Test Specification

For

Chariot

Submitted by

**Chariot Dev**

|  |
| --- |
| **Instructor:**Dr. Gregory Hislop |
| **Team Members:** Medard Azandegbe, Kamalludin Colaire, Ryan Hassing, John Ho, Juan Garcia Lopez, Christopher Mak, Enioluwa Segun |
| **Cycle:**2 |
| **Date Submitted:** |

Document template copyright 2005-2019, Gregory W. Hislop. Version 2.2. Use permitted under Creative Commons license CC-BY-NC-SA. See http://creativecommons.org/licenses/by-nc-sa/3.0/.

Grading Rubric – Test Specification

This rubric outlines the grading criteria for this document. Note that the criteria represent a plan for grading. Change is possible, especially given the dynamic nature of this course. Any change will be applied consistently for the entire class.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Achievement** | **Minimal** | **Exemplary** | **Pts** | **Score** |
| **Content** | Section(s) missing, not useful, inconsistent, or wrong. | Provides all relevant information correctly and with appropriate detail |  |  |
| Introduction |  |  | 5 |  |
| Test Specs |  |  |  |  |
| Selection | Aspects tested are trivial | Tests clearly address core system functions | 20 |  |
| Organization | Tests are disorganized, IDs or Objectives are not meaningful | Tests are well-organized with structured IDs and clear objectives | 20 |  |
| Set-up | Steps are unclear or incomplete | Complete, easy to follow conditions and steps | 20 |  |
| Results | Unclear or incomplete | Complete and clear | 20 |  |
| **Grammar and Spelling** | Many serious mistakes in grammar or spelling | Grammar, punctuation, and spelling all correct | 5 |  |
| **Expression** | Hard to follow or poor word choices | Clear and concise. A pleasure to read | 5 |  |
| **Tone** | Tone not appropriate for technical writing | Tone is consistently professional |  |  |
| **Organization** | Information difficult to locate | All information is easy to find and important points stand out | 5 |  |
| **Layout** | Layout is inconsistent, visually distracting, or hinders use | Layout is attractive, consistent, and helps guide the reader |  |  |
| **Late Submission** |  |  |  |  |
| **Total** |  |  | 100 |  |

Table of Contents

[1 Introduction 4](#_Toc33354152)

[Purpose 4](#_Toc33354153)

[References 4](#_Toc33354154)

[Priority Reference 4](#_Toc33354155)

[Target User 4](#_Toc33354156)

[Terminology 4](#_Toc33354157)

[2 Test Specifications 6](#_Toc33354158)

# Introduction

## Purpose

The purpose of this Test Specification document is to instruct a tester on how to test the functional requirements of Chariot. The corresponding functional requirements can be found in Chariot’s Software Requirements Specification. The order of the tests progresses logically from initial login and setup to data collection. If unfamiliar terms appear in this document, consult the Software Requirements Specification or Software Design Document for explanation.

## References

Humphrey, Watts. *A Discipline for Software Engineering*. Reading, MA: Addison - Wesley Publishing Co. 1995.

Chariot Dev. *Software Requirements Specification.* Chariot Dev 2020.

Chariot Dev. *Software Design Document.* Chariot Dev 2020.

## Priority Reference

Each test below is labeled as either Essential \*\*\*, High \*\*, or Low \* priority.

* Chariot must pass Essential priority tasks in order to meet stakeholder expectations.
* Chariot should pass High priority tests to significantly improve the quality of the final product. These tests should be addressed after all Essential priority tests.
* Chariot should pass Low priority tests to add minor improvements to the final product. These tests should be addressed after all High priority tests. May not make it to final product.

## Target User

If a test is labelled with “Target User: Technical”, only a Technical user may perform the actions. See Section 2.2 in the Chariot Software Requirements Specification for information about Technical users.

## Terminology

Network & Device XYZ: These terms will be used when referencing the object dictated at hand.

WSL Terminal: Windows Subsystem for Linux terminal.

# Test Specifications

1. Pre Requirements \*\*\*

|  |  |
| --- | --- |
| **Objective** | User can install WSL and pre-requirements for chariot |
| **Requirement** | FR1 |
| **Setup** | User opens windows store  User searches for Ubuntu 18.04 LTS  User opens Ubuntu 18.04(WSL Terminal)  User Proceeds to install python, pip, venv. |
| **Expected Results** | User proceeds through installation without fail.  Upon completion of installation, Chariot is installed to a file location specified by user.  User can now start to install chariot. |
| **Actual Results** | Machine Windows 10 WSL Ubuntu  User goes to start menu  User Searched for windows store  User searches for Ubuntu 18.04 within the windows store and installs this by clicking the get button.    Upon install Ubuntu WSL can be found in the start menu by searching for ubuntu 18.04  Within WSL terminal  **sudo apt update && upgrade**  Installed Python3 with the following cmd  **sudo apt-get install python3**  Installed python3-pip with the following cmd  **sudo apt-get install python3-pip**  update and upgrade above with the following command  **sudo apt update && upgrade**  install virtual enviroment on python3 with the following cmd  **sudo apt-get install python3-venv**  Errors occurred  [Yesterday 5:04 PM] Mak,Christopher  failures for building wheel for mysql-connector  ​  [Yesterday 5:04 PM] Mak,Christopher  failed building wheel for pyyaml  ​  [Yesterday 5:04 PM] Mak,Christopher  failed build wheel  for sllurp  ​  [Yesterday 5:04 PM] Mak,Christopher  failed building wheel for tinydb-serialization  ​  [Yesterday 5:04 PM] Mak,Christopher  failed building wheel for tinymongo  ​  [Yesterday 5:04 PM] Mak,Christopher  failed building wheel for wrapt  resolved mysql-connector with  **pip install mysql-connector-python**  resolved pyyaml with  **pip install PyYAML**  resolved sllurp with  **pip install sllurp**  resolved tinydb  **pip install tinydb**  resolved tinymongo  **pip install tinymongo**  resolved tinywrapt  **pip install tinywrapt**  Prep for installation complete |
|  |  |

1. Installation \*\*\*

|  |  |
| --- | --- |
| **Objective** | User can install Chariot correctly. |
| **Requirement** | FR1 |
| **Setup** | User opens web browser  User types <https://github.com/chariot-dev/chariot> in address bar  User creates space they want to install chariot  User starts up WSL terminal  User navigates to folder they want to install with WSL terminal  User within WSL terminal after navigating to the folder then clones the chariot-dev/chariot  User then navigates to install within chariot using the WSL terminal.  User runs install.sh with bash within WSL terminal. |
| **Expected Results** | Chariot is installed. |
| **Actual Results** | Opened browser and navigated to:  <https://github.com/chariot-dev/chariot>  Copied the clone address    Created folder within my desktop named test  Opened WSL terminal  Used the following commands to get to my c drive  Cd ..  Cd ..  Cd mnt  Cd c  After this navigated to the folder with the following command  Cd Users/username/Desktop/test  Used the following command to clone chariot-dev git  Git clone <https://github.com/chariot-dev/chariot.git>  Opened folder in explorer and created venv folder within install folder within newly cloned chariot folder.  Within WSL terminal changed directories to install  **Cd ../install or cd install**  ran install.sh  **bash install.sh**  encountered error stating that node was not the correct version [existing nodejs version was not at least 10]  issue was resolved with the following directions in the link below.  <https://gist.github.com/noygal/6b7b1796a92d70e24e35f94b53722219>  ran install again  bash install.sh  install runs cleanly.  Passed with a few errors |

1. Welcome screen \*\*\*

|  |  |
| --- | --- |
| **Objective** | Upon startup, Chariot’s “Login Screen” loads. |
| **Requirement** | FR2 |
| **Setup** | User has WSL terminal open and navigated to chariot folder.  User then changes directory to run  User uses the following command while in the run directory within WSL.  **Bash run.sh**  Backend starts and electron gui appears. |
| **Expected Results** | User sees login screen [i.e electron gui]  Backend is running |
| **Actual Results** | With WSL terminal running and already at the chariot folder.  Cd run  Bash run.sh  Server show up  Gui does not load  Checking to see if server is running properly by opening web browser.  Within address bar of web browser typed the following.  Localhost:3000  Gui appears.  Test failed due to gui not showing up.  Second attempt is running bash run-core.sh  In cmd prompt open up yarn start  After this opened in a cmd prompt yarn electron  Gui appears  Second test success |

1. Unsuccessful Account Creation – Empty/Invalid email \*\*

|  |  |
| --- | --- |
| **Objective** | User cannot create account with empty or invalid email. |
| **Requirement** | FR3 |
| **Setup** | User is at the “Login Screen”.  User clicks “Don’t have an account? Create one here.”  User leaves email field empty or invalid.  User clicks “Next”. |
| **Expected Results** | Chariot notifies user of empty or invalid email.  Chariot directs user back to “Register Screen”. |
| **Actual Results** | Input of all fields besides email.  Error is presented with please fill out this field.  Successful  Input of all fields, but with an improper email format i.e abc123@asd is accepted.  Failed  refer to remoteDb.docx document |

1. Unsuccessful Account Creation – Empty/Invalid password \*\*

|  |  |
| --- | --- |
| **Objective** | User cannot create account with empty or invalid password. |
| **Requirement** | FR3 |
| **Setup** | User is at the “Login Screen”.  User clicks “Don’t have an account? Create one here.”  User leaves password field empty or invalid.  User clicks “Next”. |
| **Expected Results** | Chariot notifies user of empty or invalid password.  Chariot directs user back to “Register Screen”. |
| **Actual Results** | Within create account screen all fields are filled, but the password fields are left blank.  Error is presented and user is prompted to fill in the missing field.  Successful  Within the create account screen all fields are filled, but passwords do not match.  Account creation accepts this as valid  Failed  refer to remoteDb.docx document |

1. Successful Login \*\*

|  |  |
| --- | --- |
| **Objective** | User can log in. |
| **Requirement** | FR3 |
| **Setup** | User has created an account and verified it.  User is at the “Login Screen”.  User enters their username and password.  User clicks “Login”. |
| **Expected Results** | Chariot directs user to “Welcome Screen”. |
| **Actual Results** | Current stated allows login as long as password and username field is filled WIP pending account creation function.  refer to remoteDb.docx document |

1. Unsuccessful Login \*\*

|  |  |
| --- | --- |
| **Objective** | User cannot log in with incorrect username/password. |
| **Requirement** | FR3 |
| **Setup** | User has created an account.  User is at the “Login Screen”.  User enters an incorrect username/password combination.  User clicks “Login”. |
| **Expected Results** | Chariot notifies user of incorrect username/password combination.  Chariot directs user back to “Login Screen”. |
| **Actual Results** | Current stated allows login as long as password and username field is filled WIP pending account creation function.  refer to remoteDb.docx document |

1. Initialize Network – Manual \*\*\*

|  |  |
| --- | --- |
| **Objective** | User can initialize a new network of IoT devices via manual entry. |
| **Requirement** | FR6 |
| **Setup** | User is at the “Welcome Screen”.  User navigates to “Network Manager Screen”.  User navigates to “Add a Network Screen”.  User enters network name and description.  User confirms network information and clicks “Yes”. |
| **Expected Results** | After confirming, the network will be created. The user will now have the option to add a device to that newly created network.  If user selects “Yes”, Chariot will direct the user to the “Add a Device Screen”. If user selects “No”, Chariot will direct the user to back to the “Welcome Screen”.  If user navigates to “Manage Existing Networks Screen”, “Delete a Network Screen”, or “Configure Data Collection” screens, the initialized network now appears in the list of networks. |
| **Actual Results** | After successfully logging the user will be at the welcome screen.  I then click the “Network Manager”  I then click “Add a Network”  I then fill in both network name and network description fields.  Confirmation pops up confirming inputted information.  Network is created  Checking network creation by going back to “Network Manager”  And then clicking on delete a network.  New network was available to delete.  Successful |

1. Initialize Network – From Network Configuration \*\*\*

|  |  |
| --- | --- |
| **Objective** | User can initialize a new network of IoT devices by loading a network configuration. |
| **Requirement** | FR6 |
| **Setup** | User is at “Enter Network Information Screen”.  User clicks “Load a Configuration for this Network”.  Chariot displays list of network configurations.  User selects Configuration X from list and clicks “Use this Network Configuration”. |
| **Expected Results** | Chariot enters network information based on Configuration X.  User may confirm or modify network information. |
| **Actual Results** | Not implemented. |

1. Manage Existing Network – Network Settings \*\*\*

|  |  |
| --- | --- |
| **Objective** | User can manage an existing network’s network-specific settings. |
| **Requirement** | FR5 |
| **Setup** | User is at the “Welcome Screen”.  User navigates to “Network Manager Screen”.  User navigates to “Manage Existing Networks Screen”.  User selects Network X from list of networks.  Chariot directs user to Network X’s “Manage This Network Screen”.  User navigates to “Manage Network-Specific Settings Screen”.  Chariot displays Network X’s network-specific settings.  User modifies one or more network-specific setting.  User confirms network-specific settings and clicks “Save”. |
| **Expected Results** | Chariot directs user back to “Manage Existing Networks Screen”.  The user can verify these changes by navigating back to Network X's "Manage Network-Specific Settings Screen", there Chariot will display the updated network-specific settings. |
| **Actual Results** | Navigated to “Network Manager Screen”  Navigated to “Manage Existing Network”  Selected Test Network  Clicked add device  Added device  After adding device gui kicks back to “Welcome Screen”.  Navigated to “Network Manager Screen”  Navigated to “Manage Existing Network”  Selected Test Network  New device is present on test network.  Success #1  Clicked on device I want to edit.  After editing device gui kicks back to “Welcome Screen”.  Navigated to “Network Manager Screen”  Navigated to “Manage Existing Network”  Selected Test Network  Device change was present.  Success #2  Clicked on device I want to remove.  After removing device gui kicks back to “Welcome Screen”.  Navigated to “Network Manager Screen”  Navigated to “Manage Existing Network”  Selected Test Network  Device change was not present.  Success #3 |

1. Manage Existing Network – Device Settings \*\*\*

|  |  |
| --- | --- |
| **Objective** | User can manage device-specific settings for a device on a network. |
| **Requirement** | FR5 |
| **Setup** | User is at Network X’s “Manage This Network Screen”.  User navigates to “Network Manager Screen”.  User navigates to “Manage Existing Networks Screen”.  User selects Device Y below a network to navigate to its “Manage Device-Specific Settings Screen”.  Chariot displays Device Y’s device-specific settings.  User modifies one or more device-specific setting.  User confirms device-specific settings and clicks “Save”. |
| **Expected Results** | Chariot directs user back to “Manage Existing Networks Screen”.  If user selects Device Y from the “Manage Existing Networks Screen”, Chariot will display Device Y’s updated device-specific settings. |
| **Actual Results** | Started at “Welcome Screen”  Navigated to “Network Manager Screen”  Navigated to “Manage Existing Network”  Selected Test Network  Clicked device name I wanted to edit.  After editing device gui kicks back to “Welcome Screen”.  Navigated to “Network Manager Screen”  Navigated to “Manage Existing Network”  Selected Test Network  Device change was present. |

1. Save Currently Active Network Configuration \*\*\*

|  |  |
| --- | --- |
| **Objective** | User can save currently active network’s configuration. |
| **Requirement** | None, currently |
| **Setup** | User is at Network X’s “Manage This Network Screen”.  User clicks “Save this Network Configuration”.  User enters a unique network name and clicks “Save”.  User returns to “Enter Network Information Screen” and clicks “Load a Configuration for this Network”, the saved configuration will appear in the list of network configurations. |
| **Expected Results** | New configuration should be selectable from “Load a Configuration for this Network”. |
| **Actual Results** | WIP/n/a |

1. Delete Network \*\*\*

|  |  |
| --- | --- |
| **Objective** | User can delete an existing network of IoT devices. |
| **Requirement** | FR7 |
| **Setup** | User is at the “Welcome Screen”.  User navigates to “Network Manager Screen”.  User navigates to “Delete a Network Screen”.  User selects Network X to be deleted from the list of networks.  User confirms decision to delete Network X. |
| **Expected Results** | Chariot directs user back to “Welcome Screen”.  User can verify network has been deleted by navigating to “Manage Existing Networks Screen”, “Delete a Network Screen”, or “Configure Data Collection Screen”, and seeing that Network X does not appear in the list of networks any longer. |
| **Actual Results** | Started at “Welcome Screen”  Navigated to “Network Manager Screen”  Navigated to “Delete an Existing Network Screen”  List of networks appears  Click on the Delete Network Button to the right of the network I want to delete.  Confirmed Target  After editing device gui kicks back to “Welcome Screen”.  Navigated to “Network Manager Screen”  Navigated to “Manage Existing Network”  Removed Network is not present.  Success |

1. Manage Network Permissions – Add User \*

Note: This feature is unlikely to appear in the final product.

**Target User: Technical**

|  |  |
| --- | --- |
| **Objective** | Technical user can allow other users to use a network. |
| **Requirement** | FR8 |
| **Setup** | User is at Network X’s “Manage This Network Screen”.  User navigates to “Manage Network Permissions Screen”.  User clicks “Add User”.  Chariot prompts user for another user’s username.  User enters User Y’s username and clicks “Add this User”. |
| **Expected Results** | Chariot directs user back to Network X’s “Manage Network Permissions Screen”.  If user navigates to “View Users Screen”, User Y’s username will appear.  If User Y is logged in and navigates to “Manage Existing Networks Screen”, “Delete a Network Screen”, or “Configure Data Collection Screen”, Network X will appear in list of networks. |
| **Actual Results** | Not Implimented |

1. Manage Network Permissions – Remove User \*

Note: This feature is unlikely to appear in the final product.

**Target User: Technical**

|  |  |
| --- | --- |
| **Objective** | Technical user can allow other users to use a network. |
| **Requirement** | FR8 |
| **Setup** | User is at Network X’s “Manage This Network Screen”.  User navigates to “View Users Screen”.  Chariot displays list of users with permissions on network.  User selects User Y from list. Chariot displays “Remove User” button.  User clicks “Remove User”. Chariot prompts user for confirmation, and user confirms. |
| **Expected Results** | Chariot directs user back to Network X’s “Manage Network Permissions Screen”.  If user navigates to “View Users Screen”, User Y’s username will not appear.  If User Y is logged in and navigates to “Manage Existing Networks Screen”, “Delete a Network Screen”, or “Configure Data Collection Screen”, Network X will not appear in list of networks. |
| **Actual Results** | Not Implimented |

1. Add Device to Network – Manual \*\*\*

|  |  |
| --- | --- |
| **Objective** | User can add a new device to an existing network. |
| **Requirement** | FR11 |
| **Setup** | User is at the “Welcome Screen”.  User navigates to “Manage Existing Networks Screen”.  User navigates to “Add a Device to this Network Screen” by clicking “Add Device” button beside a network.  Chariot directs user to “Configure Device Settings Screen”.  User enters nickname and description for new Device Y, and select’s Device Y’s type from the list “Select a Device Type”.  User enters Device Y’s device-specific configuration settings.  User confirms Device Y’s device-specific settings and clicks “Save”. |
| **Expected Results** | Chariot directs user back to “Manage Existing Networks Screen”.  If user selects Network X from list of networks and navigates to Network X’s “Manage Device-Specific Settings Screen”, Device Y will appear in list of devices. |
| **Actual Results** | Started at “Welcome Screen”  Navigated to “Network Manager Screen”  Navigated to “Manage Existing Network”  Clicked add device  Chose device type  Filled in fields as needed.  Port must be 6000  Confirmed device fields.  After adding device gui kicks back to “Welcome Screen”.  Navigated to “Network Manager Screen”  Navigated to “Manage Existing Network”  New device is now present in the network device list.  Success |

1. Add Device to Network – From Device Configuration \*\*

|  |  |
| --- | --- |
| **Objective** | User can add a new device to an existing network by loading a device configuration. |
| **Requirement** | FR11, FR10.2 |
| **Setup** | User is at the “Welcome Screen”.  User navigates to “Manage Existing Networks Screen”.  User selects Network X from list of networks.  User navigates to “Add a Device to this Network Screen”.  Chariot directs user to “Configure Device Settings Screen”.  User enters nickname and description for new Device Y, and select’s Device Y’s type from the list “Select a Device Type”.  User clicks “Load a Configuration for this Device”.  Chariot displays list of device configurations valid for use with Device Y.  User selects Configuration Z from list and clicks “Use this Device Configuration”. |
| **Expected Results** | Chariot enters Device Y’s device-specific configuration settings based on Configuration Z.  User may confirm or modify settings. |
| **Actual Results** | Not Implimented |

1. Save Device Configuration \*\*

|  |  |
| --- | --- |
| **Objective** | When user adds a device to a network, user may save device configuration. |
| **Requirement** | FR10.1 |
| **Setup** | User has entered Device Y’s device-specific configuration settings, either via manual entry or from a device configuration.  User clicks “Save this Device Configuration”.  User enters a unique file name and clicks “Save”. |
| **Expected Results** | Chariot directs user back to “Configure Device Settings Screen”.  If user navigates to “Configure Device Settings Screen” and clicks “Load a Configuration for this Device” while configuring a device of the same type as Device Y, the saved configuration will appear in the list of device configurations. |
| **Actual Results** | Not Implemented |

1. Remove Device from Network \*\*

|  |  |
| --- | --- |
| **Objective** | User can remove a device from an existing network. |
| **Requirement** | FR13 |
| **Setup** | User is at Device X’s “Configure Device Settings Screen”.  User clicks “Remove this Device”. Chariot prompts user for confirmation, and user confirms. |
| **Expected Results** | Chariot directs user back to “Manage This Network Screen”.  If user navigates to “Manage Device-Specific Settings Screen”, Device X will not appear in list of devices. |
| **Actual Results** | Started at “Welcome Screen”  Navigated to “Network Manager Screen”  Navigated to “Manage Existing Network”  Clicked the name of the device I want to remove.  Clicked Delete Device  Confirmed Decision  After removing the device gui kicks back to “Welcome Screen”.  Navigated to “Network Manager Screen”  Navigated to “Manage Existing Network”  New device is now missing from the network device list. |

1. Initialize Database Writer Configuration – Manual \*\*\*

|  |  |
| --- | --- |
| **Objective** | User can initialize a database writer by manually entering all settings. |
| **Requirement** | FR21 |
| **Setup** | User is at the “Welcome Screen”.  User navigates to “Add a Database Writer Screen”.  User enters nickname and description for new Database Writer X, and select’s Database Writer X’s database type from the list “Select a Database Writer Type”.  User enters Database Writer X’s database-specific configuration settings.  User confirms Database Writer X’s database-specific settings and clicks “Save”. |
| **Expected Results** | Chariot directs user back to “Welcome Screen”.  If user navigates to “Configure Database Writer Settings Screen”, Database Writer X will appear in list of database writers. |
| **Actual Results** | Not Implemented in an interactable way. |

1. Initialize Database Writer – From Database Writer Configuration \*\*

|  |  |
| --- | --- |
| **Objective** | User can initialize a database writer by loading a database writer configuration. |
| **Requirement** | FR21 |
| **Setup** | User is at the “Welcome Screen”.  User navigates to “Add a Database Writer Screen”.  User enters nickname and description for new Database Writer X, and select’s Database Writer X’s database type from the list “Select a Database Writer Type”.  User clicks “Load a Configuration for this Database Writer”.  Chariot displays list of database writer configurations valid for use with Database Writer X.  User selects Configuration Y from list and clicks “Use this Database Writer Configuration”. |
| **Expected Results** | Chariot enters Database Writer X’s configuration settings based on Configuration Y.  User may confirm or modify settings. |
| **Actual Results** | Not Implemented in an interactable way. |

1. Save Database Writer Configuration \*\*

|  |  |
| --- | --- |
| **Objective** | When user adds a database writer to a network, user may save database writer configuration. |
| **Requirement** | FR21 |
| **Setup** | User has entered Database Writer X’s configuration settings, either via manual entry or from a database writer configuration.  User clicks “Save this Database Writer Configuration”.  User enters a unique file name and clicks “Save”. |
| **Expected Results** | Chariot directs user back to “Welcome Screen”.  If user navigates to “Add a Database Writer Screen” and clicks “Load a Configuration for this Database Writer” while configuring a database writer of the same type as Database Writer X, the saved configuration will appear in the list of database writer configurations. |
| **Actual Results** | Not Implemented in an interactable way. |

1. Remove Database Writer \*\*

|  |  |
| --- | --- |
| **Objective** | User can remove a database writer. |
| **Requirement** | None |
| **Setup** | User is at the “Welcome Screen”.  User navigates to “Remove a Database Writer Screen”.  User selects Database Writer X and clicks “Remove this Database Writer”. |
| **Expected Results** | Chariot directs user back to “Welcome Screen”.  If user initializes a data collection episode, Database Writer X does not appear in list of database writers on “Choose a Database Writer Screen”. |
| **Actual Results** | Not Implemented in an interactable way. |

1. Initialize Data Collection Episode – Manual \*\*\*

|  |  |
| --- | --- |
| **Objective** | User can initialize a data collection episode by manually entering all settings. |
| **Requirement** | FR17.1 |
| **Setup** | User is at the “Welcome Screen”.  User navigates to “Configure Data Collection Screen”.  User selects one of the enabled networks.  User selects one of the enabled databases.  Chariot directs user to “Choose a Network Screen”. Chariot displays list of networks.  User selects a network from list and clicks “Use this Network”.  Chariot directs user to “Data Collection Episode Configuration Screen”.  User configures other data collection episode configuration settings and clicks “Use this Data Collection Episode Configuration”.  Chariot directs user to “Start Data Collection Episode Screen”.  User clicks “Start Data Collection Episode”. |
| **Expected Results** | Data collection episode begins. Chariot directs user to “Data Collection Status Screen”. |
| **Actual Results** | Started at “Welcome Screen”  Clicked Configure Data Collection button.  Select Network  Select Database  Input runtime and config name.  Click Begin Collection  Data collection seems to start nothing writes to DB  Partial Success. |

1. Initialize Data Collection Episode – From Data Collection Episode Configuration \*\*

|  |  |
| --- | --- |
| **Objective** | User can initialize a data collection episode by loading a data collection episode configuration. |
| **Requirement** | FR19.2 |
| **Setup** | User is at the “Configure Data Collection Screen”.  User clicks “Load a Data Collection Episode Configuration”.  Chariot displays list of data collection episode configurations.  User selects Configuration X from list and clicks “Use this Data Collection Episode Configuration”. |
| **Expected Results** | Chariot enters data collection episode configuration settings based on Configuration X.  User may confirm or modify data collection episode configuration.  Chariot directs user to “Start Data Collection Episode Screen”. |
| **Actual Results** | Not Implemented |

1. Save Data Collection Episode Configuration \*\*

|  |  |
| --- | --- |
| **Objective** | User can save a data collection episode configuration. |
| **Requirement** | FR19.1 |
| **Setup** | User has entered data collection episode configuration settings, either via manual entry or from a data collection episode configuration.  User clicks “Save this Data Collection Episode Configuration”.  User enters a unique file name and clicks “Save”. |
| **Expected Results** | Chariot directs user back to “Welcome Screen”.  If user navigates to “Configure Data Collection Screen” and clicks “Load a Data Collection Episode Configuration”, the saved configuration will appear in the list of data collection episode configurations. |
| **Actual Results** | Not Implemented |

1. Display Device Status During Data Collection Episode \*\*

|  |  |
| --- | --- |
| **Objective** | During a data collection episode, Chariot shall display the status of each device on the network. |
| **Requirement** | FR16 |
| **Setup** | User is at the “Data Collection Status Screen”. A data collection episode is in progress. |
| **Expected Results** | For each device on the network being used in the data collection episode,   * If Chariot is receiving data from the device as expected, Chariot displays “Device Status: Connected”. * If device data stream is interrupted, Chariot displays “Device Status: Disconnected”. |
| **Actual Results** | Not Implemented |

1. Pause Data Collection Episode \*

|  |  |
| --- | --- |
| **Objective** | User can pause a data collection episode in progress. |
| **Requirement** | FR17.2 |
| **Setup** | User is at the “Data Collection Status Screen”. A data collection episode is in progress.  User clicks “Pause Data Collection Episode”. |
| **Expected Results** | Chariot directs user to “Data Collection Episode Paused Screen”. |
| **Actual Results** | Not Implemented |

1. Resume Data Collection Episode \*

|  |  |
| --- | --- |
| **Objective** | User can resume a paused data collection episode. |
| **Requirement** | FR17.3 |
| **Setup** | User is at the “Data Collection Episode Paused Screen”.  User clicks “Resume Data Collection Episode”. |
| **Expected Results** | Chariot directs user to “Data Collection Status Screen”. Data collection episode resumes. |
| **Actual Results** | Not Implemented |

1. Add Device During Data Collection Episode \*

|  |  |
| --- | --- |
| **Objective** | While a data collection episode is in progress, a user can add a device to the network. |
| **Requirement** | FR12 |
| **Setup** | User is at the “Data Collection Status Screen”. A data collection episode is in progress.  User navigates to the network’s “Add a Device to this Network Screen”.  User adds a device as specified in T16 or T17. |
| **Expected Results** | If user navigates back to “Data Collection Status Screen”, newly added device’s status should appear. |
| **Actual Results** | Not Implemented |

1. Unintentionally Remove Device During Data Collection Episode \*

|  |  |
| --- | --- |
| **Objective** | During a data collection episode, if Chariot loses connection with a device, Chariot will attempt to reconnect. If Chariot cannot reconnect with the device, the data collection episode will continue without receiving data from the device. |
| **Requirement** | FR14 |
| **Setup** | User is at the “Data Collection Status Screen”. A data collection episode is in progress. The data collection episode involves Device X.  User manually interrupts Chariot’s connection with Device X. (This may involve powering the device off or turning off its networking capabilities. Exact instructions depend on the device type.) |
| **Expected Results** | Device X’s status is “Device Status: Disconnected”.  After connection timeout period, Chariot informs user that Device X has been removed from the data collection episode. |
| **Actual Results** | Not Implemented |

1. Intentionally Remove Device During Data Collection Episode \*

|  |  |
| --- | --- |
| **Objective** | During a data collection episode, user can remove a device from the network. The data collection episode will continue without receiving data from the device. |
| **Requirement** | FR15 |
| **Setup** | User is at the “Data Collection Status Screen”. A data collection episode is in progress. The data collection episode involves Device X.  User removes Device X as specified in T19. |
| **Expected Results** | Chariot informs user that Device X has been removed from the data collection episode.  Device X does not appear on “Data Collection Status Screen”. |
| **Actual Results** | Not Implemented |

1. Concurrent Device Data Collection \*

|  |  |
| --- | --- |
| **Objective** | User can run a data collection episode using multiple devices concurrently. |
| **Requirement** | FR18.1 |
| **Setup** | User has configured a data collection episode using a network with 2 devices of the same type.  User is at “Start Data Collection Episode Screen”.  User clicks “Start Data Collection Episode”. |
| **Expected Results** | Chariot directs user to “Data Collection Status Screen”.  Both devices on the network appear on “Data Collection Status Screen”.  Data collection episode progresses as expected. |
| **Actual Results** | Not Implemented |

1. Concurrent Heterogeneous Device Data Collection \*

|  |  |
| --- | --- |
| **Objective** | User can run a data collection episode using heterogeneous devices concurrently. |
| **Requirement** | FR18.1 |
| **Setup** | User has configured a data collection episode using a network with 2 devices of different types.  User is at “Start Data Collection Episode Screen”.  User clicks “Start Data Collection Episode”. |
| **Expected Results** | Chariot directs user to “Data Collection Status Screen”.  Both devices on the network appear on “Data Collection Status Screen”.  Data collection episode progresses as expected. |
| **Actual Results** | Not Implemented |

1. Terminate Data Collection Episode \*\*\*

|  |  |
| --- | --- |
| **Objective** | User can terminate a data collection episode in progress. |
| **Requirement** | FR17.2 |
| **Setup** | User is at the “Data Collection Status Screen”. A data collection episode is in progress.  User clicks “Terminate Data Collection Episode”.  Chariot prompts user for confirmation, and user confirms. |
| **Expected Results** | Chariot directs user back to “Welcome Screen”. |
| **Actual Results** | Assuming Data Collection is active.  User is on “Data Collection Episode Screen”  User clicks on End Data Collection.  Data Collection Pop up appears network traffic does not seem to stop.  Partial Fail. |

1. Write to Database \*\*

|  |  |
| --- | --- |
| **Objective** | During a data collection episode, Chariot shall write data using the database writer specified. |
| **Requirement** | FR21 |
| **Setup** | Data collection episode is complete.  User connects to the database that database writer is configured to write to.  User queries all data in table that database writer is configured to write to. |
| **Expected Results** | Data received during data collection episode should appear in database. |
| **Actual Results** | Failed. |

1. Write to Database – Timestamp Database Insertion Time \*\*

|  |  |
| --- | --- |
| **Objective** | Chariot shall timestamp all data with the time it is written to the database. |
| **Requirement** | FR22 |
| **Setup** | Data collection episode is complete.  User connects to the database that database writer is configured to write to.  User queries all data in table that database writer is configured to write to. |
| **Expected Results** | In the table, there is a column indicating database insertion time.  Each row should have a timestamp in the database insertion time column. |
| **Actual Results** | Not Implemented |

1. Write to Database – Timestamp Data Collection Time \*\*

|  |  |
| --- | --- |
| **Objective** | Chariot shall timestamp all data with the time it is collected by the device. |
| **Requirement** | FR23 |
| **Setup** | Data collection episode is complete.  User connects to the database that database writer is configured to write to.  User queries all data in table that database writer is configured to write to. |
| **Expected Results** | In the table, there is a column indicating device collection time.  Each row should have a timestamp in the device collection time column. |
| **Actual Results** | Not Implemented |