbee board receives the data via serial	

5. Traceability Matrix & Defect Tracking

5.1 Traceability Matrix

List of requirements, corresponding test cases.

5.2 Defect Severity Definitions

Critical	The defect causes a catastrophic or severe error that results in major problems and the functionality rendered is unavailable to the user. A manual procedure cannot be either implemented or a high effort is required to remedy the defect. Examples of a critical defect are as follows: ● System abends ● Data cannot flow through a business function/lifecycle ● Data is corrupted or cannot post to the database
Medium	A defect that does not seriously impair system function can be categorized as a medium defect. A manual procedure requiring medium effort can be implemented to remedy the defect. Examples of a medium defect are as follows: ● Form navigation is incorrect ● Field labels are not consistent with global terminology
Low	The defect is cosmetic or has little to no impact on system functionality. A manual procedure requiring low effort can be implemented to remedy the defect. Examples of a low defect are as follows: ● Repositioning of fields on screens ● Text font on reports is incorrect

6. Environment

6.1 Environment

- The System Testing Environment will be used for System Testing.

The Bluetooth System will be composed of the following devices
- Raspberry Pi 4
- ESP 32 Development Board with Integrated Bluetooth

The Zigbee System will be composed of the following devices
- Raspberry Pi 4
- Zigbee Development Board

The Software Defined Radio System will be composed of the following devices
- Raspberry Pi 4
- HackRF One

7. Assumptions

- All forms of interaction with bluetooth would be done under the 5.0 Standard
- The devices will only be interacting with each other and nothing not specifically specified in the environment.

8. Risks and Contingencies

During the Bluetooth Protocol Analytical Research System Testing, potential hardware malfunctions, especially with devices like the Raspberry Pi or Zigbee Development Board, are a concern. To address this, it's crucial to pre-test all components and have backups on hand. Software anomalies on these devices could also introduce errors. Using reputable software versions and preliminary software checks can help mitigate this. Data stored on SD cards or

SSDs is at risk of corruption; hence, frequent backups and using trustworthy storage devices are essential. Nearby electronic devices might interfere with Bluetooth or Zigbee signals, so tests should be conducted in controlled environments, preferably with signal isolation. Ensuring comprehensive test coverage is vital to avoid overlooking any scenarios.

9. Appendices

Not applicable.