Green Button Alliance Green Button Data Custodian Connect My Data Test Sheet

# Overview

This document contains the test data sheet and procedures for performing the Green Button Data Custodian Connect My Data certification tests.

## References

1. The latest version of this Data Test Sheet: <http://files.gbitca.org/dc-cmd-test-data-sheet>
2. Application for testing: <http://files.gbitca.org/dc-cmd-cert-form>

# Function Block Test Execution Instructions

[FB\_01] Common

Tests: [TR\_FND001] Initial configuration of the DataCustodian under test and provide test software with prerequisite information

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software loads and verifies the test applicant's supplied configuration information

2. Test software configures and activates the Stunnel proxy server

3. Test software verifies if the test applicant supplied a registration\_access\_token in the configuration information

4. If a registration\_access\_token was provided, the test software proceeds to step 6, otherwise performs the following:

a. Test software issues a POST request to the test applicant's OAuth 2.0 Token Endpoint with a grant\_type=client\_credentials

5. Test software upon receiving a response to the POST request performs the following:

a. Validates the response contains a 200 status code. Saves the contents of the JSON response structure

6. Test software issues a GET request to the test applicant's ApplicationInformation API using the registration\_access\_token

7. Test software upon receiving a response to the GET request performs the following:

a. Validates the response contains a 200 status code

b. Validates the response contains an 'espi:dataCustodianScopeSelectionScreenURI' XML <tag> entry

c. Initializes internal test software properties based on contents of GET response

8. Test software verifies if the test applicant supplied a client\_acccess\_token in the configuration information

9. If a client\_access\_token was provided, the test software terminates the test, otherwise performs the following:

a. Test software issues a POST request to the test applicant's OAuth 2.0 Token Endpoint with a grant\_type=client\_credentials

10. Test software upon receiving a response to the POST request performs the following:

a. Validates the response contains a 200 status code

b. Saves the contents of the JSON response structure

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_CPYRT001] Verify the implementer has purchased the NAESB ESPI standard.

Procedure:

1. Test software opens a browser window and accesses the "NAESB Copyright Policy and Companies with Access to NAESB Standards Under the Copyright Policy" PDF document

2. Test software displays a dialog box containing the following:

A screenshot of a social media post

Description generated with very high confidence

3. Test engineer follows the instructions in the above dialog box to confirm the test applicant has purchased the NAESB REQ.21 standard.

4. Test engineer selects '**YES**' if they can verify test applicant has purchased a copy of the NAESB REQ.21 Standard or selects '**No**' if unable to confirm test applicant has purchased a copy of the NAESB REQ.21 Standard

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_CERT001] Verify ApplicationInformation resource GET request response contains a valid Certification Link in the response

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. The test software issues an HTTP GET request to the ApplicationInformation by ID API

2. Upon receiving a response to the GET request, the test software:

a. Verifies the HTTP Status code is a “200”

b. Verifies the response contains a properly formatted Certification Link

c. Verifies the Certification Link contains the GBA issued GBA Certification ID

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_CERT002] Verify Authorization GET request response contains a valid Certification Link in the response

Procedure:

1. The test software checks to see if an Authorization has been issued by the Data Custodian

2. If the test software determines an Authorization exists, it continues execution at step 7.

3. If the test software determines an Authorization request is required, it displays a dialog box containing:

A screenshot of a cell phone

Description generated with very high confidence

4. The test engineer then gives remote control to the test applicant to complete the logon and authorization process.

5. When the test applicant has finished the following will appear in the browser:

A screenshot of a cell phone

Description generated with very high confidence

6. When the above appears in the browser, the test engineer regains control of the system and selects "OK" in the displayed dialog box.

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

7. The test software issues an HTTP GET request to the Authorization API

8. Upon receiving a response to the GET request, the test software:

a. Verifies the HTTP Status code is a “200”

b. Verifies the response complies with the Green Button schema

c. Verifies the response contains a properly formatted Certification Link

d. Verifies the Certification Link contains the GBA issued GBA Certification ID

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test steps.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_CERT003] Verify Authorization resource GET request response contains a valid Certification Link in the response

Procedure:

1. The test software checks to see if an Authorization has been issued by the Data Custodian

2. If the test software determines an Authorization exists, it continues execution at step 7.

3. If the test software determines an Authorization request is required, it displays a dialog box containing:

A screenshot of a cell phone

Description generated with very high confidence

4. The test engineer then gives remote control to the test applicant to complete the logon and authorization process.

5. When the test applicant has finished the following will appear in the browser:

A screenshot of a cell phone

Description generated with very high confidence

6. When the above appears in the browser, the test engineer regains control of the system and selects "OK" in the displayed dialog box.

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

7. The test software issues an HTTP GET request to the Authorization API

8. Upon receiving a response to the GET request, the test software:

a. Verifies the HTTP Status code is a “200”

b. Verifies the response complies with the Green Button schema

c. Verifies the response contains a properly formatted Certification Link

d. Verifies the Certification Link contains the GBA issued GBA Certification ID

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_CERT004] Verify Batch/Subscription GET request response contains a valid Certification Link in the response

Procedure:

1. The test software checks to see if an Authorization has been issued by the Data Custodian

2. If the test software determines an Authorization exists, it continues execution at step 7.

3. If the test software determines an Authorization request is required, it displays a dialog box containing:

A screenshot of a cell phone

Description generated with very high confidence

4. The test engineer then gives remote control to the test applicant to complete the logon and authorization process.

5. When the test applicant has finished the following will appear in the browser:

A screenshot of a cell phone

Description generated with very high confidence

6. When the above appears in the browser, the test engineer regains control of the system and selects "OK" in the displayed dialog box.

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

7. The test software issues an HTTP GET request to the Batch/Subscription API

8. Upon receiving a response to the GET request, the test software:

a. Verifies the HTTP Status code is a “200”

b. Verifies the response complies with the Green Button schema

c. Verifies the response contains a properly formatted Certification Link

d. Verifies the Certification Link contains the GBA issued GBA Certification ID

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_03] Connect My Data

Tests: [TR\_FND002] Verify positive GET behavior for atom:entry with ApplicationInformation as the source using registration\_access\_token

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Steps \*\*\*\*\*\*\*\*\*\*

1. Test software issues an HTTP GET request for Data Custodian's ApplicationInformation resource

2. Upon receiving a response to the GET request, the test software:

a. Verifies the HTTP Status code is a “200”

b. Verifies the response complies with the Green Button schema

c. Verifies the {dataCustodianApplicationStatus} field contains a value of "1", "2", "3" or "4"

d. Verifies the {token\_endpoint\_auth\_method} field contains the value "client\_secret\_basic"

e. Verifies the {grant\_types} field contains a value of "authorization\_code", "refresh\_token" or "client\_credentials"

f. Verifies the {response\_types} field contains a value of "code"

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_FND003] Verify positive HTTP GET response to the Authorization API using client\_access\_token

Procedure:

1. The test software checks to see if an Authorization has been issued by the Data Custodian

2. If the test software determines an Authorization exists, it continues execution at step 7.

3. If the test software determines an Authorization request is required, it displays a dialog box containing:

A screenshot of a cell phone

Description generated with very high confidence

4. The test engineer then gives remote control to the test applicant to complete the logon and authorization process.

5. When the test applicant has finished the following will appear in the browser:

A screenshot of a cell phone

Description generated with very high confidence

6. When the above appears in the browser, the test engineer regains control of the system and selects "OK" in the displayed dialog box.

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

7. Test software issues an HTTP GET request to the Authorization API using the client\_access\_token

8. Upon receiving a response to the GET request, the test software:

a. Verifies the HTTP Status code is a “200”

b. Verifies the response complies with the Green Button schema

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_FND004] Verify positive HTTP GET response to the Authorization by ID API using client\_access\_token

Procedure:

1. The test software checks to see if an Authorization has been issued by the Data Custodian

2. If the test software determines an Authorization exists, it continues execution at step 7.

3. If the test software determines an Authorization request is required, it displays a dialog box containing:

A screenshot of a social media post

Description generated with very high confidence

4. The test engineer then gives remote control to the test applicant to complete the logon and authorization process.

5. When the test applicant has finished the following will appear in the browser:

A screenshot of a cell phone

Description generated with very high confidence

6. When the above appears in the browser, the test engineer regains control of the system and selects "OK" in the displayed dialog box.

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

7. Test software issues an HTTP GET request to the Authorization by ID API using the client\_access\_token

8. Upon receiving a response to the GET request, the test software:

a. Verifies the HTTP Status code is a “200”

b. Verifies the response complies with the Green Button schema

c. Verifies the {expires\_at} field contains data

d. Verifies the {scope} field contains data

e. Verifies the {token\_type} field contains data

f. Verifies the {resourceURI} field contains data

g. Verifies the {authorizationURI} field contains data

h. Verifies the {status} field contains "0", "1", or "2"

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_FND005] Verify positive HTTP GET response for request to the Batch/Subscription API using the access\_token

Procedure:

1. The test software checks to see if an Authorization has been issued by the Data Custodian

2. If the test software determines an Authorization exists, it continues execution at step 7.

3. If the test software determines an Authorization request is required, it displays a dialog box containing:

A screenshot of a cell phone

Description generated with very high confidence

4. The test engineer then gives remote control to the test applicant to complete the logon and authorization process.

5. When the test applicant has finished the following will appear in the browser:

A screenshot of a cell phone

Description generated with very high confidence

6. When the above appears in the browser, the test engineer regains control of the system and selects "OK" in the displayed dialog box.

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

7. Test software issues an HTTP GET request to the Batch/Subscription by ID API using the access\_token

8. Upon receiving a response to the GET request, the test software:

a. Verifies the HTTP Status code is a “200”

b. Verifies the response contains a <feed> with at least 1 UsagePoint

c. Verifies the response complies with the Green Button schema

d. Verifies the response passes all requested resource FB XLST test

9. If all requested FB XLST tests completed with PASS results, the test is terminated

10. If any of the requested FB XSLT tests completed with **FAIL** results, the test software displays the following dialog box:

A screenshot of a cell phone

Description generated with very high confidence

11. The test engineer asks the test applicant if they desire to continue the test

12. If the test applicant indicates they want to continue the test, the test engineer selects “Yes” in the dialog box and the test continues at step 1.

13. If the test applicant indicates they do NOT want to continue the test, the test engineer selects “No” in the dialog box and the test is terminated.

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test steps.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_FND006] Verify positive HTTP GET response for request to the ReadServiceStatus API using the client\_access\_token

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software issues an HTTP GET request to the ReadServiceStatus API using the client\_access\_token

2. Upon receiving a response to the GET request, the test software:

a. Verifies the HTTP Status code is a “200”

b. Verifies the response complies with the Green Button schema

c. Verifies the {currentStatus} field contains a “1”

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_04] Interval Metering

This Function Block is executed by the [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite if the test applicant selected to be certified for [FB\_04] Interval Metering.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. The test software performs the following tests:
2. Verifies the received response contains a <espi:MeterReading> entry with:
   * 1. A valid <atom:id> value
     2. A valid <atom:title> value
     3. A valid <atom:published> value
     4. A valid <atom:updated> value
     5. A valid <atom:link[@rel=’self’]/@href> (self-link) value
     6. A unique <atom:link[@rel=’self’]/@href> (self-link) value
     7. A valid <atom:link[@rel=’up’]/@href> (up-link) value
     8. A correct <atom:link[@rel=’up’]/@href> (up-link) value
        1. Verify <espi:UsagePoint> entry referenced contains a <atom:link[@rel=’related’> entry
     9. Verify there is a single <espi:ReadingType> entry associated with the <espi:MeterReading> entry
     10. Verify associated <espi:IntervalBlock> entries exist for each <espi:MeterReading> entry
     11. Verify associated <espi:IntervalBlock> entries exist for each <espi:MeterReading> entry with an associated <espi:ReadingType> entry containing a <espi:accumulationBehavior> value of 4 (delta data)
     12. Verify each <espi:MeterReading> entry points to a  
         <espi:ReadingType> entry
   1. Verifies the received response contains a <atom:feed> entry with:
      1. A valid <espi:MeterReading> entry
      2. A valid <espi:MeterReading> entry with content
      3. A valid <espi:IntervalBlock> entry
      4. A valid <espi:ReadingType> entry
   2. Verifies the received response contains a <espi:IntervalBlock> entry with:
      1. A valid <atom:id> value
      2. A valid <atom:title> value
      3. A valid <atom:published> value
      4. A valid <atom:updated> value
      5. A valid <atom:link[@rel=’self’]/@href> (self-link) value
      6. A unique <atom:link[@rel=’self’]/@href> (self-link) value
      7. A valid <atom:link[@rel=’up’]/@href> (up-link) value
      8. Verify there is a single <espi:ReadingType> entry associated with the <espi:IntervalBlock> entry
      9. A valid <espi:interval> entry with:
         1. A valid <espi:duration> value
         2. A valid <espi:start> value
      10. Verify the first interval in the block start time (<espi:IntervalBlock.interval.start>) matches the start time of the block (<espi:IntervalBlock.IntervalReading.timePeriod.start)
      11. A valid <espi:IntervalReading> entry with:
          1. A valid <espi:timePeriod> entry with:
             1. A valid <espi:duration> value
             2. A valid <espi:start> value
          2. A valid <espi:value> value
   3. Verifies the received response contains a <espi:ReadingType> entry with:
      1. A valid <atom:id> value
      2. A valid <atom:title> value
      3. A valid <atom:published> value
      4. A valid <atom:updated> value
      5. A valid <atom:link[@rel=’self’]/@href> (self-link) value
      6. A unique <atom:link[@rel=’self’]/@href> (self-link) value
      7. A valid <atom:link[@rel=’up’]/@href> (up-link) value
      8. A valid <espi:IntervalLength> value
      9. A valid <espi:kind> value
      10. A valid <espi:powerOfTenMultiplier> value
      11. A valid <espi:uom> value
   4. Verify <espi:MeterReading>.<espi:IntervalBlock> entry is unique
      1. Verify each <espi:IntervalBlock> start time is unique
   5. Verify <espi:MeterReading>.<espi:IntervalBlock>.<espi:IntervalReading> entry is unique
      1. Verify each <espi:IntervalReading> start time is unique

Passing Result:

The **Passing Result:** section of [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite indicates if the test applicant’s application responses [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_05] Interval Electricity Metering

This Function Block is executed by the [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite if the test applicant selected to be certified for [FB\_05] Interval Electricity Metering.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. The test software performs the following tests:
2. Verifies the received response contains a <espi:MeterReading> entry with:
   * 1. A <atom:link[@rel-‘related’> entry
3. Verifies the received response contains a <espi:ReadingType> entry with:
   * 1. A <espi:accumulationBehavior> value of 4 (delta data) and
     2. A <espi:commodity> value of 1 (electricity Secondary Metered) and
     3. A <espi:flowDirection> value of 1 (forward) and
     4. A <espi:kind> value of 12 (energy) and
     5. A <espi:uom> value of 72 (Watt hours (Wh)) and
     6. A valid <espi:phase> value
4. Verifies the received response contains a <espi:UsagePoint> entry with:
   * 1. A <espi:ServiceCategory> entry with:
        1. A <espi:kind> value of 0 (electricity)

Passing Result:

The **Passing Result:** section of [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite indicates if the test applicant’s application responses [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_06] Demand Electricity Metering

This Function Block is executed by the [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite if the test applicant selected to be certified for [FB\_06] Demand Electricity Metering.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. The test software performs the following tests:
2. Verifies the received response contains a <espi:ReadingType> entry with:
   * 1. A <espi:accumulationBehavior> value of 12 (instantaneous) and
     2. A <espi:commodity> value of 1 (electricity Secondary Metered) and
     3. A <espi:flowDirection> value of 1 (forward) and
     4. A <espi:kind> value of 37 (power) and
     5. A <espi:uom> value of 38 (Watt (W))
3. Verifies the received response contains a <espi:ReadingType> entry with:
   * 1. A <espi:accumulationBehavior> value of 12 (instantaneous) and
     2. A <espi:commodity> value of 1 (electricity Secondary Metered) and
     3. A <espi:flowDirection> value of 1 (forward) and
     4. A <espi:kind> value of 12 (energy) and
     5. A <espi:uom> value of 61 (Volt Ampere (VA))
4. Verifies the received response contains a <espi:ReadingType> entry with:
   * 1. A <espi:accumulationBehavior> value of 12 (instantaneous) and
     2. A <espi:commodity> value of 1 (electricity Secondary Metered) and
     3. A <espi:flowDirection> value of 1 (forward) and
     4. A <espi:kind> value of 12 (energy) and
     5. A <espi:uom> value of 63 (Volt Ampere reactive (VAr))

Passing Result:

The **Passing Result:** section of [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite indicates if the test applicant’s application responses [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_07] Net Metering

This Function Block is executed by the [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite if the test applicant selected to be certified for [FB\_07] Net Metering.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. The test software performs the following tests:
2. Verifies the received response contains a <espi:ReadingType> entry with:
   * 1. A <espi:accumulationBehavior> value of 4 (delta data) and
     2. A <espi:commodity> value of 1 (electricity Secondary Metered) and
     3. A <espi:flowDirection> value of 4 (net) and
     4. A <espi:kind> value of 12 (energy) and
     5. A <espi:uom> value of 72 (Watt hours (Wh))

Passing Result:

The **Passing Result:** section of [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite indicates if the test applicant’s application responses [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_08] Forward and Reverse Metering

This Function Block is executed by the [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite if the test applicant selected to be certified for [FB\_08] Forward and Reverse Metering.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. The test software performs the following tests:
2. Verifies the received response contains a <espi:ReadingType> entry with:
   * 1. A <espi:accumulationBehavior> value of 4 (delta data) and
     2. A <espi:commodity> value of 1 (electricity Secondary Metered) and
     3. A <espi:flowDirection> value of 19 (reverse) and
     4. A <espi:kind> value of 12 (energy) and
     5. A <espi:uom> value of 72 (Watt hours (Wh))

Passing Result:

The **Passing Result:** section of [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite indicates if the test applicant’s application responses [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_09] Register Values

This Function Block is executed by the [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite if the test applicant selected to be certified for [FB\_09] Register Values.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. The test software performs the following tests:
2. Verifies the received response contains a <espi:ReadingType> entry with:
   * 1. A <espi:accumulationBehavior> value of 1 (bulk quantity) and
     2. A <espi:commodity> value of 1 (electricity Secondary Metered) and
     3. A <espi:flowDirection> value of 1 (forward) and
     4. A <espi:kind> value of 12 (energy) and
     5. A <espi:uom> value of 72 (Watt hours (Wh))

Passing Result:

The **Passing Result:** section of [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite indicates if the test applicant’s application responses [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_10] Gas

This Function Block is executed by the [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite if the test applicant selected to be certified for [FB\_10] Gas.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. The test software performs the following tests:
2. Verifies the received response contains a <espi:MeterReading> entry with:
   * 1. A <espi:ReadingType> entry with:
        1. A <espi:accumulationBehavior> value of 4 (delta data) and
        2. A <espi:commodity> value of 7 (natural Gas) and
        3. A <espi:flowDirection> value of 1 (forward) and
        4. A <espi:kind> value of 58 (volume) and
        5. A <espi:uom> value of 169 (therm), 31 (joule), 119 (cubic feet), or 42 (cubic meter)
3. Verifies the received response contains a <espi:UsagePoint> entry with:
   * 1. A <espi:ServiceCategory> entry with:
        1. A <espi:kind> value of 1 (gas)

Passing Result:

The **Passing Result:** section of [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite indicates if the test applicant’s application responses [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_11] Water

This Function Block is executed by the [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite if the test applicant selected to be certified for [FB\_11] Water.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. The test software performs the following tests:
2. Verifies the received response contains a <espi:MeterReading> entry with:
   * 1. A <espi:ReadingType> entry with:
        1. A <espi:accumulationBehavior> value of 4 (delta data) and
        2. A <espi:commodity> value of 9 (potable water) and
        3. A <espi:flowDirection> value of 1 (forward) and
        4. A <espi:kind> value of 58 (volume) and
        5. A <espi:uom> value of 128 (US gallons), 119 (cubic feet), or 42 (cubic meter)
3. Verifies the received response contains a <espi:UsagePoint> entry with:
   * 1. A <espi:ServiceCategory> entry with:
        1. A <espi:kind> value of 2 (water)

Passing Result:

The **Passing Result:** section of [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite indicates if the test applicant’s application responses [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_12] Cost of Interval Data

This Function Block is executed by the [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite if the test applicant selected to be certified for [FB\_12] Cost of Interval Data.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. The test software performs the following tests:
2. Verifies the received response contains a <espi:IntervalBlock> entry with:
   * 1. A <espi:IntervalReading> entry with
        1. A valid <espi:cost> value
3. Verifies the received response contains a <espi:ReadingType> entry with:
   * 1. A valid <espi:currency> value

Passing Result:

The **Passing Result:** section of [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite indicates if the test applicant’s application responses [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_13] Security and Privacy Classes

Tests: [TR\_TC001] Verify the Data Custodian uses https for all endpoints.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software issues an HTTP GET request to the ApplicationInformation by ID API

2. Upon receiving a response to the GET request, the test software:

a. Verifies the {**thirdPartyNotifyUri**} field contains a URI using the HTTPs protocol

b. Verifies the {**authorizationServerAuthorizationEndpoint**} field contains a URI using the HTTPs protocol

c. Verifies the {**authorizationServerTokenEndpoint**} field contains a URI using the HTTPs protocol

d. Verifies the {**dataCustodianBulkRequestURI**} field contains a URI using the HTTPs protocol

e. Verifies the {**dataCustodianResourceEndpoint**} field contains a URI using the HTTPs protocol

f. Verifies the {**client\_uri**} field contains a URI using the HTTPs protocol

g. Verifies the {**redirect\_uri**} field contains a URI using the HTTPs protocol

h. Verifies the {**registration\_client\_uri**} field contains a URI using the HTTPs protocol

i. If the {**authorizationServerUri**} field exists in the ApplicationInformation response, verifies it contains a URI using the HTTPs protocol

j. If the {**authorizationServerRegistrationEndpoint**} field exists in the ApplicationInformation response, verifies it contains a URI using the HTTPs protocol

k. If the {**thirdPartyUserPortalScreenURI**} field exists in the ApplicationInformation response, verifies it contains a URI using the HTTPs protocol

l. If the {**thirdPartyScopeSelectionScreenURI**} field exists in the ApplicationInformation response, verifies it contains a URI using the HTTPs protocol

m. If the {**dataCustodianScopeSelectionScreenURI**} field exists in the ApplicationInformation response, verifies it contains a URI using the HTTPs protocol

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_TC003] Verify the Data Custodian under test implements TLS 1.2.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software issues an OpenSSL s\_client connection request to the Authorization Server

2. Upon receiving a response from the OpenSSL request, the test software verifies the Authorization Server supports the SSL TLS1.2 protocol

3. Test software issues an OpenSSL s\_client connection request to the Resource Server

4. Upon receiving a response from the OpenSSL request, the test software verifies the Resource Server supports TLS 1.2

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_TC004] Verify that when communicating with a Retail Customer the Data Custodian negotiates the highest level of TLS supported by both parties.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software issues an OpenSSL s\_client connection request to the Authorization Server requesting to use SSL Protocols TLS1, TLS1.1, or TLS1.2

2. Upon receiving a response from the OpenSSL request, the test software verifies the Authorization Server negotiated a SSL TLS1.2 protocol based session

3. Test software issues an OpenSSL s\_client connection request to the Authorization Server requesting to use SSL Protocols TLS1 or TLS1.1

4. Upon receiving a response from the OpenSSL request, the test software verifies the Authorization Server negotiated a SSL TLS1.1 protocol based session

5. Test software issues an OpenSSL s\_client connection request to the Authorization Server requesting to use SSL Protocols TLS1

6. Upon receiving a response from the OpenSSL request, the test software verifies the Authorization Server negotiated a SSL TLS1 protocol based session

7. Test software issues an OpenSSL s\_client connection request to the Resource Server requesting to use SSL Protocols TLS1, TLS1.1, or TLS1.2

8. Upon receiving a response from the OpenSSL request, the test software verifies the Resource Server negotiated a SSL TLS1.2 protocol based session

9. Test software issues an OpenSSL s\_client connection request to the Resource Server requesting to use SSL Protocols TLS1 or TLS1.1

10. Upon receiving a response from the OpenSSL request, the test software verifies the Resource Server negotiated a SSL TLS1.1 protocol based session

11. Test software issues an OpenSSL s\_client connection request to the Resource Server requesting to use SSL Protocols TLS1

12. Upon receiving a response from the OpenSSL request, the test software verifies the Resource Server negotiated a SSL TLS1 protocol based session

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_TC005] Verify that when communicating with a Retail Customer the Data Custodian rejects TLS\_RSA\_WITH\_NULL\_SHA cipher suites.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software issues an OpenSSL s\_client connection request to the Authorization Server requesting to use the ‘NULL-SHA’ cipher suite

2. Upon receiving a response from the OpenSSL request, the test software verifies the Authorization Server rejected the use of the ‘NULL-SHA’ cipher suite

3. Test software issues an OpenSSL s\_client connection request to the Resource Server requesting to use the ‘NULL-SHA’ cipher suite

4. Upon receiving a response from the OpenSSL request, the test software verifies the Resource Server rejected the use of the ‘NULL-SHA’ cipher suite

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedures.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_TC006] Verify that when communicating with a Retail Customer at a minimum the Data Custodian accepts the TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA cipher suite.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedures \*\*\*\*\*\*\*\*\*\*

1. Test software issues an OpenSSL s\_client connection request to the Authorization Server requesting to use the ‘AES128-SHA’ cipher suite

2. Upon receiving a response from the OpenSSL request, the test software verifies the Authorization Server accepted the use of the ‘AES128-SHA’ cipher suite

3. Test software issues an OpenSSL s\_client connection request to the Resource Server requesting to use the ‘AES128-SHA’ cipher suite

4. Upon receiving a response from the OpenSSL request, the test software verifies the Resource Server accepted the use of the ‘AES128-SHA’ cipher suite

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedures.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_TC008] Verify the Data Custodian SSL Certificate is NOT expired and uses a RSA encrypted 2048 bit length public key.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software issues an OpenSSL s\_client connection request to the Authorization Server

2. Upon receiving a response from the OpenSSL request, the test software verifies the Authorization Server’s SSL Certificate uses a 2048 bit length public key

3. Test software issues an OpenSSL s\_client connection request to the Authorization Server

4. Upon receiving a response from the OpenSSL request, the test software verifies:

a. The Authorization Server’s SSL Certificate has not expired

b. The Authorization Server’s SSL Certificate public key uses the RSA Encryption algorithm

5. Test software issues an OpenSSL s\_client connection request to the Resource Server

6. Upon receiving a response from the OpenSSL request, the test software verifies the Resource Server’s SSL Certificate uses a 2048 bit length public key

7. Test software issues an OpenSSL s\_client connection request to the Resource Server

8. Upon receiving a response from the OpenSSL request, the test software verifies:

a. The Resource Server’s SSL Certificate has not expired

b. The Resource Server’s SSL Certificate public key uses the RSA Encryption algorithm

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedures.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_TC009] Test software or manual inspection shall verify the Data Custodian SSL Certificate is chained to a Certificate Authority (CA) that has been successfully audited as complying with WebTrust or ETSI audit standards.

Procedure:

1. Test software issues an OpenSSL s\_client connection request to the Authorization Server

2. Upon receiving a response from the OpenSSL request, the test software verifies the Authorization Server supports the SSL TLS1.2 protocol

3. If the Authorization Server does NOT support the SSL TLS1.2 protocol, test execution continues at step 8

4. Test software issues an OpenSSL s\_client connection request to the Authorization Server

5. Upon receiving a response from the OpenSSL request, the test software extracts the Certificate Chain and displays a dialog box containing the Root Certificate issuer information:

A screenshot of a social media post

Description generated with very high confidence

6. The test engineer verifies the Data Custodian SSL Certificate is chained to a Certificate Authority (CA) that has been successfully audited as complying with the WebTrust or ETSI audit standards by following the steps described in Appendix A -- “Procedure to Verify Root Certificate Complies with WebTrust or ETSI Audit Standards”

7. The test engineer selects the ‘Yes’ button if they confirm the Data Custodian SSL Certificate is chained to a Certificate Authority (CA) that has been successfully audited as complying with WebTrust or ETSI Audit Standards

8. Test software issues an OpenSSL s\_client connection request to the Resource Server

9. Upon receiving a response from the OpenSSL request, the test software verifies the Resource Server supports the SSL TLS1.2 protocol

10. If the Resource Server does NOT support the SSL TLS1.2 protocol, test execution terminates

11. Test software issues an OpenSSL s\_client connection request to the Resource Server

12. Upon receiving a response from the OpenSSL request, the test software extracts the Certificate Chain and displays a dialog box containing the Root Certificate issuer information:

A screenshot of a cell phone

Description generated with very high confidence

13. The test engineer verifies the Data Custodian SSL Certificate is chained to a Certificate Authority (CA) that has been successfully audited as complying with the WebTrust or ETSI audit standards by following the steps described in the “Root Certificate Proof of WebTrust or ETSI Audit Standards” Appendix

14. The test engineer selects the ‘Yes’ button if they confirm the Data Custodian SSL Certificate is chained to a Certificate Authority (CA) that has been successfully audited as complying with WebTrust or ETSI Audit Standards

Passing Result:

The test engineer visually confirms the Data Custodian SSL Certificate is chained to a Certificate Authority (CA) that has been successfully audited as complying with WebTrust or ETSI Audit Standards.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_TC010] Test software or manual inspection shall verify Tokens and IDs communicated by the Data Custodian are opaque and if based on Customer information are randomized to protect privacy.

Procedure:

1. The test software checks to see if an Authorization has been issued by the Data Custodian

2. If the test software determines an Authorization exists, it continues execution at step 7.

3. If the test software determines an Authorization request is required, it displays a dialog box containing:

A screenshot of a cell phone

Description generated with very high confidence

4. The test engineer then gives remote control to the test applicant to complete the logon and authorization process.

5. When the test applicant has finished the following will appear in the browser:

A screenshot of a cell phone

Description generated with very high confidence

6. When the above appears in the browser, the test engineer regains control of the system and selects "OK" in the displayed dialog box.

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

7. The test software issues an HTTP GET request to the Batch/Subscription by ID API

8. Upon receiving a response to the GET request, the test software:

a. Verifies the HTTP Status code is a “200”

b. Verifies the response complies with the Green Button schema

9. The test software extracts the first 30 <atom:id> entries in the response and displays the following dialog box:

A screenshot of a cell phone

Description generated with very high confidence

10. The test engineer reviews the displayed list of <atom:id> entries to confirm they do NOT contain account or other personal identifiers

11. If the test engineer determines the displayed list of <atom:id> entries does NOT contain account or other personal identifiers, they select the “**No**” button in the displayed dialog box

12. If the test engineer determines the displayed list of <atom:id> entries **CONTAINS** account or other personal identifiers, they select the “**Yes**” button in the displayed dialog box and the test is terminated

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

13. The test software extracts the first 30 <atom:link> entries in the response and displays the following dialog box:

A screenshot of a social media post

Description generated with very high confidence

14. The test engineer reviews the displayed list of <atom:link>entries to confirm they do NOT contain account or other personal identifiers

15. If the test engineer determines the displayed list of <atom:link> entries does NOT contain account or other personal identifiers, they select the “**No**” button in the displayed dialog box

16. If the test engineer determines the displayed list of <atom:link> entries **CONTAINS** account or other personal identifiers, they select the “**Yes**” button in the displayed dialog box and the test is terminated

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

17. The test software extracts the first 30 <atom:title> entries in the response and displays the following dialog box:

A screenshot of a cell phone

Description generated with very high confidence

18. The test engineer reviews the displayed list of <atom:title> entries to confirm they do NOT contain account or other personal identifiers

19. If the test engineer determines the displayed list of <atom:title> entries does NOT contain account or other personal identifiers, they select the “No” button in the displayed dialog box

20. If the test engineer determines the displayed list of <atom:title> entries CONTAINS account or other personal identifiers, they select the “Yes” button in the displayed dialog box and the test is terminated

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_TC011] Test software or manual inspection shall verify that Tokens and IDs communicated by the Data Custodian consist of at least 48 bits and can be the random number part of an RFC2422 UUID.

Procedure:

1. The test software checks to see if an Authorization has been issued by the Data Custodian

2. If the test software determines an Authorization exists, it continues execution at step 7.

3. If the test software determines an Authorization request is required, it displays a dialog box containing:

A screenshot of a cell phone

Description generated with very high confidence

4. The test engineer then gives remote control to the test applicant to complete the logon and authorization process.

5. When the test applicant has finished the following will appear in the browser:

A screenshot of a cell phone

Description generated with very high confidence

6. When the above appears in the browser, the test engineer regains control of the system and selects "OK" in the displayed dialog box.

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

7. The test software issues an HTTP GET request to the Batch/Subscription by ID API

8. Upon receiving a response to the GET request, the test software:

a. Verifies the HTTP Status code is a “200”

b. Verifies the response complies with the Green Button schema

9. The test software displays the following dialog box:

A screenshot of a social media post

Description generated with very high confidence

10. The test engineer reviews the displayed list of tokens to confirm they contain at least 48-bits

11. If the test engineer determines the displayed list of tokens CONTAIN at least 48-bits, they select the “Yes” button in the displayed dialog box

12. If the test engineer determines the displayed list of tokens does NOT contain at least 48-bits, they select the “No” button in the displayed dialog box and the test is terminated

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

13. The test software extracts the first 30 <atom:id> entries in the response and displays the following dialog box:

A screenshot of a cell phone

Description generated with very high confidence

14. The test engineer reviews the displayed list of <atom:id> entries to confirm they contain at least 48-bits

15. If the test engineer determines the displayed list of <atom:id> entries CONTAIN at least 48-bits, they select the “**Yes**” button in the displayed dialog box

16. If the test engineer determines the displayed list of <atom:id> entries does NOT contain at least 48-bits, they select the “No” button in the displayed dialog box and the test is terminated

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

17. The test software extracts the first 30 <atom:link> entries in the response and displays the following dialog box:

A screenshot of a cell phone

Description generated with very high confidence

18. The test engineer reviews the displayed list of <atom:link> entries to confirm they contain at least 48-bits

19. If the test engineer determines the displayed list of <atom:link> entries CONTAIN at least 48-bits, they select the “Yes” button in the displayed dialog box

20. If the test engineer determines they displayed list of <atom:link> entries does **NOT** contain at least 48-bits, they select the “**No**” button in the displayed dialog box and the test is terminated

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_14] Authorization and Authentication w/ Pre-Negotiated Scope

Tests: [TR\_OAD001] Verify Data Custodian rejects malformed OAuth Authorization Request

Procedure:

1. Test software starts the SoapUI “ThirdParty NEG A test Authorization Mock Service” session

2. Test software issues an HTTP GET to the test applicant's OAuth 2.0 Authorization Endpoint omitting the required response\_type query parameter

3. Upon receiving a response to the GET request, the test software:

a. Verifies the HTTP Status code is a "302"

b. Verifies the response body contains "error="

4. Test software issues an HTTP GET to the test applicant's OAuth 2.0 Authorization Endpoint omitting the required client\_id query parameter

5. Upon receiving a response to the GET request, the test software:

a. Verifies the HTTP Status code is a "302"

b. Verifies the response body contains "error="

6. Test software issues an HTTP GET to the test applicant's OAuth 2.0 Authorization Endpoint with an invalid value in the required response\_type query parameter

7. Upon receiving a response to the GET request, the test software:

a. Verifies the HTTP Status code is a "302"

b. Verifies the response body contains "error="

8. Test software issues an HTTP GET to the test applicant's OAuth 2.0 Authorization Endpoint with an invalid value in the required client\_id query parameter

9. Upon receiving a response to the GET request, the test software:

a. Verifies the HTTP Status code is a "302"

b. Verifies the response body contains "error="

10. Test software issues an HTTP GET to the test applicant's OAuth 2.0 Authorization Endpoint with an invalid value in the required redirect\_uri query parameter

11. Upon receiving a response to the GET request, the test software:

a. Verifies the HTTP Status code is a "302"

b. Verifies the response body contains "error="

12. Test software issues an HTTP GET to the test applicant's OAuth 2.0 Authorization Endpoint with an invalid value in the required scope query parameter

13. Upon receiving a response to the GET request, the test software:

a. Verifies the HTTP Status code is a "302"

b. Verifies the response body contains "error="

14. Test software terminates the SoapUI "ThirdParty NEG A test Authorization Mock Service" session

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_OAD002] Verify Data Custodian properly handles a Retail Customer who DENIES access while processing a valid Authorization Code Request

Procedure:

1. Test software starts the SoapUI "ThirdParty Authorization Mock Service" session

2. Test software reconfigures /etc/hosts to use the original /etc/hosts configuration

3. Test software activates the browser window to access the test applicant's Scope Selection Screen {dataCustodianScopeSelectionScreenURI}

4. Test software displays the following dialog box:

A screenshot of a cell phone

Description generated with very high confidence

5. The test engineer then gives remote control to the test applicant to complete the logon, select a Scope option, and DENY the authorization request

6. When the test applicant has finished the following will appear in the browser:

A screenshot of a cell phone

Description generated with very high confidence

7. When the above appears in the browser, the test engineer regains control of the system and selects "OK" in the displayed dialog box

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

8. Test software terminates the browser window

9. Test software reconfigures the /etc/hosts to use the stunnel /etc/hosts configurations

10. Test software terminates the SoapUI "ThirdParty Authorization Mock Service" session

11. Test software verifies the Mock Server response body contains "error=access\_denied"

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_OAD003] Verify Data Custodian rejects an Access Token Request with an INVALID HTTP BASIC Authorization header or a malformed Authorization Code Access Token Request

Procedure:

1. Test software starts the SoapUI "ThirdParty NEG B test Authorization Mock Service" session

2. Test software reconfigures /etc/hosts to use the original /etc/hosts configuration

3. Test software activates the browser window to access the test applicant's Scope Selection Screen {dataCustodianScopeSelectionScreenURI}

4. Test software displays the following dialog box:

A screenshot of a cell phone

Description generated with very high confidence

5. The test engineer then gives remote control to the test applicant to complete the logon, select Scope Option, and APPROVE the authorization request

6. When the test applicant has finished the following will appear in the browser:

A screenshot of a cell phone

Description generated with very high confidence

7. When the above appears in the browser, the test engineer regains control of the system and selects "OK" in the displayed dialog box

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

8. Test software terminates the browser window

9. Test software issues an HTTP POST to the test applicant's OAuth 2.0 Token Endpoint omitting the HTTP Authorization Header

12. Upon receiving a response to the POST request, the test software:

a. Verifies the HTTP Status code is a "400" or "401"

13. Test software issues an HTTP POST to the test applicant's OAuth 2.0 Token Endpoint with an invalid HTTP Authorization Header

14. Upon receiving a response to the POST request, the test software:

a. Verifies the HTTP Status code is a "400" or "401"

15. Test software issues an HTTP POST to the test applicant's OAuth 2.0 Token Endpoint with an invalid value in the required grant\_type query parameter

16. Upon receiving a response to the POST request, the test software:

a. Verifies the HTTP Status code is a "400"

17. Test software issues an HTTP POST to the test applicant's OAuth 2.0 Token Endpoint omitting the required code query parameter

18. Upon receiving a response to the POST request, the test software:

a. Verifies the HTTP Status code is a "400"

19. Test software issues an HTTP POST to the test applicant's OAuth 2.0 Token Endpoint omitting the required redirect\_uri query parameter

20. Upon receiving a response to the POST request, the test software:

a. Verifies the HTTP Status code is a "400"

21. Test software issues an HTTP POST to the test applicant's OAuth 2.0 Token Endpoint with HTTP Basic Authorization Header and required client\_id query parameter

22. Upon receiving a response to the POST request, the test software:

a. Verifies the HTTP Status code is a "400"

23. Test software issues an HTTP POST to the test applicant's OAuth 2.0 Token Endpoint with a redirect\_uri query parameter value that does not match the redirect\_uri query parameter provided in the OAuth 2.0 Authorization Code request

24. Upon receiving a response to the POST request, the test software:

a. Verifies the HTTP Status code is a "400"

25. Test software issues an HTTP POST to the test applicant's OAuth 2.0 Token Endpoint with a code query parameter value issued to another client

26. Upon receiving a response to the POST request, the test software:

a. Verifies the HTTP Status code is a "400"

27. Test software issues an HTTP POST to the test applicant's OAuth 2.0 Token Endpoint with an invalid code query parameter

28. Upon receiving a response to the POST request, the test software:

a. Verifies the HTTP Status code is a "400"

29. Test software displays the following dialog box:

A screenshot of a cell phone

Description generated with very high confidence

30. When the above appears in the browser, the test engineer selects "OK" in the displayed dialog box

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

31. Test software issues an HTTP POST to the test applicant's OAuth 2.0 Token Endpoint with a code query parameter value issued 6-minutes earlier

32. Upon receiving a response to the POST request, the test software:

a. Verifies the HTTP Status code is a "400"

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_OAD004] Verify Data Custodian rejects an Authorization Code Access Token Request containing a previously used authorization\_code

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Steps \*\*\*\*\*\*\*\*\*\*

1. Test software issues an HTTP POST OAuth 2.0 authorization\_code request to the test applicant's OAuth 2.0 Authorization Endpoint with a previously used code value in the code query parameter

2. Upon receiving a response to the POST request, the test software:

a. Verifies the HTTP status code is a "400"

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test steps.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_OAD005] Verify Data Custodian successfully issues an Authorization Code Access Token

Procedure:

1. Test software displays the following dialog box:

A screenshot of a cell phone

Description generated with very high confidence

2. Test engineer requests the test applicant to either reduce the duration an access token is valid to a few minutes, or confirm they can manually revoke access tokens

3. If the test applicant can reduce the duration an access token is valid, the test engineer waits until the test applicant has completed reducing the duration and then selects the "Yes" button in the dialog box

4. If the test applicant indicates they can manually revoke access tokens, the test engineer selects the "Yes" button in the dialog box

5. If the test applicant is unable to reduce the duration an access token is valid or manually revoke access tokens, the test engineer should point out that the current test may **PASS** but a future test may **FAIL**

6. If the test applicant decides to continue the test, the test engineer selects "Yes" in the dialog box, else the test engineer selects "No" in the dialog box and the test is terminated

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

7. Test software starts the SoapUI "ThirdParty NEG B test Authorization Mock Service" session

8. Test software reconfigures /etc/hosts to use the original /etc/hosts configuration

9. Test software activates the browser window to access the test applicant's Scope Selection Screen {dataCustodianScopeSelectionScreenURI}

10. Test software displays the following dialog box:

A screenshot of a cell phone

Description generated with very high confidence

11. The test engineer then gives remote control to the test applicant to complete the logon, select Scope Option, and APPROVE the authorization request

12. When the test applicant has finished the following will appear in the browser:

A screenshot of a cell phone

Description generated with very high confidence

13. When the above appears in the browser, the test engineer regains control of the system and selects "OK" in the displayed dialog box

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

14. Test software terminates the browser window

15. Test software reconfigures the /etc/hosts to use the stunnel /etc/hosts configurations

16. Test software terminates the SoapUI "ThirdParty NEG B test Authorization Mock Service" session

17. Test software issues an HTTP POST OAuth 2.0 token request to the test applicant's OAuth 2.0 Token Endpoint using the "authorization\_code" returned during the browser session in the code query parameter

18. Upon receiving a response to the POST, the test software:

a. Verifies the {"access\_token"} field is present in the JSON response and has data

b. Verifies the {"token\_type"} field is present in the JSON response and has data

c. Verifies the {"expires\_in"} field is present in the JSON response and has data

d. Verifies the {"refresh\_token"} field is present in the JSON response and has data

e. Verifies the {"scope"} field is present in the JSON response and has data

f. Verifies the {"resourceURI"} field is present in the JSON response and has data

g. Verifies the {"authorizationURI"} field is present in the JSON response and has data

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_OAD006] Verify Data Custodian rejects an HTTP GET request to the ApplicationInformation by ID API that contains an incorrect "access token"

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software issues an HTTP GET request to the test applicant's ApplicationInformation by ID API using a "client\_access\_token"

2. Upon receiving the GET response, the test software:

a. Verifies the HTTP status code is "403"

3. Test software issues an HTTP GET request to the test applicant's ApplicationInformation by ID API using an "access\_token"

4. Upon receiving the GET response, the test software:

a. Verifies the HTTP status code is "403"

5. Test software issues an HTTP GET request to the test applicant's ApplicationInformation by ID API without an HTTP Authorization field in the HTTP header

6. Upon receiving the GET response, the test software:

a. Verifies the HTTP Status code is "400" or “401”

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_OAD007] Verify Data Custodian rejects an HTTP GET request to the Authorization by ID API that contains an incorrect "access token"

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software issues an HTTP GET request to the test applicant's Authorization by ID API using a "registration\_access\_token"

2. Upon receiving the GET response, the test software:

a. Verifies the HTTP status code is "403"

3. Test software issues an HTTP GET request to the test applicant's Authorization by ID API using an invalid "access\_token"

4. Upon receiving the GET response, the test software:

a. Verifies the HTTP status code is "403"

5. Test software issues an HTTP GET request to the test applicant's Authorization by ID API without an HTTP Authorization field in the HTTP header

6. Upon receiving the GET response, the test software:

a. Verifies the HTTP Status code is "400" or “401”

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_OAD008] Verify Data Custodian rejects an HTTP GET request to the Authorization API that contains an incorrect "access token"

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Steps \*\*\*\*\*\*\*\*\*\*

1. Test software issues an HTTP GET request to the test applicant's Authorization API using a "registration\_access\_token"

2. Upon receiving the GET response, the test software:

a. Verifies the HTTP status code is "403"

3. Test software issues an HTTP GET request to the test applicant's Authorization API using an invalid "access\_token"

5. Upon receiving the GET response, the test software:

a. Verifies the HTTP status code is "403"

6. Test software issues an HTTP GET request to the test applicant's Authorization API without an HTTP Authorization field in the HTTP header

7. Upon receiving the GET response, the test software:

a. Verifies the HTTP Status code is "400" or “401”

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test steps.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_OAD009] Verify Data Custodian rejects an HTTP GET request to the Batch/Subscription API that contains an incorrect "access token"

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software issues an HTTP GET request to the test applicant's Batch/Subscription by ID API using a "registration\_access\_token"

2. Upon receiving the GET response, the test software:

a. Verifies the HTTP status code is "403"

3. Test software issues an HTTP GET request to the test applicant's Batch/Subscription by ID API using an invalid "access\_token"

5. Upon receiving the GET response, the test software:

a. Verifies the HTTP status code is "403"

6. Test software issues an HTTP GET request to the test applicant's Batch/Subscription by ID API without an HTTP Authorization field in the HTTP header

7. Upon receiving the GET response, the test software:

a. Verifies the HTTP Status code is "400" or “401”

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_OAD010] Verify Data Custodian rejects an HTTP GET request to the ServiceStatus API that contains an incorrect "access token"

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software issues an HTTP GET request to the test applicant's ServiceStatus API using a "registration\_access\_token"

2. Upon receiving the GET response, the test software:

a. Verifies the HTTP status code is "403"

3. Test software issues an HTTP GET request to the test applicant's ServiceStatus API using an invalid "access\_token"

4. Upon receiving the GET response, the test software:

a. Verifies the HTTP status code is "403"

5. Test software issues an HTTP GET request to the test applicant's ServiceStatus API without an HTTP Authorization field in the HTTP header

6. Upon receiving the GET response, the test software:

a. Verifies the HTTP Status code is "400" or “401”

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_OAD011] Verify Data Custodian rejects a malformed Refresh Token Access Token Request

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software issues an HTTP POST Oauth 2.0 refresh\_token to the test applicant's OAuth 2.0 Token Endpoint omitting the required grant\_type query parameter

2. Upon receiving a response to the POST request, the test software:

a. Verifies the HTTP Status code is a "400"

3. Test software issues an HTTP POST Oauth 2.0 refresh\_token to the test applicant's OAuth 2.0 Token Endpoint omitting the required refresh\_token query parameter

4. Upon receiving a response to the POST request, the test software:

a. Verifies the HTTP Status code is a "400"

5. Test software issues an HTTP POST Oauth 2.0 refresh\_token to the test applicant's OAuth 2.0 Token Endpoint with a scope query parameter that doesn't match the scope query parameter value used when the original access\_token was issued

6. Upon receiving a response to the POST request, the test software:

a. Verifies the HTTP Status code is a "400"

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_OAD012] Verify Data Custodian successfully processes a refresh\_token request

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software saves current "access\_token"

2. Test software issues an HTTP POST Oauth 2.0 refresh\_token request to the test applicant's OAuth 2.0 Authorization Server's Token endpoint

3. Upon receiving the response to the POST request, the test software:

a. Verifies the HTTP status code is "200"

b. Verifies the response contains an {access\_token} field that isn’t null

c. Verifies the response contains a {token\_type} field that isn’t null

d. Verifies the response contains a {expires\_in} field that isn’t null

e. Verifies the response contains a {scope} field that isn’t null

f. Verifies the response contains a {resourceURI} field that isn’t null

g. Verifies the response contains a {authorizationURI} field that isn’t null

h. Verifies the response contains a {refresh\_token} field that isn’t null

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_OAD013] Verify Data Custodian invalidates a “refreshed” access\_token

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software issues an HTTP GET request to the test applicant's Batch/Subscription by ID API using an "access\_token" that has been "refreshed"

2. Upon receiving the response to the GET request, the test software:

a. Verifies the HTTP status code is a "403"

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_OAD014] Verify Data Custodian rejects a malformed Client Credentials Access Token Request

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software issues an HTTP POST Oauth 2.0 client\_credentials request to the test applicant's OAuth 2.0 Authorization Server's Token endpoint omitting the required "grant\_type" query parameter

2. Upon receiving the response to the POST request, the test software:

a. Verifies the HTTP status code is a "400"

3. Test software issues an HTTP POST Oauth 2.0 client\_credentials request to the test applicant's OAuth 2.0 Authorization Server's Token endpoint with a value in the scope query parameter that does not match any of the "supported" scope values contained in the ApplicationInformation structure. (**NOTE: The client\_access\_token and registration\_access\_token scope values are "supported" scope values but MUST NOT appear in the ApplicationInformation structure).**

4. Upon receiving the response to the POST request, the test software:

a. Verifies the HTTP status code is a "400"

5. Test software issues an HTTP GET to the test applicant's ApplicationInformation by ID API

6. Upon receiving the response to the GET request, the test software:

a. Verifies the HTTP Status code is a "200"

b. Verifies the response complies with the Green Button schema

c. Verifies the response contains a {dataCustodianId} field

d. Verifies the response contains a {dataCustodianApplicationStatus} field

e. Verifies the response contains a {thirdPartyNotifyUri} field

f. Verifies the response contains a {authorizationServerAuthorizationEndpoint} field

g. Verifies the response contains a {authorizationServerTokenEndpoint} field

h. Verifies the response contains a {dataCustodianBulkRequestURI} field

i. Verifies the response contains a {dataCustodianResourceEndpoint} field

j. Verifies the response contains a {thirdPartyScopeSelectionScreenURI} field

k. Verifies the response contains a {thirdPartyUserPortalScreenURI} field

l. Verifies the response contains a {client\_secret} field

m. Verifies the response contains a {client\_name} field

n. Verifies the response contains a {client\_uri} field

o. Verifies the response contains a {client\_id} field

p. Verifies the response contains a {software\_id} field

q. Verifies the response contains a {software\_version} field

r. Verifies the response contains a {client\_id\_issued\_at} field

s. Verifies the response contains a {client\_secret\_expires\_at} field

t. Verifies the response contains a {token\_endpoint\_auth\_method} field

u. Verifies the response contains a {scope} field

v. Verifies the response contains a {grant\_types} field

w. Verifies the response contains a {response\_types} field

x. Verifies the response contains a {registration\_client\_uri} field

y. Verifies the response contains a {registration\_access\_token} field

z. Verifies the response contains a {dataCustodianScopeSelectionScreen} field

aa. Verifies the response {grant\_type} field contains a value of "client\_credentials"

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_OAD015] Verify Data Custodian responds with a valid Client Credentials Access Token Request JSON response

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software issues an HTTP POST Oauth 2.0 client\_credentials request to the test applicant's OAuth 2.0 Authorization Server's Token endpoint omitting the required "grant\_type" query parameter

2. Upon receiving the response to the POST request, the test software:

a. Verifies the HTTP status code is a "200"

b. Verifies the response contains an {access\_token} field that isn't null

c. Verifies the response contains a {token\_type} field that isn't null

d. Verifies the response contains a {expires\_in} field that isn't null

e. Verifies the response contains a {scope} field that isn't null

f. Verifies the response contains a {resourceURI} field that isn't null

g. Verifies the response contains a {authorizationURI} field that isn't null

h. Verifies the response does NOT contain a {refresh\_token} field

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_OAD016] Verify Data Custodian rejects a request containing a expired Access Token

Procedure:

1. Test software displays the following dialog box:

A screenshot of a cell phone

Description generated with very high confidence

2. If the test applicant reduced the duration of an access token in [TR\_OAD005], the test engineer waits the duration indicated by the test applicant since [TR\_OAD005] was performed and then selects "Yes" in the dialog box

3. If the test applicant indicated they can manually revoke access tokens in [TR\_OAD005], the test engineer waits for the test applicant to manually revoke the access token and then selects "Yes" in the dialog box

4. If the test applicant can neither manually revoke access tokens or the access token duration is unreasonable, the test engineer selects "No" in the dialog box and the test is terminated

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

5. Test software issues an HTTP GET request to the test applicant's Batch/Subscription by ID API

6. Upon receiving a response to the GET request, the test software:

a. Verifies the HTTP status code is "403"

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_15] Usage Summary

This Function Block is executed by the [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite if the test applicant selected to be certified for [FB\_15] Usage Summary.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. The test software performs the following tests:
   1. Verifies the received response contains:
      1. A valid <atom:id> value
      2. A valid <atom:title> value
      3. A valid <atom:published> value
      4. A valid <atom:updated> value
      5. A valid <atom:link[@rel=’self’]/@href> value
      6. A unique <atom:link[@rel-‘self’> value
      7. A valid <atom:link[@rel=’up’]/@href> value
      8. A unique <atom:link[@rel-‘up’> value
   2. Verifies the received response contains an <espi:ElectricPowerUsageSummary> or <espi:UsageSummary> entry with:
      1. A valid <espi:overallConsumptionLastPeriod> value
      2. A valid <espi:QualityOfReading> value
      3. A valid <espi:statusTimeStamp> value
   3. Verifies the received response contains a   
      <espi:billingPeriod> entry with:
      1. A valid <espi:duration> value
      2. A valid <espi:start> value
   4. Verifies the received response contains a <espi:currentBillingPeriodOverAllConsumption> entry with:
      1. A valid <espi:powerOfTenMultiplier> value
      2. A valid <espi:timeStamp> value
      3. A valid <espi:uom> value
      4. A valid <espi:value> value

Passing Result:

The **Passing Result:** section of [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite indicates if the test applicant’s application responses [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_16] Usage Summary with Cost

This Function Block is executed by the [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite if the test applicant selected to be certified for [FB\_16] Usage Summary with Cost.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. The test software performs the following tests:
   1. Verifies the received response contains an <espi:ElectricPowerUsageSummary> or <espi:UsageSummary> entry with:
      1. A valid <espi:billLastPeriod> value
      2. A valid <espi:costAdditionalLastPeriod> value
      3. A valid <espi:currency> value

Passing Result:

The **Passing Result:** section of [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite indicates if the test applicant’s application responses [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_17] Power Quality Summary

This Function Block is executed by the [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite if the test applicant selected to be certified for [FB\_17] Power Quality Summary.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. The test software performs the following tests:
   1. Verifies the received response contains a <espi:ElectricPowerQualitySummary> entry with:
      1. A valid <atom:id> value
      2. A valid <atom:title> value
      3. A valid <atom:link[@rel=’self’/@href> value
      4. A valid <atom:link[@rel=’up’/@href> value
      5. A valid <atom:published> value
      6. A valid <atom:updated> value
      7. A unique <atom:link[@rel-‘self’> value
      8. A valid <espi:flickerPlt> value
      9. A valid <espi:flickerPst> value
      10. A valid <espi:harmonicVoltage> value
      11. A valid <espi:longInterruptions> value
      12. A valid <espi:mainsVoltage> value
      13. A valid <espi:powerFrequency> value
      14. A valid <espi:rapidVoltageChanges> value
      15. A valid <espi:shortInterruptions> value
      16. A valid <espi:summaryInterval> value
      17. A valid <espi:supplyVoltageDips> value
      18. A valid <espi:supplyVoltageImbalance> value
      19. A valid <espi:supplyVoltageVariations> value
      20. A valid <espi:tempOverVoltage> value
      21. A valid <espi:summaryInterval> entry with:
          1. A valid <espi:duration> and <espi:start> value

Passing Result:

The **Passing Result:** section of [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite indicates if the test applicant’s application responses [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_27] Usage Summary with Demands and Previous Day Attributes

This Function Block is executed by the [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite if the test applicant selected to be certified for [FB\_27] Usage Summary with Demands and Previous Day Attributes.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. The test software performs the following tests:
2. Verifies the received response contains an <espi:ElectricPowerUsageSummary> or <espi:UsageSummary> entry with:
   * 1. A valid <espi:ratchetDemand> value
     2. A valid <espi:ratchetDemandPeriod> value
     3. A valid <espi:peakDemand> value
     4. A <espi:currentDayOverallConsumption> entry with:
        1. A valid <espi:powerOfTenMultiplier> and <espi:uom> value
     5. A <espi:previousDayOverallConsumption> entry with:
        1. A valid <espi:powerOfTenMultiplier> and <espi:uom> value
     6. A <espi:ratchetDemand> entry with:
        1. A valid <espi:powerOfTenMultiplier> and <espi:uom> value
     7. A <espi:ratchetDemandPeriod> entry with:
        1. A valid <espi:duration> and <espi:start> value
     8. A <espi:peakDemand> entry with:
        1. A valid <espi:powerOfTenMultiplier> and <espi:uom> value

Passing Result:

The **Passing Result:** section of [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite indicates if the test applicant’s application responses [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_28] Usage Summary Costs for Current Billing Period

This Function Block is executed by the [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite if the test applicant selected to be certified for [FB\_28] Usage Summary Costs for Current Billing Period.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. The test software performs the following tests:
   1. Verifies the received response contains an <espi:ElectricPowerUsageSummary> or <espi:UsageSummary> entry with:
      1. A valid <espi:billToDate> value

Passing Result:

The **Passing Result:** section of [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite indicates if the test applicant’s application responses [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_29] Temperature

This Function Block is executed by the [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite if the test applicant selected to be certified for [FB\_29] Temperature.

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. The test software performs the following tests:
   1. Verifies the received response contains at least one <espi:ReadingType> entry with:
      1. A <espi:kind> value of 46 (Temperature) and <espi:uom> value of 6 (Kelvin)

Passing Result:

The **Passing Result:** section of [FB\_03] Core Green Button Connect My Data – FND005 [POS][A] Authorized GET access to Batch/Subscription feed Test Suite indicates if the test applicant’s application responses [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_34] SFTP for Bulk

Tests: [TR\_SFTP001] Verify Third Party notification of Bulk data by Data Custodian and data transfer using SFTP

Procedure:

1. Test software issues an HTTP GET request to the test applicant's Authorization API

2. Upon receiving a response to the GET request, the test software:

a. Verifies the HTTP status code is a "200"

3. If the above verification fails, the test is terminated

4. Test software starts the SoapUI "ThirdParty Notification Mock Service"

5. Test software displays the following dialog box:

A screenshot of a cell phone

Description generated with very high confidence

6. Test engineer waits for test applicant to issue a SFTP Bulk transfer request to the Third Party Notification endpoint and then selects "Yes" in the dialog box

7. If the test applicant is unable to issue a SFTP Bulk transfer request to the Third Party Notification endpoint, the test engineer selects "No" in the dialog box and the test is terminated

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

8. Test software stops the SoapUI "ThirdParty Notification Mock Service"

9. Test software receives the message sent to the Third Party Notification endpoint, and performs the following:

a. Verifies the body of the received message is NOT empty

b. Verifies the body of the received message does NOT cause a XML processing exception

c. Verifies the received message provides an SFTP based URI address

10. If any of the above verifications fail, the test is terminated

11. Test software issues a SFTP GET request to the SFTP URI address contained in the body of the received Third Party Notification endpoint message

12. Upon receiving the response to the SFTP GET request, the test software:

a. Verifies the retrieved files do NOT cause an XML processing exception

b. Verifies the retrieved files contain a <feed> tag

c. Verifies the retrieved files contain at least one <entry> tag

d. Verifies the retrieved files comply with the Green Button schema

e. Verifies a valid authorization exist for each <entry> in the retrieved files

f. Verifies the scope of each valid authorization contains a BR=Bulkid value

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_35] REST for Bulk

Tests: [TR\_RBK001] Verify Third Party notification of Bulk data by DataCustodian and data transfer using REST GET with client\_access\_token

Procedure:

1. Test software issues an HTTP GET request to the test applicant's Authorization API

2. Upon receiving a response to the GET request, the test software:

a. Verifies the HTTP status code is a "200"

3. If the above verification fails, the test is terminated

4. Test software starts the SoapUI "ThirdParty Notification Mock Service"

5. Test software displays the following dialog box:

A screenshot of a cell phone

Description generated with very high confidence

6. Test engineer waits for test applicant to issue a REST Bulk transfer request to the Third Party Notification endpoint and then selects "Yes" in the dialog box

7. If the test applicant is unable to issue a REST Bulk transfer request to the Third Party Notification endpoint, the test engineer selects "No" in the dialog box and the test is terminated

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

8. Test software stops the SoapUI "ThirdParty Notification Mock Service"

9. Test software receives the message sent to the Third Party Notification endpoint, and performs the following:

a. Verifies the body of the received message is NOT empty

b. Verifies the body of the received message does NOT cause a XML processing exception

c. Verifies the received message provides an HTTP or HTTPS based URI address

10. If any of the above verifications fail, the test is terminated

11. Test software issues an HTTP GET request to the HTTP/HTTPS URI address contained in the body of the received Third Party Notification endpoint message using an “client\_access\_token”

12. Upon receiving the response to the HTTP/HTTPS GET request, the test software:

a. Verifies the retrieved files do NOT cause an XML processing exception

b. Verifies the retrieved files contain a <feed> tag

c. Verifies the retrieved files contain at least one <entry> tag

d. Verifies the retrieved files comply with the Green Button schema

e. Verifies a valid authorization exist for each <entry> in the retrieved files

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_RBK002] Verify Third Party notification of Bulk data and forbidden access using access\_token

Procedure:

1. Test software starts the SoapUI "ThirdParty Notification Mock Service"

2. Test software displays the following dialog box:

A screenshot of a cell phone

Description generated with very high confidence

3. Test engineer waits for test applicant to issue a REST Bulk transfer request to the Third Party Notification endpoint and then selects "Yes" in the dialog box

4. If the test applicant is unable to issue a REST Bulk transfer request to the Third Party Notification endpoint, the test engineer selects "No" in the dialog box and the test is terminated

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

5. Test software stops the SoapUI "ThirdParty Notification Mock Service"

6. Test software receives the message sent to the Third Party Notification endpoint, and performs the following:

a. Verifies the body of the received message is NOT empty

b. Verifies the body of the received message does NOT cause a XML processing exception

c. Verifies the received message provides an HTTP or HTTPS based URI address

7. If any of the above verifications fail, the test is terminated

8. Test software issues an HTTP GET request to the HTTP/HTTPS URI address contained in the body of the received Third Party Notification endpoint message using an “access\_token”

9. Upon receiving the response to the HTTP/HTTPS GET request, the test software:

a. Verifies the HTTP status code is a “403”

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above test procedure.

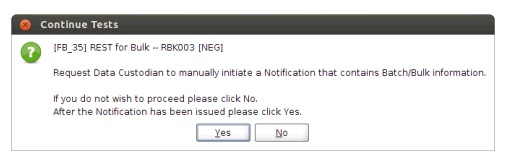
Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_RBK003] Verify Third Party notification of Bulk data and forbidden access using registration\_access\_token

Procedure:

1. Test software starts the SoapUI "ThirdParty Notification Mock Service"

2. Test software displays the following dialog box:



3. Test engineer waits for test applicant to issue a REST Bulk transfer request to the Third Party Notification endpoint and then selects "Yes" in the dialog box

4. If the test applicant is unable to issue a REST Bulk transfer request to the Third Party Notification endpoint, the test engineer selects "No" in the dialog box and the test is terminated

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

5. Test software stops the SoapUI "ThirdParty Notification Mock Service"

6. Test software receives the message sent to the Third Party Notification endpoint, and performs the following:

a. Verifies the body of the received message is NOT empty

b. Verifies the body of the received message does NOT cause a XML processing exception

c. Verifies the received message provides an HTTP or HTTPS based URI address

7. If any of the above verifications fail, the test is terminated

8. Test software issues an HTTP GET request to the HTTP/HTTPS URI address contained in the body of the received Third Party Notification endpoint message using an “registration\_access\_token”

9. Upon receiving the response to the HTTP/HTTPS GET request, the test software:

a. Verifies the HTTP status code is a “403”

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above test procedure.

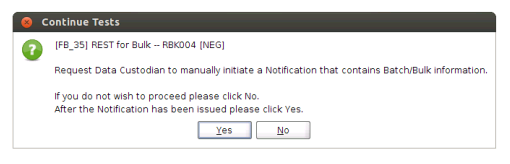
Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_RBK004] Verify Third Party notification of Bulk data and forbidden access using empty access token

Procedure:

1. Test software starts the SoapUI "ThirdParty Notification Mock Service"

2. Test software displays the following dialog box:



3. Test engineer waits for test applicant to issue a REST Bulk transfer request to the Third Party Notification endpoint and then selects "Yes" in the dialog box

4. If the test applicant is unable to issue a REST Bulk transfer request to the Third Party Notification endpoint, the test engineer selects "No" in the dialog box and the test is terminated

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

5. Test software stops the SoapUI "ThirdParty Notification Mock Service"

6. Test software receives the message sent to the Third Party Notification endpoint, and performs the following:

a. Verifies the body of the received message is NOT empty

b. Verifies the body of the received message does NOT cause a XML processing exception

c. Verifies the received message provides an HTTP or HTTPS based URI address

7. If any of the above verifications fail, the test is terminated

8. Test software issues an HTTP GET request to the HTTP/HTTPS URI address contained in the body of the received Third Party Notification endpoint message without an Authorization entry in the HTTP header

9. Upon receiving the response to the HTTP/HTTPS GET request, the test software:

a. Verifies the HTTP status code is a “403”

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_37] Query Parameters

Tests: [TR\_QRY001] Verify Data Custodian Supports published-min and published-max Query Parameters

Procedure:

1. The test software checks to see if an Authorization has been issued by the Data Custodian

2. If the test software determines an Authorization exist, it continues execution at step 7.

3. If the test software determines an Authorization request is required, it displays a dialog box containing:

A screenshot of a cell phone

Description generated with very high confidence

4. The test engineer then gives remote control to the test applicant to complete the logon and authorization process.

5. When the test applicant has finished the following will appear in the browser:

A screenshot of a cell phone

Description generated with very high confidence

6. When the above appears in the browser, the test engineer regains control of the system and selects "OK" in the displayed dialog box.

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

7. The test software issues an HTTP GET request to the Batch/Subscription by ID API using the active Authorization's resourceURI value

8. Upon receiving the response to the HTTP GET, the test software:

a. Verifies the HTTP status code is a "200"

b. Verifies the received response does not cause an XML processing error

c. Verifies the received response contains at least one <feed> entry

d. Verifies the received response <feed> entries have at least 1 <IntervalBlock> entry

e. Verifies the received response complies with the Green Button schema

9. If any of the above verifications FAIL, the test is terminated

10. The test software counts the number of <IntervalBlock> entries and determines the earliest and oldest dates of <IntervalBlock> entries contained in the received response

11. The test software re-issues an HTTP GET request to the Batch/Subscription by ID API adding published-min and published-max query parameters adjusted to return fewer IntervalBlock entries than original issued HTTP GET to the Batch/Subscription by ID API

12. Upon receiving the response to the HTTP GET, the test software:

a. Verifies the HTTP status code is a "200"

b. Verifies the received response complies with the Green Button schema

c. Verifies the received response does not cause an XML processing error

d. Verifies the received response contains at least one <feed> entry

e. Verifies the received response contains at least 1 <IntervalBlock> entry

f. Verifies the received response contains 2 fewer <IntervalBlock> entries than the response returned by the originally issued HTTP GET to the Batch/Subscription by ID API

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_QRY002] Verify Data Custodian Validate all Query Parameters

Procedure:

1. The test software checks to see if an Authorization has been issued by the Data Custodian

2. If the test software determines an Authorization exist, it continues execution at step 7.

3. If the test software determines an Authorization request is required, it displays a dialog box containing:

A screenshot of a cell phone

Description generated with very high confidence

4. The test engineer then gives remote control to the test applicant to complete the logon and authorization process.

5. When the test applicant has finished the following will appear in the browser:

A screenshot of a cell phone

Description generated with very high confidence

6. When the above appears in the browser, the test engineer regains control of the system and selects "OK" in the displayed dialog box.

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

7. The test software issues an HTTP GET request to the Batch/Subscription by ID API using the active Authorization's resourceURI value with an ALPHA published-min query parameter value

8. Upon receiving the response to the HTTP GET, the test software:

a. Verifies the HTTP status code is a "400"

9. If the above verify FAILs, the test is terminated

10. The test software issues an HTTP GET request to the Batch/Subscription by ID API using the active Authorization's resourceURI value with an invalid date published-min query parameter value

11. Upon receiving the response to the HTTP GET, the test software:

a. Verifies the HTTP status code is a "400"

12. If the above verify FAILs, the test is terminated

13. The test software issues an HTTP GET request to the Batch/Subscription by ID API using the active Authorization's resourceURI value with an ALPHA published-max query parameter value

14. Upon receiving the response to the HTTP GET, the test software:

a. Verifies the HTTP status code is a "400"

15. If the above verify FAILs, the test is terminated

16. The test software issues an HTTP GET request to the Batch/Subscription by ID API using the active Authorization's resourceURI value with an invalid date published-max query parameter value

17. Upon receiving the response to the HTTP GET, the test software:

a. Verifies the HTTP status code is a "400"

18. If the above verify FAILs, the test is terminated

19. The test software issues an HTTP GET request to the Batch/Subscription by ID API using the active Authorization's resourceURI value with an ALPHA updated-min query parameter value

20. Upon receiving the response to the HTTP GET, the test software:

a. Verifies the HTTP status code is a "400"

21. If the above verify FAILs, the test is terminated

22. The test software issues an HTTP GET request to the Batch/Subscription by ID API using the active Authorization's resourceURI value with an invalid date updated-min query parameter value

23. Upon receiving the response to the HTTP GET, the test software:

a. Verifies the HTTP status code is a "400"

24. If the above verify FAILs, the test is terminated

25. The test software issues an HTTP GET request to the Batch/Subscription by ID API using the active Authorization's resourceURI value with an ALPHA updated-max query parameter value

26. Upon receiving the response to the HTTP GET, the test software:

a. Verifies the HTTP status code is a "400"

27. If the above verify FAILs, the test is terminated

28. The test software issues an HTTP GET request to the Batch/Subscription by ID API using the active Authorization's resourceURI value with an invalid date updated-max query parameter value

29. Upon receiving the response to the HTTP GET, the test software:

a. Verifies the HTTP status code is a "400"

30. If the above verify FAILs, the test is terminated

31. The test software issues an HTTP GET request to the Batch/Subscription by ID API using the active Authorization's resourceURI value with an ALPHA max-results query parameter value

32. Upon receiving the response to the HTTP GET, the test software:

a. Verifies the HTTP status code is a "400"

33. If the above verify FAILs, the test is terminated

34. The test software issues an HTTP GET request to the Batch/Subscription by ID API using the active Authorization's resourceURI value with an ALPHA start-index query parameter value

35. Upon receiving the response to the HTTP GET, the test software:

a. Verifies the HTTP status code is a "400"

36. If the above verify FAILs, the test is terminated

37. The test software issues an HTTP GET request to the Batch/Subscription by ID API using the active Authorization's resourceURI value with an ALPHA depth query parameter value

38. Upon receiving the response to the HTTP GET, the test software:

a. Verifies the HTTP status code is a "400"

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test steps.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_39] PUSH Model

Tests: [TR\_PSH001] Verify Data Custodian Can Send a Notification Message that Contains a Valid ApplicationInformation URI

Procedure:

1. Test software activates the SoapUI "Batch List Notification Mock Service"

2. Test software displays the following dialog box:

A screenshot of a cell phone

Description generated with very high confidence

3. The test engineer requests the test applicant to initiate a Notification request that contains an ApplicationInformation resource URI

4. After the test applicant indicates the notification has been sent, the test engineer selects 'Yes' in the dialog box

5. If the test applicant indicates they are unable to initiate the Notification request, the test engineer selects "No" in the dialog box and the test is terminated

6. Test software stops the SoapUI "Batch List Notification Mock Service"

7. Upon receiving the Notification message, the test software:

a. Verifies a Notification message was received

b. Verifies the received Notification message does not cause an XML processing error

c. Verifies the received Notification message contains a valid resource URI

d. Verifies the received Notification message URI contains "https://", or "sftp://"

8. If the received Notification message URI contains "https://", the test software issues an HTTP GET to the URI contained in the response

9. Upon receiving the response to the HTTP GET, the test software:

a. Verifies the HTTP status code is a "200"

b. Verifies the response does not cause an XML processing error

c. Verifies the response contains at least one <ApplicationInformation> entry

d. Verifies the response complies with the Green Button schema

10. If the received Notification message URI contains "sftp://", the test software issues an SFTP GET to the URI contained in the response

11. Upon receiving the response to the HTTP GET, the test software:

a. Verifies the response does not cause an XML processing error

b. Verifies the response contains at least one <ApplicationInformation> entry

c. Verifies the response complies with the Green Button schema

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_PSH002] Verify Data Custodian Can Send a Notification Message that Contains a Valid Authorization URI

Procedure:

1. Test software activates the SoapUI "Batch List Notification Mock Service"

2. Test software displays the following dialog box:

A screenshot of a cell phone

Description generated with very high confidence

3. The test engineer requests the test applicant to initiate a Notification request that contains an Authorization resource URI

4. After the test applicant indicates the notification has been sent, the test engineer selects 'Yes' in the dialog box

5. If the test applicant indicates they are unable to initiate the Notification request, the test engineer selects "No" in the dialog box and the test is terminated

6. Test software stops the SoapUI "Batch List Notification Mock Service"

7. Upon receiving the Notification message, the test software:

a. Verifies a Notification message was received

b. Verifies the received Notification message does not cause an XML processing error

c. Verifies the received Notification message contains a valid resource URI

d. Verifies the received Notification message URI contains "https://", or "sftp://"

8. If the received Notification message URI contains "https://", the test software issues an HTTP GET to the URI contained in the response

9. Upon receiving the response to the HTTP GET, the test software:

a. Verifies the HTTP status code is a "200"

b. Verifies the response does not cause an XML processing error

c. Verifies the response contains at least one <Authorization> entry

d. Verifies the response complies with the Green Button schema

10. If the received Notification message URI contains "sftp://", the test software issues an SFTP GET to the URI contained in the response

11. Upon receiving the response to the HTTP GET, the test software:

a. Verifies the response does not cause an XML processing error

b. Verifies the response contains at least one <Authorization> entry

c. Verifies the response complies with the Green Button schema

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_PSH003] Verify Data Custodian Can Send a Notification Message that Contains a Valid Batch/Subscription URI

Procedure:

1. Test software activates the SoapUI "Batch List Notification Mock Service"

2. Test software displays the following dialog box:

A screenshot of a cell phone

Description generated with very high confidence

3. The test engineer requests the test applicant to initiate a Notification request that contains an Batch/Subscription resource URI

4. After the test applicant indicates the notification has been sent, the test engineer selects 'Yes' in the dialog box

5. If the test applicant indicates they are unable to initiate the Notification request, the test engineer selects "No" in the dialog box and the test is terminated

6. Test software stops the SoapUI "Batch List Notification Mock Service"

7. Upon receiving the Notification message, the test software:

a. Verifies a Notification message was received

b. Verifies the received Notification message does not cause an XML processing error

c. Verifies the received Notification message contains a valid resource URI

d. Verifies the received Notification message URI contains "https://", or "sftp://"

8. If the received Notification message URI contains "https://", the test software issues an HTTP GET to the URI contained in the response

9. Upon receiving the response to the HTTP GET, the test software:

a. Verifies the HTTP status code is a "200"

b. Verifies the response does not cause an XML processing error

c. Verifies the response contains at least one <Batch/Subscription> entry

d. Verifies the response complies with the Green Button schema

10. If the received Notification message URI contains "sftp://", the test software issues an SFTP GET to the URI contained in the response

11. Upon receiving the response to the HTTP GET, the test software:

a. Verifies the response does not cause an XML processing error

b. Verifies the response contains at least one <Subscription> entry

c. Verifies the response complies with the Green Button schema

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_PSH004] Verify Data Custodian Detects Notification Message Delivery Failures

Procedure:

1. Test software stops the stunnel proxy

2. Test software displays the following dialog box:

A screenshot of a cell phone

Description generated with very high confidence

3. The test engineer requests the test applicant to initiate a Notification request and wait to be notified the message delivery failed

4. After the test applicant indicates they recorded the Notification message delivery failure, the test engineer selects 'Yes' in the dialog box

5. If the test applicant indicates they are unable to initiate the Notification request or they did **NOT** record the Notification message delivery failed, the test engineer selects "No" in the dialog box

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_40] Offline Authorization

Tests: [TR\_OFA001] Verify the test applicant can provide valid XML files containing ApplicationInformation and Authorization entries to Third Party vendors as part of the registration process.

Procedure:

NOTE: Test steps do not meet stated Test Purpose/Requirements

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_41] Authorized PUT/DELETE ApplicationInformation Resource

Tests: [TR\_MGA001] Verify Data Custodian's Authorization API supports PUT of the Authorization with the use of the proper OAuth 2.0 access\_token

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software issues an HTTP GET request to the Authorization API by Id

2. Upon receiving a response to the GET request, the test software:

a. Validates the response contains a 200 status code

b. Validates the response contains an Authorization <entry>

3. Test software copies the contents of the response

4. Test software issues an HTTP PUT request to the Authorization API by Id

5. Upon receiving a response to the PUT request, the test software:

a. Validates the response contains a 200 status code

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_MGA002] Verify Data Custodian's Authorization API supports DELETE of the Authorization with the use of the proper OAuth 2.0 access\_token

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software issues an HTTP GET request to the Authorization API by Id

2. Upon receiving a response to the GET request, the test software:

a. Validates the response contains a 200 status code

b. Validates the response contains an Authorization <entry>

3. Test software copies the contents of the response

4. Test software issues an HTTP DELETE request to the Authorization API by Id

5. Upon receiving a response to the DELETE request, the test software:

a. Validates the response contains a 200 status code

6. Test software issues an HTTP PUT request to the Authorization API by id

7. Upon receiving a response to the PUT request, the test software:

a. Validates the response contains a 200 status code

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_MGA003] Verify Data Custodian does not allow access to the Authorization API without use of the proper OAuth 2.0 access\_token

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software issues an HTTP GET request to the Authorization API by Id

2. Upon receiving a response to the GET request, the test software:

a. Validates the response contains a 200 status code

b. Validates the response contains an Authorization <entry>

3. Test software copies the contents of the response

4. Test software issues an HTTP DELETE request to the Authorization API by Id using the registration\_access\_token

5. Upon receiving a response to the DELETE request, the test software:

a. Validates the response contains a 403 status code

6. Test software issues an HTTP DELETE request to the Authorization API by Id using the access\_token

7. Upon receiving a response to the DELETE request, the test software:

a. Validates the response contains a 403 status code

8. Test software issues an HTTP DELETE request to the Authorization API by Id without providing an access\_token

9. Upon receiving a response to the DELETE request, the test software:

a. Validates the response contains a 403 status code

10. Test software issues an HTTP PUT request to the Authorization API by Id using the registraion\_access\_token

11. Upon receiving a response to the PUT request, the test software:

a. Validates the response contains a 403 status code

12. Test software issues an HTTP PUT request to the Authorization API by Id using the access\_token

13. Upon receiving a response to the PUT request, the test software:

a. Validates the response contains a 403 status code

14. Test software issues an HTTP PUT request to the Authorization API by Id without providing an access\_token

15. Upon receiving a response to the PUT request, the test software:

a. Validates the response contains a 403 status code

16. Test software issues an HTTP PUT request to the Authorization API by id using the data\_custodian\_access\_token

17. Upon receiving a response to the PUT request, the test software:

a. Validates the response contains a 200 status code

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

[FB\_44] PUT/DELETE Authorization Resource

Tests: [TR\_MGT001] Verify Data Custodian's ApplicationInformation API supports PUT of the ApplicationInformation with the use of the proper OAuth 2.0 access\_token

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software issues an HTTP GET request to the ApplicationInformation API by Id

2. Upon receiving a response to the GET request, the test software:

a. Validates the response contains a 200 status code

b. Validates the response contains an ApplicationInformation <entry>

3. Test software copies the contents of the response

4. Test software issues an HTTP PUT request to the ApplicationInformation API by Id using the client\_access\_token

5. Upon receiving a response to the PUT request, the test software:

a. Validates the response contains a 200 status code

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_MGT002] Verify Data Custodian's ApplicationInformation API supports DELETE of the ApplicationInformation with the use of the proper OAuth 2.0 access\_token

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software issues an HTTP GET request to the ApplicationInformation API by Id

2. Upon receiving a response to the GET request, the test software:

a. Validates the response contains a 200 status code

b. Validates the response contains an ApplicationInformation <entry>

3. Test software copies the contents of the response

4. Test software issues an HTTP DELETE request to the ApplicationInformation API by Id using the registration\_access\_token

5. Upon receiving a response to the PUT request, the test software:

a. Validates the response contains a 200 status code

6. Test software issues an HTTP PUT request to the ApplicationInformation API by id using the registration\_access\_token

7. Upon receiving a response to the PUT request, the test software:

a. Validates the response contains a 200 status code

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Tests: [TR\_MGT003] Verify Data Custodian does not allow access to the ApplicationInformation API without use of the proper OAuth 2.0 access\_token

Procedure:

\*\*\*\*\*\*\*\*\*\* Automated Test Procedure \*\*\*\*\*\*\*\*\*\*

1. Test software issues an HTTP GET request to the ApplicationInformation API by Id

2. Upon receiving a response to the GET request, the test software:

a. Validates the response contains a 200 status code

b. Validates the response contains an ApplicationInformation <entry>

3. Test software copies the contents of the response

4. Test software issues an HTTP PUT request to the ApplicationInformation API by Id using the registration\_access\_token

5. Upon receiving a response to the PUT request, the test software:

a. Validates the response contains a 403 status code

6. Test software issues an HTTP PUT request to the ApplicationInformation API by Id using the access\_token

7. Upon receiving a response to the PUT request, the test software:

a. Validates the response contains a 403 status code

8. Test software issues an HTTP PUT request to the ApplicationInformation API by Id without providing an access\_token

9. Upon receiving a response to the PUT request, the test software:

a. Validates the response contains a 403 status code

10. Test software issues an HTTP DELETE request to the ApplicationInformation API by Id using the client\_access\_token

11. Upon receiving a response to the DELETE request, the test software:

a. Validates the response contains a 403 status code

12. Test software issues an HTTP DELETE request to the ApplicationInformation API by Id using the access\_token

13. Upon receiving a response to the DELETE request, the test software:

a. Validates the response contains a 403 status code

14. Test software issues an HTTP DELETE request to the ApplicationInformation API by Id without providing an access\_token

15. Upon receiving a response to the DELETE request, the test software:

a. Validates the response contains a 403 status code

16. Test software issues an HTTP PUT request to the ApplicationInformation API by id using the data\_custodian\_access\_token

17. Upon receiving a response to the PUT request, the test software:

a. Validates the response contains a 200 status code

Passing Result:

The test applicant's application response(s) [did/did not] meet all Validation requirements by indication of "**FINISHED**" notification after completion of above automated test procedure.

Tester Initials: \_\_\_\_\_\_\_\_ Submitter Initials: \_\_\_\_\_\_\_\_

Appendix A

Procedure to Verify Root Certificate Complies with WebTrust or ETSI Audit Standards

The following procedure is used to verify the Root Certificate of the Certificate Chain has been audited and found to comply with WebTrust or ETSI audit standards. The “*Mozilla Included CA Certificate List*” is used to perform the verification.

Procedure:

1. The test engineer opens a browser and access <https://wiki.mozilla.org/CA/Included_Certificates> website

A screenshot of a social media post

Description generated with very high confidence

2. The test engineer clicks on the “Included CA Certificates (HTML)” link

A screenshot of a social media post

Description generated with very high confidence

3. The following webpage appears:

A screenshot of a computer

Description generated with very high confidence

4. The test operator then locates the Root Certificate Name shown in the [FB\_13] Security and Privacy – TC009 dialog box:

A screenshot of a social media post

Description generated with very high confidence

5. If the webpage search is successful, the cell containing the Certificate Name will appear highlighted:

A screenshot of a computer

Description generated with very high confidence

6. The test operator then moves to the last 4-columns of the line containing the Certificate name which show the Auditor, Standard Audit Type, and Standard Audit Statement Date:

A screenshot of a computer

Description generated with very high confidence

7. If the test engineer confirms the Root Certificate was found to comply with either the WebTrust or ETSI audit standard, they select “Yes” in the dialog box

8. If the test engineer is either unable to locate the Certificate Name on the Mozilla Included CA Certificate List website, or they are unable to confirm the Root Certificate complies with either the WebTrust or ETSI audit standard, they select “No” in the dialog box