Certification Criteria - iClient - Retail/Hospitality

This document specifies the minimum criteria that needs to be met by the implementation of each the given features from the Tyro integration feature-set, for the POS integration to be considered fit for certification.

Given below is a summary of the retail and hospitality feature-set along with the requirement level associated with them:

Features	**Requirement level (Optional/Compulsory/Recommended)**	
	Hospitality	**Retail**
Basic Purchase/Refund features		
Integrated Purchases	**Compulsory**	**Compulsory**
Integrated Refunds	**Compulsory**	**Compulsory**
Integrated Cashout	Optional	Optional
Integrated Receipts	**Highly Recommended**	**Highly Recommended**
Integrated Surcharging	**Highly Recommended**	**Highly Recommended**
Tyro Settings Page	**Compulsory**	**Compulsory**
End of Day features		
Integrated Manual Settlement	Optional	Optional
Integrated Reports	**Highly Recommended**	Optional
Value-add features		
Integrated Pre-authorisation	Optional	Optional
Integrated Split Payments	**Highly Recommended**	Optional
Integrated Tipping	**Highly Recommended**	Optional
Integrated Bar-Tabs	Optional	Optional
Headless Pairing	Optional	Optional
Headless transactional User Interface (UI)	Optional	Optional
Technical Features		

POS Information	**Compulsory**	**Compulsory**
API Key Configuration	**Compulsory**	**Compulsory**

Feature descriptions and examples

Each of the Tyro integration features from the feature set must meet a certain criteria in terms of functionality, workflow, and outputs or end results delivered to be eligible for certification for a given feature. This section defines each feature in terms of its functionality and gives specific examples of the deliverables that the POS must provide, as well as the ideal end result that is to be achieved.

Integrated Purchases

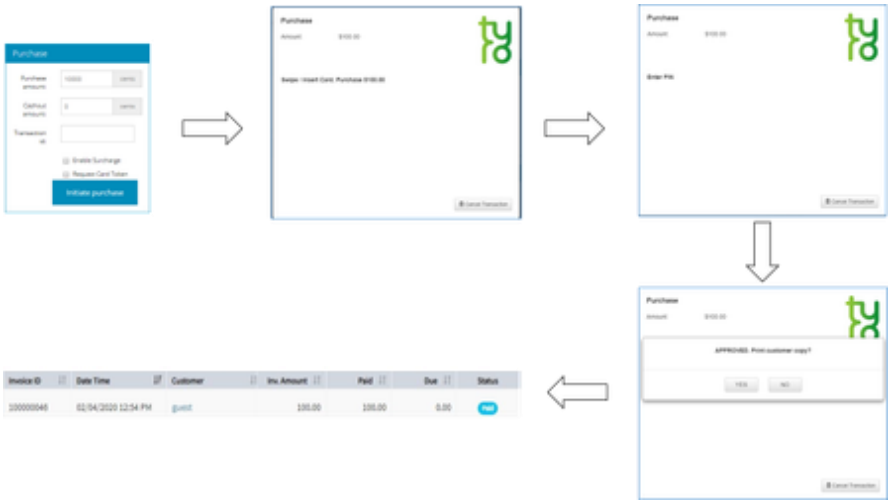
Integrated purchases are a core, compulsory functionality of the TYRO Integrated EFTPOS product that each Point-of-sale must provide to the merchants. The Integrated purchase feature entails initiating and sending a transaction to the Tyro terminal through the POS interface, the Tyro terminal processing the transaction following the card being presented via tap, swipe, or insert, and the POS registering the result or outcome of the transaction, all the while conducting error-handling and delivering status-updates to keep the user informed.

In terms of workflow and deliverables, the Integrated Purchase feature criteria is:

1. The purchase is initiated through the POS software using the `initiatePurchase()` function, with the amount specified in the `amount` parameter of the `requestParams` object, with any cashout amount (which is optional) to be included in the `cashout` request parameter.
2. The POS does not round off the sale amount.
3. The purchase amount should be stored in an integer variable instead of a double to avoid any floating-point precision issues.
4. The purchase transaction is sent through to the Tyro terminal.
5. The Tyro purchase user-interface (dialog box) is displayed on the POS device.
6. The terminal shows the prompts allowing the cardholder to tap, swipe, or insert the card.
7. The terminal processes the transaction, upon doing so, and depending on the outcome of the transaction the POS does one of the following:
    1. Displays the result of the transaction, please ensure that this is being obtained from the `result` of the `transactionCompleteCallback` only.
    2. In case of transaction being approved i.e. `result` containing APPROVED, the transaction is finalized on the POS UI.
    3. Transactions with the result: DECLINED, CANCELLED, SYSTEM ERROR, or REVERSED should be handled appropriately on the POS UI, and not be finalized on the POS.
8. The POS UI is cleared once the transaction has been completed, and the POS is ready to process the next transaction. Please ensure that this does not happen until the

`transactionCompleteCallback` delivers the result or outcome of the transaction through the `transactionData.result` object, this is to ensure graceful operation and better readability for the POS user.

9. Please note that your integration must **not be critically dependent** on card type fields i.e., `cardType` field of the `transactionCompleteCallback`, since these are subject to change.



As seen above, the POS UI is used to initiate a purchase, and the flow through the Tyro transaction UI screens to the paid invoice being stored in the POS can be seen.

Please click here to navigate to the top of the page.

Integrated Refunds

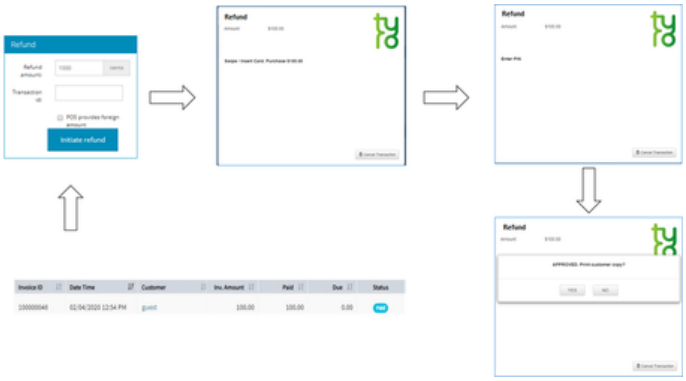
Integrated refunds are also a mandatory feature of the TYRO Integrated EFTPOS product that each Point-of-sale software must make available to the merchants. The Integrated refund feature entails initiating and sending a refund transaction to the Tyro terminal through the POS interface after any security verification deemed necessary on the POS UI, the Tyro terminal processing the transaction following verification via pin and the card being presented via tap, swipe, or insert, and the POS registering the result or outcome of the refunded transaction, all the while conducting error-handling (disallowing overdraw, cashout) and delivering status-updates to keep the user informed.

In terms of criteria and deliverables, the Integrated Refund feature essentially involves:

1. The refund being initiated through the POS software, including any surcharge (which is compulsory if the transaction contained a surcharge amount and the integrated surcharge feature has been developed), please ensure that the surcharge-inclusive amount is

specified in the `amount` parameter of the `requestParams` object of the `initiateRefund()` function.

2. The POS does not round off the sale amount.
3. The purchase amount should be stored in an integer variable instead of a double to avoid any floating-point precision issues.
4. The refund transaction being sent through to the Tyro terminal.
5. The Tyro refund user-interface (dialog box) being displayed on the POS device.
6. The terminal shows the prompts allowing the merchant to enter the terminal refund pin, and the cardholder to then tap, swipe, or insert the card.
7. The terminal processes the transaction upon doing so, and depending on the outcome of the transaction the POS does the following:
    1. Displays the result of the transaction, please ensure that this is being obtained from the `result` of the `transactionCompleteCallback` only.
    2. In case of transaction being APPROVED, the transaction is finalized on the POS UI.
    3. Transactions with the result: DECLINED, CANCELLED, SYSTEM ERROR, or REVERSED should be handled appropriately on the POS UI, and not be finalized on the POS.
8. The POS UI is cleared once the transaction has been completed, and the POS is ready to process the next transaction. Please ensure that this does not happen until the `transactionCompleteCallback` delivers the result or outcome of the transaction through the `transactionData.result` object, this is to ensure graceful operation and better readability for the POS user.
9. Please note that your integration must **not be critically dependent** on card type fields i.e., `cardType` field of the `transactionCompleteCallback`, since these are subject to change.



The image above, shows the workflow wherein an approved stored transaction can be used to initiate a refund, and then the subsequent transaction workflow allows the transaction to be completed.

Please click here to navigate to the top of the page.

Integrated Cashout

Integrated Cashout is an optional feature provided by Tyro Integrated EFTPOS feature-set that allows cashout amounts to be applied, handled, and reconciled as part of integrated **purchase** transactions from within the POS. This leads to the cashout amount being registered end-to-end from within the POS, i.e., on the terminal and the Tyro system, as well as the POS front-end and sale register. The ideal workflow is:

1. An integrated purchase transaction is initiated from within the POS system, and a cashout amount is added onto it. The purchase is initiated through the POS software using the `initiatePurchase()` function, with the amount specified in the `amount` parameter of the `requestParams` object, with the cashout amount to be included in the `cashout` request parameter of the `requestParams` object.
2. The transaction is transmitted to the terminal where the purchase amount and the cashout amounts are recognized individually. The customer is prompted to swipe or insert their card.
3. The transaction is processed following step 2, and a merchant copy is made available by the Tyro iClient API, this merchant copy clearly has the purchase and cashout amounts specified.
4. Depending on the printing preferences set in the POS, the receipt is either printed through the POS printer as an integrated receipt (see Integrated Receipt section for more details) or through the Tyro terminal.
5. The POS sale receipt must display the cashout amount and the purchase amounts separately clearly labelled as such (as shown in the image below).
6. The POS sales invoice must store and register the cashout amount and the purchase amounts separately.

Please note that cashout cannot be applied to refunds.

The POS sales receipt below can be seen clearly displaying the cashout amount separately:

```
        TAX INVOICE

      Merchant 1655650
      Address line 1
      Address line 2

        PERFECT POS

Items                  $
Coffee Cake            $4.00
Croissant              $6.00
Sub-total              $10.00

Tyro Surcharge         $1.00
Tip                    $2.00
Cashout                $50

Total                  $63.00

GST inc                $1


POS ID: 2
Ref: 12345abc
04 Jan 2019 at 02:12 PM
```

Please click here to navigate to the top of the page.

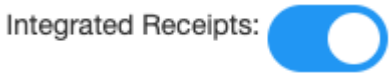
Integrated Receipts

Tyro's Integrated EFTPOS system offers integrated receipts which allow the POS to include the Tyro EFTPOS receipt on the POS sales receipt, and therefore provide the customer with one comprehensive receipt containing both the POS receipt as well as the Tyro EFTPOS receipt. This is an important feature as it provides many benefits to the customer and merchant. Some of the benefits are:

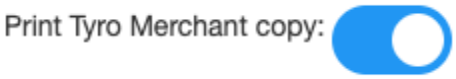
- The customer can be given a single receipt rather than two separate receipts.
- POS systems typically employ fast POS printers which speed up the process of completing the sale.
- Merchants don't have to maintain stock of both terminal printer rolls and POS printer rolls.

Retail/Hospitality POS vendors are therefore **required to implement integrated receipts** for the reasons above. The workflow and deliverables should be as follows:

1. An Integrated Purchase or Refund transaction is processed.
2. There needs to be an option to toggle integrated receipts on or off such as the one seen below:

3. **If Integrated receipts are turned off** the receipts are then printed from the Tyro EFTPOS terminal only, and not the POS printer.
4. **If they are turned on** (as above) on the POS, and the transaction is **not signature-verified,** the POS can optionally print out a merchant copy, and a sale receipt with the Tyro receipt customer copy appended or prepended to it.
5. If integrated receipts are turned on, for customer copies the requirement is that where iClient provides a customer copy in the `transactionCompleteCallback` the POS must make customer copy available to card holder or print it automatically.
6. If integrated receipts are turned on, there needs to be an option to toggle the signature-verification merchant copy printing on or off, such as the one seen below, this is to reduce paper consumption for the merchant:

   Please note that the toggle will turn the merchant copy printing on or off **only** for the merchant copies where signature verification is not required, the signature-verification merchant copy must always be printed. The `signatureRequired` parameter in the

`receiptCallback` is to be used to determine whether the transaction requires signature verification.

1. If the merchant copy printing is turned **"on"** the POS prints out a Tyro merchant copy and the POS sale receipt with the Tyro receipt customer copy appended or prepended to it regardless of the result of the transaction.
2. If the merchant copy printing is turned **"off"** the POS only prints out the POS sale receipt with the Tyro receipt customer copy appended or prepended to it upon completion of the transaction.

7. If merchant copy printing is turned on **"on"** the POS, and the transaction is **signature-verified,** the POS prints out a merchant copy with the signature line appended to it for signature-verification, following the transaction it prints out the Tyro merchant copy and the POS sale receipt with the Tyro receipt customer copy appended or prepended to it regardless of the result of the transaction.
8. If merchant copy printing is turned "**off"** on the POS, and the transaction is **signature-verified,** the POS prints out a merchant copy with the signature line appended to it for signature-verification and then only prints out the POS sales receipt with the Tyro customer copy appended to it, regardless of the transaction outcome.
9. If the transaction is declined, cancelled, or reversed, the iClient will make the customer copy receipt available via the `transactionCompleteCallback` this copy should be obtained and printed for all cancelled, declined, and reversed transactions.
10. Please note that the character count for the DCC transaction Tyro customer receipt is 1200, please ensure that the integrated receipt is able to display the POS sales invoice and the Tyro customer copy on one contiguous printout containing the following items:

    1. Local transaction Amount
    2. Local Currency Symbol
    3. DCC transaction Amount
    4. DCC Currency Symbol
    5. Exchange rate
    6. Commission/Fee
    7. The Words "Transaction Currency"
    8. Proof of the Cardholder's Choice
    9. Statement indicating the Cardholder was offered a choice
    10. Statement indicating who is providing DCC

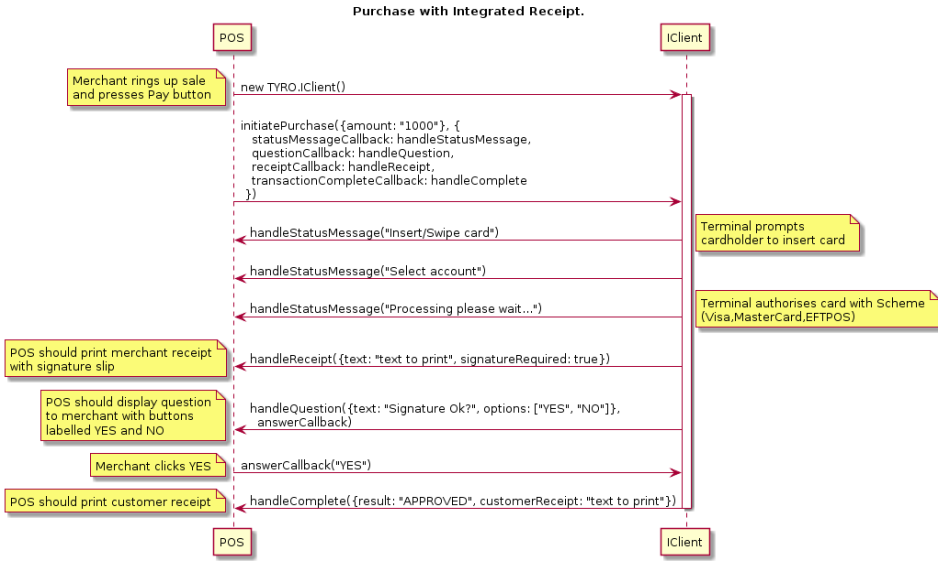
    You must not amend or edit the receipt text provided should be any way. An example receipt is given below:

```
     CUSTOMER COPY            <-- POS adds this line
                              <-- POS adds this line
    Tyro HealthPoint          <--
    125 York Street              |
    Sydney NSW 2000              |
                                 |
  Tyro Payments EFTPOS           |
                                 |
Card: xxxxxxxxxxxx0010(c)        |
```

```
VISA CREDIT                          |
AID: A0000000031010                  |
                                     | Pre-formatted receipt content from
Receipt object
Purchase     AUD     $100.00         |
                  ----------         |
Total        AUD     $100.00         |
                                     |
     Transaction amount              |
         EUR 37.56                   |
       Exchange rate                 |
      1 AUD = 1.1 EUR                |
                                     |
With Dynamic currency                |
conversion, I have been              |
offered by the merchant a            |
choice of currencies for             |
payment, Australian Dollars          |
or my home currency. The             |
payment in my home currency          |
at the point of sale is              |
converted at the Pure                |
Commerce provided wholesale          |
rate including a 5% mark-up.         |
                                     |
This Dynamic Currency                |
Conversion is not associated         |
or endorsed by VISA.                 |
                                     |
Alternatively I can pay in           |
Australian Dollars which is          |
later converted into my home         |
currency at a rate selected          |
by my issuing bank.                  |
                                     |
I understand that my                 |
selected transaction                 |
currency is final.                   |
                                     |
The cardholder expressly             |
agrees to the transaction            |
receipt information by               |
marking the accept box               |
below.                               |
                                     |
            [x]                      |
                                     |
APPROVED                  00         |
                                     |
Terminal ID: 1                       |
Transaction Ref: 170553              |
Authorisation No: 634742             |
01 Jan 2019 at 02:15 PM         <--
```

The sequence diagrams given below should provide further clarification:

Purchase with Integrated Receipt



Purchase with Signature-verified integrated purchase

Sample integrated receipt (non-signature verified transaction):

```
        TAX INVOICE

      Merchant 1655650
      Address line 1
      Address line 2

        PERFECT POS

Items                        $
Coffee                   $4.00
Croissant                $6.00

Sub-total               $10.00

Surcharge                $1.00
Tip                      $2.00

Total                   $13.00

GST inc                  $1.00

POS ID: 2
Ref: 12345abc
04 Jan 2019 at 02:12 PM

------------------------------

     CUSTOMER COPY

      Tyro HealthPoint
      125 York Street
      Sydney NSW 2000

    Tyro Payments EFTPOS

Card: xxxxxxxxxxxx0010(c)
VISA CREDIT
AID: A0000000031010

Purchase     AUD     $10.00
Surcharge    AUD     $1.00
Tip          AUD     $2.00
                  ----------
Total        AUD     $13.00

Terminal ID: 4
Transaction Ref: 547286
Authorisation No: 014458
08 Apr 2020 at 09:55 AM
```

Sample integrated receipt (signature-verified transaction):

```
        MERCHANT COPY

      Merchant 1655650
       Address line 1
       Address line 2

        Tyro EFTPOS

VISA CREDIT
Card: XXXXXXX0010(C)

Purchase     AUD     $100.08
                   ----------
Total        AUD     $100.08

APPROVED W/ SIGNATURE     08

Terminal ID: 4
Transaction Ref: 160086
Authorisation No: F99047
08 Apr 2020 at 09:55 AM


----------------------------
```

The above image shows a merchant copy for signature verified transactions.

```
        TAX INVOICE

      Merchant 1655650
      Address line 1
      Address line 2

        PERFECT POS

Items                      $
Cake                  $60.00
Coffee                $40.08
Sub-total            $100.08


Total                $100.08

GST inc               $1.00

POS ID: 2
Ref: 12345abc
04 Jan 2019 at 02:12 PM

------------------------------


      CUSTOMER COPY

      Merchant 99922
      Address line 1
      Address line 2

       Tyro EFTPOS

EFTPOS
Card: XXXXXXXX5412(s)

Purchase     AUD     $100.08
                   ----------
Total        AUD     $100.08

APPROVED W/ SIGNATURE     08

Terminal ID: 1
Transaction Ref: 101417
Authorisation No: 234433
28 Jan 2024 at 01:50 AM
```

The above image shows an integrated POS receipt for a signature verified transaction, the Customer Copy can be seen appended to the POS sale invoice.

Integrated Surcharging

Integrated Surcharging is a compulsory feature for the retail and hospitality industries, which allows Tyro's dynamic surcharging to be applied to any given purchase transaction and reflected on the POS UI and the POS sales receipt. A prerequisite is that the surcharge rates specified on the Online **Tyro Merchant Portal** and surcharging be enabled on the terminal.

The workflow should ideally be as follows:

1. If the POS vendor is implementing the Tyro Surcharge feature and does not have a surcharge feature of its own, please set the `enableSurcharge` flag in the `initiatePurchase()` request to always be “true”, this will allow the POS to receive the applied surcharge through the Tyro terminal and record it against an approved sale, so that the surcharge-inclusive amount can be refunded if the transaction is to be refunded. In this case, there is no need to implement a toggle button.
2. If the POS vendor isn’t implementing the Tyro surcharge feature and has a surcharge feature of its own too i.e. the ability to apply a surcharge through the POS, then there needs to be a toggle button to ensure that POS surcharging and Tyro surcharging cann**ot** both be on at the same time, an example is as shown below:

   This is to prevent accidental application of a doubled or excessive surcharge amount.

   a. When the toggle is on, please send the enableSurcharge flag as true in the purchase request so that only the Tyro surcharge is applied. Please ensure that when the toggle is on, POS must not apply its surcharge.
   b. When the toggle is off, please send the enableSurcharge flag as false in the purchase request so that only the POS surcharge is applied. Please note that when the enableSurcharge flag is sent as false in the purchase request, the Tyro surcharge will not be applied.
3. The POS initiates a purchase of any amount from the POS UI and includes `enableSurcharge` flag setting it as per point 1, point 2a, or 2b depending on the scenario.
4. The Tyro terminal receives the purchase transaction and prompts the cardholder to tap, swipe, or insert the card.
5. The Tyro terminal dynamically applies surcharge to the transaction. returns data to POS.
6. Please record the additional `surchargeAmount` field from the `transactionCompleteCallback` response for the transaction.
7. Using the surcharge amount returned to the POS display as a line item on the sales invoice and the POS UI.
   a. Make sure to label it as “Tyro Surcharge” on the sales invoice when `enableSurcharge` flag is sent as “true”.
   b. And to label it as “POS Surcharge” on the sales invoice when `enableSurcharge` flag is sent as “false” for more clarity.
8. Surcharge amounts must be stored in the POS in the sales journal with the approved transactions.
9. POS recalls surcharge amount when refunding a transaction and includes it in the full amount to be refunded.

```
        TAX INVOICE

      Merchant 1655650
      Address line 1
      Address line 2

        PERFECT POS

Items                  $
Coffee Cake            $4.00
Croissant              $6.00
Sub-total              $10.00

Tyro Surcharge         $1.00
Tip                    $2.00
Cashout                $50

Total                  $63.00

GST inc                $1


POS ID: 2
Ref: 12345abc
04 Jan 2019 at 02:12 PM
```

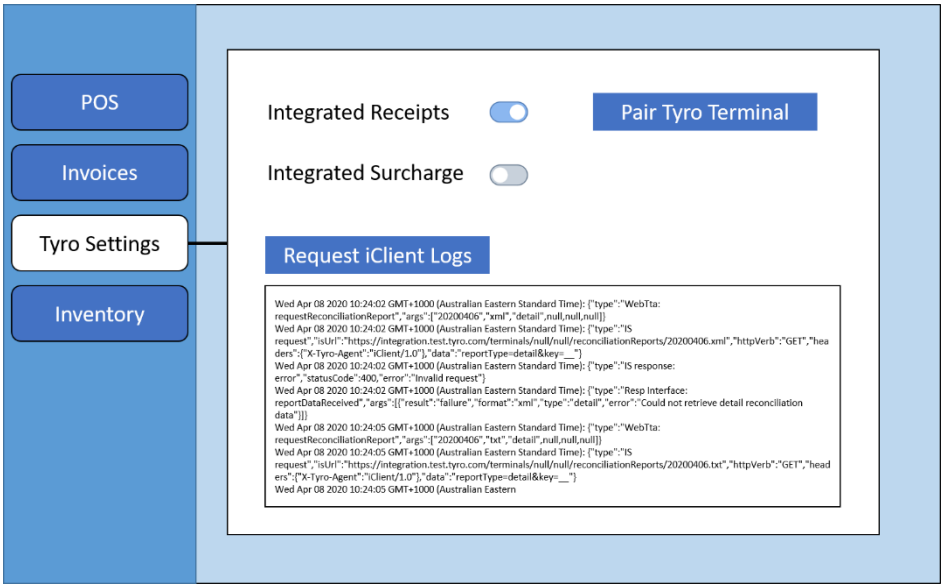
The figures above show the surcharge amount clearly listed on the **POS sales invoice**.

Tyro Settings Page

The Tyro settings page is a compulsory feature for your POS integration, which is designed to provide a simplified, unified interface allowing all Tyro settings to be toggled from a single location within the POS UI. This will not only make it easier for our mutual merchants to change the settings for themselves, but also for our customer support department to provide support for your POS system. The criteria are as follows:

1. The Settings section UI within your POS contains a “Tyro settings“ page.
2. This page contains the following:
    1. Toggles allowing integrated receipt printing and integrated surcharge to be turned on or off.
    2. The Tyro Pairing UI allowing the pairing process to be initiated and handled.
    3. A mechanism e.g., a button and an iFrame to display the Tyro iClient logs on the POS UI. the mechanism can also be a button that has a link to the iClient logs web page.

A sample UI can be seen below:



Integrated Manual Settlement

Integrated Manual Settlement is an optional feature that allows the POS to initiate the closing off of the current business day/settlement period, and the beginning of a new settlement period on the Tyro terminal using a function that is available in the iClient API.
The ideal workflow is as follows:

1. The POS initiates the manual settlement command using the `manualSettlement()` function for the ongoing settlement period from the POS user interface, the POS must have suitable user interface constructs required to initiate the request clearly and accurately labeled to indicate the functionality e.g. “Tyro Manual Settlement”.
2. The POS must display suitable messaging on the interface advising the user of the consequences of the manual settlement request and get the user’s confirmation to initiate the request through a question. an example is given below:

3. The terminal receives the command and the manual settlement workflow is initiated on the terminal.
4. If the request is successful, the POS must use the `result` , `message`, and `currentTerminalBusinessDay` fields from the `responseReceivedCallback` function and display this information for the user’s reference via a POS modal.

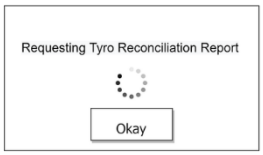
5. If the request is unsuccessful, the POS must use the `result` and `message` fields from the `responseReceivedCallback` function and display this information for the user's reference via a POS modal.

Integrated Reports

The Integrated reports feature essentially, allows the POS to pull the reconciliation reports from the terminal for any of the previous 7 business days. The reports can then be printed by the POS so that the end-of-day functions can all be actioned/automated from within the POS. The report is available either summarized or detailed, in a plain text format suitable for printing as well as a machine readable XML format suitable for line by line reconciliation.

The ideal workflow is:

1. The POS user interface is used to initiate the reconciliation report for any day in the past seven business days using the `reconciliationReport` function, the POS should have the required user-interface constructs to allow the date, the report type (summary or detailed), and type (text or XML - if the POS supports both) to be specified as the `terminalBusinessDay` , `type` , and the `format` request parameters before the request is made to the terminal
2. The iClient status and result modal is not generated for integrated report requests, the POS must therefore display a suitable modal with the correct messaging and/or animation advising the user that the request has been transmitted and is awaiting a response, an example is shown below, other variations/implementations of the below are acceptable:



3. In the case of an error i.e. when the `result` field of the `responseReceivedCallback` returns 'failure', the POS must display the error message returned in the `error` field of the `responseReceivedCallback` on a pop-up modal, an example is shown below, other variations/implementations of the below are acceptable:

4. If a text report has been requested, The POS must parse out and remove the labels included on the pre-formatted report (as shown below) before presenting the report on the screen or making it available for printing.

```
5. MEDIUM_CENTRED: Merchant Name Here
6. MEDIUM_CENTRED: 125 York Street
7. MEDIUM_CENTRED: Sydney NSW 2000
8. NEW_LINE
9. LARGE_CENTRED: DETAIL REPORT
10.  NEW_LINE
11.  MEDIUM_CENTRED: Tyro EFTPOS
12.  NEW_LINE
13.  MEDIUM: Merchant ID: 200
14.  MEDIUM: Terminal ID: 200
15.  NEW_LINE
16.  MEDIUM: Printed: 07/12/21 19:04
17.  NEW_LINE
18.  MEDIUM: Card type: All cards
19.  NEW_LINE
20.  MEDIUM_BOLD: Date: 07/12/21
21.  NEW_LINE
22.  MEDIUM: 08:39       Amex 6082   (s)
23.  MEDIUM: Purchase             $19.90
24.  NEW_LINE
25.  MEDIUM: 08:58     EFTPOS 9815   (s)
26.  MEDIUM: Purchase             $39.95
27.  NEW_LINE
28.  MEDIUM: 09:27     EFTPOS 0836   (c)
29.  MEDIUM: Purchase            $106.80
30.  NEW_LINE
31.  MEDIUM: 09:32 MasterCard 2785   (c)
32.  MEDIUM: Purchase             $24.95
33.  NEW_LINE
34.  MEDIUM: 09:48     EFTPOS 7926   (c)
35.  MEDIUM: Purchase              $9.90
36.  NEW_LINE
37.  MEDIUM: 11:36 MasterCard 6818   (c)
38.  MEDIUM: Purchase            $110.10
39.  NEW_LINE
40.  MEDIUM: 11:47       Amex 1003   (s)
41.  MEDIUM: Refund              -$39.95
42.  NEW_LINE
43.  MEDIUM: 12:04     EFTPOS 1759   (c)
44.  MEDIUM: Purchase             $44.00
45.  NEW_LINE
46.  MEDIUM: 12:18     EFTPOS 5161   (c)
47.  MEDIUM: Purchase             $43.40
48.  NEW_LINE
49.  MEDIUM: 12:21 MasterCard 7826   (c)
50.  MEDIUM: Purchase             $69.95
51.  NEW_LINE
52.  MEDIUM: 12:58 MasterCard 9977   (c)
53.  MEDIUM: Purchase             $54.90
54.  NEW_LINE
55.  MEDIUM: 13:06     EFTPOS 1090   (c)
56.  MEDIUM: Purchase              $6.95
57.  NEW_LINE
```

58. MEDIUM: 13:12 Visa 3527 (c)
59. MEDIUM: Purchase $27.00
60. NEW_LINE
61. MEDIUM: 13:14 EFTPOS 4795 (s)
62. MEDIUM: Purchase $25.90
63. NEW_LINE
64. MEDIUM: 13:22 EFTPOS 4709 (s)
65. MEDIUM: Purchase $10.08
66. NEW_LINE
67. MEDIUM: 13:33 Amex 4264 (s)
68. MEDIUM: Purchase $69.95
69. NEW_LINE
70. MEDIUM: 13:45 Visa 0930 (s)
71. MEDIUM: Purchase $29.95
72. NEW_LINE
73. MEDIUM: 14:04 Amex 8661 (s)
74. MEDIUM: Purchase $11.95
75. NEW_LINE
76. MEDIUM: 14:15 Visa 2066 (c)
77. MEDIUM: Purchase $32.30
78. NEW_LINE
79. MEDIUM: 14:16 Amex 3005 (s)
80. MEDIUM: Purchase $53.60
81. NEW_LINE
82. MEDIUM: 14:49 EFTPOS 4739 (s)
83. MEDIUM: Purchase $21.95
84. NEW_LINE
85. MEDIUM: 15:06 EFTPOS 4709 (c)
86. MEDIUM: Purchase $8.95
87. NEW_LINE
88. MEDIUM: 15:08 Visa 2369 (c)
89. MEDIUM: Purchase $46.85
90. NEW_LINE
91. MEDIUM: 15:30 EFTPOS 9475 (s)
92. MEDIUM: Purchase $19.75
93. NEW_LINE
94. MEDIUM: 15:36 Amex 0043 (s)
95. MEDIUM: Purchase $59.95
96. NEW_LINE
97. MEDIUM: 16:02 EFTPOS 2923 (s)
98. MEDIUM: Purchase $24.95
99. NEW_LINE
100. MEDIUM: 16:40 EFTPOS 2410 (c)
101. MEDIUM: Purchase $21.95
102. NEW_LINE
103. MEDIUM: 16:44 Visa 3895 (c)
104. MEDIUM: Purchase $58.60
105. NEW_LINE
106. MEDIUM: 16:49 EFTPOS 3750 (s)
107. MEDIUM: Purchase $12.95
108. NEW_LINE
109. MEDIUM: 16:50 EFTPOS 3750 (s)
110. MEDIUM: Purchase $22.90
111. NEW_LINE
112. MEDIUM: 17:34 EFTPOS 8617 (c)
113. MEDIUM: Purchase $49.55
114. NEW_LINE

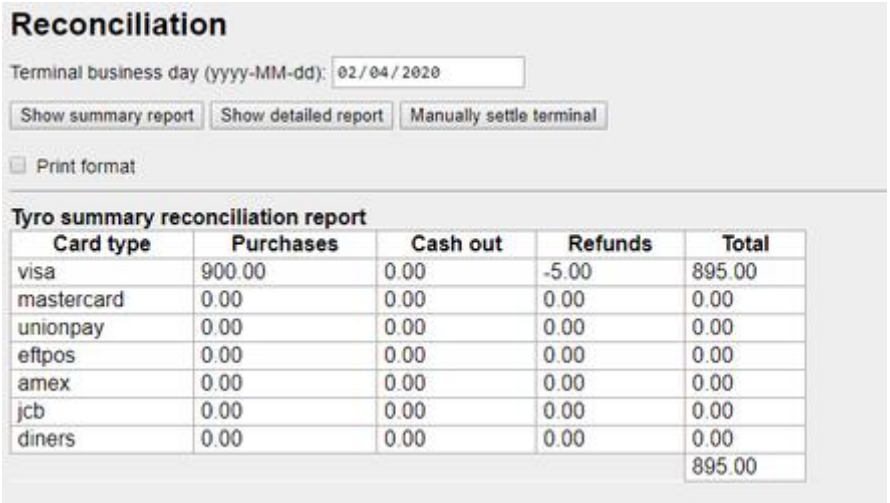
```
115. SMALL: s: swiped, c: chip, m: manual
116. SMALL: e: express payment
117. NEW_LINE
118. LARGE_CENTRED: SUMMARY REPORT
119. MEDIUM: Purchase    30    $1,139.88
120. MEDIUM: Cash out (0)          $0.00
121. MEDIUM: Tip (0)               $0.00
122. MEDIUM: Refund       1      -$39.95
123. MEDIUM: Void         0        $0.00
124. MEDIUM_RIGHT: ------------
125. MEDIUM: NET TOTAL   31    $1,099.93
126. NEW_LINE
127. MEDIUM: EFTPOS      16      $469.93
128. MEDIUM: Visa         5      $194.70
129. MEDIUM: MasterCard   4      $259.90
130. MEDIUM_RIGHT: ------------
131. MEDIUM: TYRO TOTAL  25      $924.53
132. NEW_LINE
133. MEDIUM: AMEX         6      $175.40
134. MEDIUM: JCB          0        $0.00
135. MEDIUM: Diners       0        $0.00
136. MEDIUM_RIGHT: ------------
137. MEDIUM: NET TOTAL   31    $1,099.93
138. NEW_LINE
139. FORM_FEED
140.
```

141. If the POS requests an XML formatted report, then the POS must parse the XML to obtain the data and present the data to the user in a presentable manner, for example as shown in Point 6 below.

```
142. <?xml version="1.0" encoding="UTF-8"?>
143. <reconciliation-detail mid="200" tid="200" terminal-business-
   day="2021-12-07" total="1620.00">
144. <transaction type="purchase" card-type="visa" amount="100.00"
   tip="10.00" transaction-local-date-time="2021-12-07 12:30:30" tyro-
   reference="870020" merchant-reference="07106104823130635121967 7"
   settlement-date="2021-12-09"/>
145. <transaction type="purchase" card-type="mastercard" amount="200.00"
   transaction-local-date-time="2021-12-07 12:31:20" tyro-
   reference="885214" merchant-reference="07106104823130635121967 8"
   settlement-date="2021-12-09"/>
146. <transaction type="purchase" card-type="jcb" amount="40.00"
   transaction-local-date-time="2021-12-07 12:33:23" tyro-
   reference="896534" merchant-reference="07106104823130635121967 9"/>
147. <transaction type="purchase" card-type="amex" amount="30.00"
   transaction-local-date-time="2021-12-07 12:36:35" tyro-
   reference="905845" merchant-reference="07106104823130635121968 0"/>
148. <transaction type="purchase" card-type="eftpos" amount="650.00" cash-
   out="50.00" transaction-local-date-time="2021-12-07 12:40:30" tyro-
   reference="912556" merchant-reference="07106104823130635121968 1"
   settlement-date="2021-12-09"/>
149. <transaction type="purchase" card-type="eftpos" amount="450.00"
   transaction-local-date-time="2021-12-07 12:50:30" tyro-
   reference="920187" merchant-reference="07106104823130635121968 2"
   settlement-date="2021-12-09"/>
```

```
<transaction type="purchase" card-type="diners" amount="70.00"
   transaction-local-date-time="2021-12-07 13:30:30" tyro-
   reference="935587" merchant-reference="07106104823130635121968 3"/>
<transaction type="void" card-type="mastercard" amount="-80.00"
   transaction-local-date-time="2021-12-07 13:50:30" tyro-
   reference="946585" merchant-reference="07106104823130635121968 4"
   settlement-date="2021-12-09"/>
<transaction type="purchase" card-type="visa" amount="170.00"
   transaction-local-date-time="2021-12-07 14:23:25" tyro-
   reference="953594" merchant-reference="07106104823130635121968 5"
   settlement-date="2021-12-09"/>
<transaction type="refund" card-type="visa" amount="-70.00"
   transaction-local-date-time="2021-12-07 15:41:12" tyro-
   reference="962548" merchant-reference="07106104823130635121968 5"
   settlement-date="2021-12-09"/>
</reconciliation-detail>
```

155. The terminal pulls the reports off the integration server and the POS uses the response delivered by the terminal to either print the receipt returned by the terminal through the POS printer or display it on the screen.



Card type	Purchases	Cash out	Refunds	Total
visa	900.00	0.00	-5.00	895.00
mastercard	0.00	0.00	0.00	0.00
unionpay	0.00	0.00	0.00	0.00
eftpos	0.00	0.00	0.00	0.00
amex	0.00	0.00	0.00	0.00
jcb	0.00	0.00	0.00	0.00
diners	0.00	0.00	0.00	0.00
				895.00

The above image shows a sample interface that may be used to retrieve and display reconciliation reports on the POS interface, there are two kinds of reconciliation reports i.e. summary and detailed. The snippets below show a sample summary and detailed reconciliation reports:

Summary Reconciliation Report

```
Merchant Name Here
 125 York Street
 Sydney NSW 2000

  SUMMARY REPORT

   Tyro EFTPOS

Merchant ID: 200
```

Terminal ID: 200

Printed: 08/04/20 10:34

Card type: All cards

Date: 08/04/20

Purchase	30	$1,139.88
Cash out (0)		$0.00
Tip (0)		$0.00
Refund	1	-$39.95
Void	0	$0.00
NET TOTAL	31	$1,099.93

EFTPOS	16	$469.93
Visa	5	$194.70
MasterCard	4	$259.90
TYRO TOTAL	25	$924.53

AMEX	6	$175.40
JCB	0	$0.00
Diners	0	$0.00
NET TOTAL	31	$1,099.93

Detailed Reconciliation Report

Merchant Name Here
125 York Street
Sydney NSW 2000

DETAIL REPORT

Tyro EFTPOS

Merchant ID: 200
Terminal ID: 200

Printed: 08/04/20 10:26

Card type: All cards

Date: 08/04/20

08:39 Amex 6082 (s)
Purchase $19.90

08:58 EFTPOS 9815 (s)
Purchase $39.95

09:27 EFTPOS 0836 (c)
Purchase $106.80

09:32 MasterCard 2785 (c)
Purchase $24.95

09:48 EFTPOS 7926 (c)
Purchase $9.90

11:36 MasterCard 6818 (c)
Purchase $110.10

11:47 Amex 1003 (s)
Refund -$39.95

12:04 EFTPOS 1759 (c)
Purchase $44.00

12:18 EFTPOS 5161 (c)
Purchase $43.40

12:21 MasterCard 7826 (c)
Purchase $69.95

12:58 MasterCard 9977 (c)
Purchase $54.90

13:06 EFTPOS 1090 (c)
Purchase $6.95

13:12 Visa 3527 (c)
Purchase $27.00

13:14 EFTPOS 4795 (s)
Purchase $25.90

13:22 EFTPOS 4709 (s)
Purchase $10.08

13:33 Amex 4264 (s)
Purchase $69.95

13:45 Visa 0930 (s)
Purchase $29.95

14:04 Amex 8661 (s)
Purchase $11.95

14:15 Visa 2066 (c)
Purchase $32.30

14:16 Amex 3005 (s)
Purchase $53.60

14:49 EFTPOS 4739 (s)
Purchase $21.95

15:06 EFTPOS 4709 (c)
Purchase $8.95

```
 15:08       Visa 2369   (c)
 Purchase             $46.85

 15:30     EFTPOS 9475   (s)
 Purchase             $19.75

 15:36       Amex 0043   (s)
 Purchase             $59.95

 16:02     EFTPOS 2923   (s)
 Purchase             $24.95

 16:40     EFTPOS 2410   (c)
 Purchase             $21.95

 16:44       Visa 3895   (c)
 Purchase             $58.60

 16:49     EFTPOS 3750   (s)
 Purchase             $12.95

 16:50     EFTPOS 3750   (s)
 Purchase             $22.90

 17:34     EFTPOS 8617   (c)
 Purchase             $49.55

s: swiped, c: chip, m: manual
e: express payment

      SUMMARY REPORT
 Purchase    30    $1,139.88
 Cash out (0)          $0.00
 Tip (0)               $0.00
 Refund       1      -$39.95
 Void         0        $0.00
       ------------
 NET TOTAL   31    $1,099.93

 EFTPOS      16      $469.93
 Visa         5      $194.70
 MasterCard   4      $259.90
       ------------
 TYRO TOTAL  25      $924.53

 AMEX         6      $175.40
 JCB          0        $0.00
 Diners       0        $0.00
       ------------
 NET TOTAL   31    $1,099.93
```

Integrated Pre-authorisation

Integrated Pre-authorisation is an optional feature provided by the iClient API that allows the POS to initiate, increment, complete, or void an integrated pre-authorisation transaction. A pre-authorisation transaction is where an amount is debited from a debit card and “held“ on the pre-authorisation, this transaction can be incremented, completed, voided later through the POS using the functions provided in the iClient API.

The criteria for the development of the pre-authorisation feature is as given below:

1. The POS must have the required user interface constructs to allow the user to specify the pre-authorisation amount and initiate the request.
2. The POS initiates a pre-authorisation transaction for the given amount using the `initiateOpenPreAuth()` function.
3. Upon a successful pre-authorisation request, the POS must:
    1. Use the `result` and `preAuthCompletionReference` from the `transactionCompleteCallback` and display the result, the amount, and the pre-authorisation completion reference for the user’s reference.
    2. Mark the pre-authorisation in the appropriate status indicating that it is open and due for completion.
    3. Print the integrated receipt (if integrated receipts have been developed) with the Tyro Customer Copy which is returned in the `customerReceipt` field of the `transactionCompleteCallback` appended to the bottom as shown below.
4. Upon an unsuccessful open pre-authorisation request, the POS must use the `result` field from the `transctionCompleteCallback` and display this on the interface for the user’s reference and not indicate the pre-authorisation as having been successfully opened.
5. The POS must also store the pre-authorisation completion reference and the amount against the open pre-authorisation, to ensure that the user does not have to manually enter the completion reference in to increment, close, or void the pre-authorisation.
6. The POS must also store the current pre-authorisation amount against the open pre-auth, so that the POS can conduct the required validation to ensure that the maximum pre-authorisation amount is not exceeded when the pre-authorisation is closed.
7. The POS must clearly display the current maximum pre-authorisation completion amount for the user’s reference.
8. The POS must allow the user to increment, close, and void the pre-auth using the appropriate user interface constructs, for incrementing and closing the pre-authorisation the POS must allow the user to specify the increment amount and completion amount respectively.
9. The POS must use the `initiateIncrementPreAuth()` function to initiate requests to increment the pre-authorisation, the pre-auth increment amount the user has specified and the original pre-auth completion reference must be used as the `amount` and `completionReference` parameter in the request.
10. Upon successful increment pre-authorisation request i.e. the `result` field of the `responseReceivedCallback` including **‘APPROVED’**:

    1. The POS must display the `result` field as well along with any other messaging to signify that the pre-auth has been successfully incremented by the amount specified.

    2. The POS must then adjust the pre-authorisation amount to indicate the addition of the pre-auth increment amount.
11. Upon an unsuccessful increment pre-authorisation request i.e. the `result` field of the `responseReceivedCallback` not including “APPROVED”:

    1. The POS must display the `result` field and any other messaging to signify that the increment pre-auth request was unsuccessful.
12. The POS must use the `closePreAuth()` function to initiate the close pre-auth request, the stored and maintained pre-auth completion amount and the original pre-auth completion reference must be used as the `amount` and `completionReference` parameter in the request.
13. Upon successful close pre-authorisation request i.e. the `result` field of the `responseReceivedCallback` including **‘success’**:

    1. The POS must display the `result` field and the message that is returned in the `message` field as well along with any other messaging to signify that the pre-auth has been successfully closed.
    2. The POS must then mark the pre-authorisation off in the appropriate status to indicate that it has been closed e.g. “Closed”
14. Upon an unsuccessful close pre-authorisation request i.e. the `result` field of the `responseReceivedCallback` including **‘failure’**:

    1. The POS must display the `result` field and the message that is returned in the `message` field as well along with any other messaging to signify that the close pre-auth request was unsuccessful.
    2. The POS must not mark the pre-auth as closed.
15. The POS must use the `voidPreAuth()` function to initiate the void pre-authorisation request, the stored, and maintained pre-auth completion reference must be used as the `completionReference` parameter in the request.
16. Upon successful void pre-auth request i.e. the `result` field of the `responseReceivedCallback` including **‘success’**:

    1. The POS must display the `result` field and the message that is returned in the `message` field as well along with any other messaging to signify that the pre-auth has been successfully voided.
    2. The POS must then mark the pre-authorisation off in the appropriate status to indicate that it has been voided e.g. “Voided”
17. Upon an unsuccessful void pre-auth request i.e. the `result` field of the `responseReceivedCallback` including **‘failure’**:

    1. The POS must display the `result` field and the message that is returned in the `message` field as well along with any other messaging to signify that the void pre-auth request was unsuccessful.
    2. The POS must not mark the pre-auth as void.

The integrated receipt as referred to by point 3.c

```
     Merchant 1655650
      Address line 1
      Address line 2

        PERFECT POS

PreAuth Total             $102.00
Completion Reference      306543



POS ID: 2
Ref: 12345abc
10 Nov 2021 at 05:18 PM


------------------------------


        CUSTOMER COPY

       Merchant 10001
       Address line 1
       Address line 2

        Tyro EFTPOS

Visa
Card: XXXXXXXX8026(s)

PreAuth      AUD     $102.00

APPROVED                  00

A hold has been placed on
your account in the amount
of $102.00. When you check
out, up to $102.00 will be
debited from your account.

Completion Ref: 238290

Terminal ID: 10004
Transaction Ref: 172207
Authorisation No: P39351
17 Nov 2021 at 06:33 PM
```

Integrated Split payments.

Integrated split payment is a feature optional for the retail and mandatory for the hospitality industry that allows a **purchase or refund** transaction to be paid in split payments of cash and Eftpos. These payments can be a combination of cash and/or eftpos, i.e.

- Cash + EFTPOS + EFTPOS
- EFTPOS + EFTPOS
- EFTPOS + EFTPOS+EFTPOS,

and can be made in **any order**.

The criteria is as follows:

1. The POS UI provides a mechanism to allow the transaction to be completed in multiple split payments of cash and EFTPOS allowing the amount and the payment type to be specified for each split payment e.g. $ 50 cash.
2. If the Tyro EFTPOS option is selected, the portion of the split payment is sent through to the terminal wherein the card is presented via tap, swipe, or insert and the payment is processed.
3. The POS reflects the payment as having gone through, and the amount still outstanding on the UI.
4. The remaining amount is also paid off in EFTPOS and/or cash payments as specified in steps 2 - 4, at the completion of the transaction the POS prints an integrated receipt with the Tyro merchant copies for each individual payments appended to the sale receipt.
5. Please note that once a payment has been accepted, and there is an outstanding tendered amount remaining, the transaction **can not be voided or deleted without refunding the accepted payments.**
6. Please note that for refunds, split payments must only be allowed for purchase transactions that were processed via split payments.

The image below shows an integrated receipt for a split payment transaction:

```
        TAX INVOICE

      Merchant 1655650
      Address line 1
      Address line 2

        PERFECT POS

Items                   $
Coffee Cake             $60.00
Icecream Cake           $60.00
Sub-total               $120.00


Total                   $120.00



POS ID: 2
Ref: 12345abc
04 Jan 2019 at 02:12 PM

------------------------------
```

```
     CUSTOMER COPY

      Tyro HealthPoint
      125 York Street
      Sydney NSW 2000

    Tyro Payments EFTPOS

Card: xxxxxxxxxxxx0010(c)
VISA CREDIT
AID: A0000000031010

Purchase     AUD     $60.00
                  ----------
Total        AUD     $60.00



Terminal ID: 4
Transaction Ref: 547286
Authorisation No: 014458
08 Apr 2020 at 09:55 AM



     CUSTOMER COPY

      Tyro HealthPoint
      125 York Street
      Sydney NSW 2000

    Tyro Payments EFTPOS

Card: xxxxxxxxxxxx0010(c)
Mastercard CREDIT
AID: A0000000025010

Purchase     AUD     $60.00
                  ----------
Total        AUD     $60.00



Terminal ID: 4
Transaction Ref: 547287
Authorisation No: 014459
08 Apr 2020 at 09:55 AM
```

Integrated Tipping

Integrated tipping is an optional feature for the retail industry and mandatory for hospitality POS systems, this feature allows tips applied to any given purchase transaction to be registered and reflected end-to-end from the Tyro reporting to the POS UI, and POS cashdrawer.

The workflow should ideally be as follows:

1. The POS initiates a purchase of any amount from the POS UI.
2. The Tyro terminal receives the purchase transaction and confirms whether the cardholder would like to leave a tip, provided '**Yes'** is selected on the Tyro terminal, the tip amount is entered in, and the cardholder is prompted to to tap, swipe, or insert the card.
3. The Tyro terminal applies the tip amount to the transaction.
4. Given that the transaction is approved, the terminal returns the transaction data back to the POS which can be used to register the sale amount and the tip amount separately on the POS sale invoice and the POS UI.
5. The POS must use the `tipAmount` field from the `transactionCompleteCallback` to retrieve the tip applied and display it as a line item on the POS sales invoice accurately labeled as 'Tip' (as shown below).
   The tip amount can be clearly seen labelled as such on the POS invoice below.

```
        TAX INVOICE

      Merchant 1655650
      Address line 1
      Address line 2

        PERFECT POS

Items                        $
Coffee                    $4.00
Croissant                 $6.00

Sub-total                $10.00

Surcharge                 $1.00
Tip                       $2.00

Total                    $13.00

GST inc                   $1.00

Cashout:                 $50.00

POS ID: 2
Ref: 12345abc
04 Jan 2019 at 02:12 PM
```

6. Tip amounts must be stored in the POS in the sales journal with the approved transactions so that the POS recalls the tip amount when refunding a transaction, and includes it in the full amount to be refunded.

Integrated Bar-tabs

The Integrated bar-tabs feature is a value-added optional feature available for hospitality businesses that is aimed at allowing the customer to initiate a flexible pre-authorisation style, amount-limited transaction which does not require ID documents or credit/debit cards to be kept behind the counter as security, thereby allowing both the customer and the merchant greater ease, flexible, security, and convenience.
The criteria for the feature is as follows:

1. The POS must have the required user interface constructs to allow the user to specify the maximum bar-tab amount and initiate the bar-tab request.
2. An open bar-tab request is initiated for the specified maximum amount through the POS UI using the `initiateOpenTab()` function.
3. Upon successful open tab request, the POS must:
    1. Use the `result` and `tabCompletionReference` from the `transactionCompleteCallback` and display the result, the maximum amount, and the tab completion reference for the user's reference.
    2. Mark the tab in the appropriate status indicating that the tab is open e.g. "Open Tab"
    3. Print the integrated receipt (if integrated receipts have been developed) with the Tyro Customer Copy which is returned in the `customerReceipt` field of the `transactionCompleteCallback` appended to the bottom as shown below.
4. Upon an unsuccessful open tab request, the POS must use the `result` field from the `transctionCompleteCallback` and display this on the interface for the user's reference and not indicate the tab as having been successfully opened.
5. The POS must also store the tab completion reference and the maximum amount against the open tab, to ensure that the user does not have to manually enter the completion reference in to close or void the bar-tab.
6. The POS must also store maximum amount against the open tab, so that the POS can conduct the required validation to ensure that the maximum tab amount is not exceeded as items are added to the bar-tab.
7. The POS must also keep a track of the bar-tab completion amount (the amount out of the maximum amount that has been availed) as items are added/removed to/from the bar-tab, and not allow any more items to be added to the bar-tab once the maximum tab amount has been equaled by the completion amount, or if adding the item will result in the amount exceeding the maximum bar-tab amount.
8. The POS must clearly display the current bar-tab completion amount for the user's reference and allow the user to close and void the bar-tab using the appropriate user interface constructs.
9. The POS must use the `closeTab()` function to initiate the close tab request, the stored and maintained bar-tab completion amount and the tab completion reference must be used as the `amount` and `completionReference` parameter in the request.
10. Upon successful close tab request i.e. the `result` field of the `responseReceivedCallback` including **'success'**:

    1. The POS must display the `result` field and the message that is returned in the `message` field as well along with any other messaging to signify that the bar-tab has been successfully closed.
    2. The POS must then mark the tab off in the appropriate status to indicate that it has been closed e.g. "Closed"
11. Upon an unsuccessful close tab request i.e. the `result` field of the `responseReceivedCallback` including **'failure'**:

    1. The POS must display the `result` field and the message that is returned in the `message` field as well along with any other messaging to signify that the close bar-tab request was unsuccessful.
    2. The POS must not mark the bar-tab as closed.
12. The POS must use the `voidTab()` function to initiate the void tab request, the stored and maintained bar-tab completion reference must be used as the `completionReference` parameter in the request.
13. Upon successful void tab request i.e. the `result` field of the `responseReceivedCallback` including **'success'**:

    1. The POS must display the `result` field and the message that is returned in the `message` field as well along with any other messaging to signify that the bar-tab has been successfully voided.
    2. The POS must then mark the tab off in the appropriate status to indicate that it has been voided e.g. "Voided"
14. Upon an unsuccessful void tab request i.e. the `result` field of the `responseReceivedCallback` including **'failure'**:

    1. The POS must display the `result` field and the message that is returned in the `message` field as well along with any other messaging to signify that the void bar-tab request was unsuccessful.
    2. The POS must not mark the bar-tab as void.

The integrated receipt as referred to by point 3.c

```
     Merchant 1655650
      Address line 1
      Address line 2

        PERFECT POS

Tab Total                     $102.00
Tab Completion Reference      306543



POS ID: 2
Ref: 12345abc
10 Nov 2021 at 05:18 PM


------------------------------


       CUSTOMER COPY

        Merchant 795
       Address line 1
       Address line 2

        Tyro EFTPOS
```

```
Visa
Card: XXXXXXXXXXXXXX1199(s)

Tab          AUD     $102.00

APPROVED                  00

A hold has been placed on
your account in the amount
of $102.00. When the tab is
closed, up to $102.00 will
be debited from your
account.

Completion Ref: 306543

Terminal ID: 12
Transaction Ref: 107561
Authorisation No: G50476
10 Nov 2021 at 05:18 PM
```

Headless Pairing

Only required if local storage is blocked in your POS. Pairing is normally done in Tyro hosted pairing page.

Integration with Tyro involves the Tyro terminal pairing process being handled via a pairing UI. There are two methods available for this process to be implemented:

1. Headless pairing
2. Headful pairing

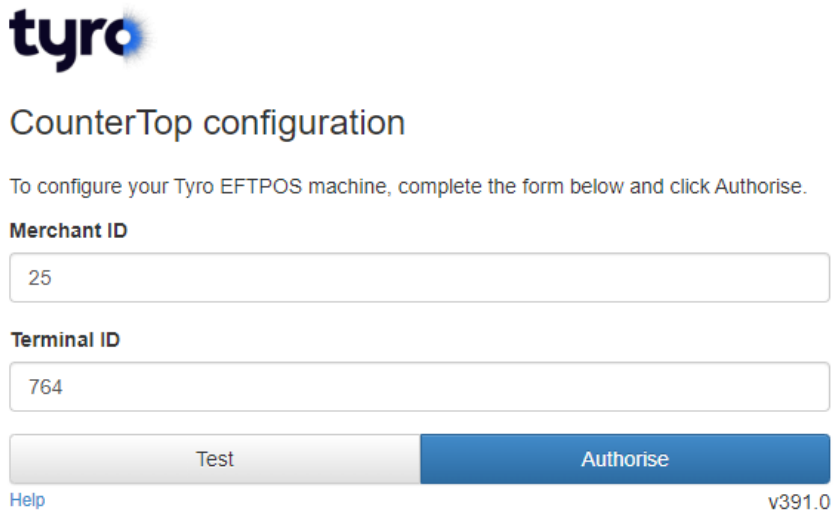
The main difference between the headless and headful pairing is that instead of the Tyro UI, the POS provider has to design a customised UI that allows the merchant ID, and terminal ID to be input, the pairing to be initiated, and to handle the result of the process including any and all error-handling.

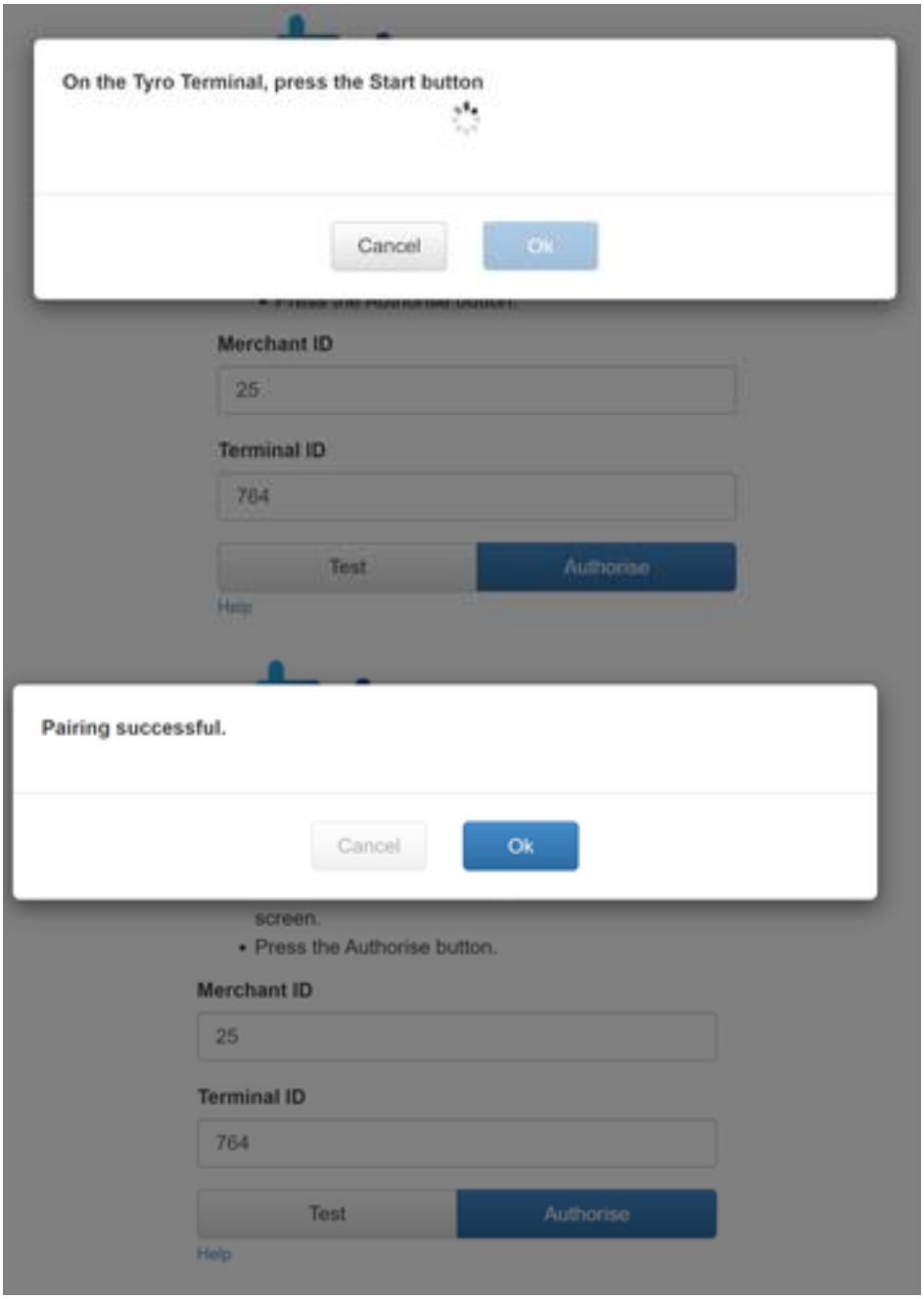
“Headless” pairing is the terminal pairing process that uses a **custom-made POS UI** instead of the ready-made Tyro UI that is available for iClient. This feature should **only** be used if your browser/integration does not have or can not use local storage
The ideal workflow is as follows:

1. The Custom UI pops up allowing the Merchant ID and terminal ID to be entered.
2. The Custom UI must have a mechanism built into it that allows the pairing to be initiated e.g. a “Pair“ button.

3. Once the merchant and terminal ID are entered and the pairing process is initiated by pressing the pair button the POS UI displays prompts the user to perform the Authorize POS function on the terminal by pressing the **“Start“** button from within the Integrated EFTPOS section in the Configuration menu on the terminal.
4. The pairing UI should use the `response.status` parameter to keep a track of the pairing process and display message/s to the user on the UI accordingly (addressed in point 5), this parameter returns statuses **‘Success', ‘Failure’, or ‘inProgress’.**
5. Please note that the status messages e.g., **Pairing successful** and **“Please perform the Authorize POS function on the terminal”** pertaining to the pairing process be obtained from the `responseReceivedCallback` in the `response.message` object.
6. The Start button is pressed on the terminal, and provided that the Integration key is received the POS displays a message advising that the pairing has been successful.
7. Provided there is an error, the POS UI must handle that error gracefully and display the Error message/related diagnostic messaging as is returned by the iClient API.
8. Once the pairing has been initiated by pressing the 'Pair' button, The POS must be allowed to run the pairing process for 90 seconds or until resolution via a **Success** or **Failure** status message being delivered by `response.status`, during this period the user must not be allowed to navigate away from the pairing UI e.g. by closing the pairing UI through a "X" close button. This is because navigating away while a pairing is in progress can cause might cause the user to try the request again even though a pairing request is already pending.

The images below indicate the layout, messaging, and status updates delivered by the Tyro headful pairing UI, the headless UI is to follow the same layout.





Please click here to navigate to the top of the page.

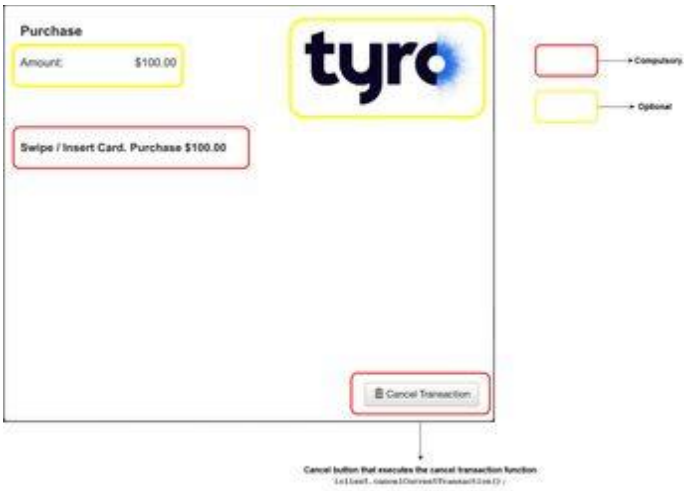
Headless transaction User Interface (UI)

The headless transaction UI must **only** be used if your integration is not able to display an iframe or a modal.

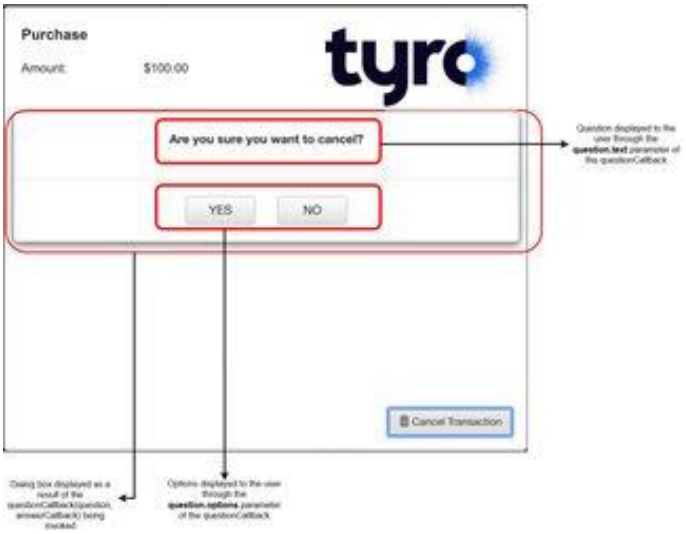
The headless transaction user interface is a value-added optional feature, that allows the POS developer to incorporate the unique look and feel of their POS branding into the transaction UI that is presented to the POS operator **once a transaction has been initiated** from the POS.
This UI must fully mirror the operation of the Tyro ready-built transaction UI, and must have the same error-handling and diagnostic message reporting, status-message reporting, and option-answer handling mechanisms as the Tyro UI. The headless transaction UI must **only** be used if your integration is not able to display an iFrame or a modal.
The ideal workflow is as follows:

1. A transaction (purchase, refund, pre-authorization) is initiated from the POS.
2. The POS transaction UI is presented to the POS operator, presenting the following:
   a. The status messages for that transaction (from the `statusCallback`) e.g. **Processing transaction - please wait, Enter PIN, Select account, Swipe card. Purchase: $100.00, Purchase started - Amount: $100.00, Cashout: $0.00**
   b. A button that allows the transaction to be cancelled from the POS UI using the `cancelCurrentTransaction()` function.
3. The card is tapped, inserted, or swiped through the terminal, and the transaction is processed, the POS UI keeps displaying the status messages being reported by iClient as the transaction proceeds.
4. The POS UI must not have a “Close X” button allowing the modal to be closed down while a transaction is being processed on the terminal, Clicking on the screen outside the modal must also not cause the POS UI to be closed down or dismissed with the transaction still being processed.
5. Any questions or options (buttons) as required are displayed (using the `questionCallback`) by the POS UI to the customer e.g. **Approved with Signature - Signature OK? 1-Yes 2-No** or **Approved - Print Customer Copy? 1-Yes 2-No**
   During signature transactions make sure your POS can handle multiple questions in the same transaction.
6. Error-handling and diagnostic reporting is conducted by the custom UI in the background and if any error is encountered e.g HTTP error messages 202, 400, 403, 404, 409, 410, 500, 501, 502, 503, and 504, the relevant error message that is provided to the POS through the `transactionCompleteCallback` is displayed and the subsequent workflow handled accordingly.
7. The transaction is processed until completion, when the POS is required to display the end-result of the transaction that is returned to the POS through the `transactionCompleteCallback`.
   Please ensure that the POS UI is only cleared 5 or 7 seconds after the `transactionCompleteCallback` delivers the result or outcome of the transaction through the `transactionData.result` object, this is to ensure graceful operation and better readability for the POS user.

If you are developing your own UI through the headless UI method, please ensure that there is a mechanism to cancel the transaction in line with the following images taken from the Tyro UI:
The UI developed as part of your integration should follow the options and provide the features specified (in addition to the ones detailed above) in the figure below:



Clicking the **'Cancel Transaction'** Button will trigger the question callback. This will result in the options being presented to the user as shown in the figure below:



POS Information

POS information is a mandatory feature of Tyro integration, it entails the specification of correct, accurate, and up-to-date information pertaining to the POS product in the relevant transaction request fields. These transaction requests are recorded in both the iClient logs as well as the server-side logs, this allows Tyro to use these logs to generate accurate reporting which can be used to provide better help and support to our partners and merchants, as well as to assist Tyro in making informed decisions based on accurate information.

The criteria pertaining to POS information is as given below:

1. The POS information fields i.e. `posProductVendor` (String), `posProductName`(String), and `posProductVersion` (String) are specified in the `posProductData` (Object) parameter of the Tyro iClient constructor which is defined as `TYRO.iClient(apiKey, posProductData)`.
   The POS must declare these fields with the correct, valid, meaningful, and accurate information pertaining to the POS product when the constructor is created, as an example:
   ```
   var iclient = new TYRO.IClient("apiKey", {posProductVendor: "Tyro Soft", posProductName: "Tyro Soft Smart Retail POS", posProductVersion: "v1.00"})
   ```
2. These fields must not contain spaces, or null values, for example:
   ```
   var iclient = new TYRO.IClient("apiKey", {posProductVendor: "Tyro Soft", posProductName: " ", posProductVersion: "v1.00"})
   var iclient = new TYRO.IClient("apiKey", {posProductVendor: "Tyro Soft", posProductName: "", posProductVersion: "v1.00"})
   ```
3. The version number must be updated regularly with each new version that is released into production, and all information that is displayed anywhere on the POS interface pertaining to the POS version must be consistent with what is declared in the `posProductData` object in the constructor. e.g. the `posProductData` reflecting the product version to be **v1.00** and the POS interface reflecting the version to be **v1.01.**
4. The POS information must be reflected into the iClient logs as is shown below (line 1) as an example for a 100.0 $ transaction:

```
5. Thu Jan 28 2021 14:42:59 GMT+1100 (Australian Eastern Daylight Time): {"type":"IS request","isUrl":"https://integration.test.tyro.com/terminals/850/437/transactions","httpVerb":"POST","headers":{"X-Tyro-Agent":"iClient/1.0","Pos-Product-Version":"1.0.0","Pos-Product-Name":"Acme Cloud POS","Pos-Product-Vendor":"Acme Co"},"data":"type=purchase&key=93__89&transactionId=4740e5643d8cf24370bbc22b2fc05196677e&integratedReceipt=true&purchaseAmount=10000&cashoutAmount=0&returnExtraDataOnCompletion=true&returnTipDataOnCompletion=true"}
6. Thu Jan 28 2021 14:42:59 GMT+1100 (Australian Eastern Daylight Time): {"type":"IS response: error","statusCode":0,"error":""}
7. Thu Jan 28 2021 14:43:00 GMT+1100 (Australian Eastern Daylight Time): {"type":"IS response","statusCode":200}
8. Thu Jan 28 2021 14:43:00 GMT+1100 (Australian Eastern Daylight Time): {"type":"IS request","isUrl":"https://integration.test.tyro.com/terminals/850/437/transactions/4740e5643d8cf24370bbc22b2fc05196677e","httpVerb":"GET","headers":{"X-Tyro-Agent":"iClient/1.0"},"data":"key=93__89"}
9. Thu Jan 28 2021 14:43:00 GMT+1100 (Australian Eastern Daylight Time): {"type":"IS response","statusCode":200}
10. Thu Jan 28 2021 14:43:00 GMT+1100 (Australian Eastern Daylight Time): {"type":"UI: statusMessageChanged","args":["Press OK to continue"]}
11. Thu Jan 28 2021 14:43:00 GMT+1100 (Australian Eastern Daylight Time): {"type":"IS request","isUrl":"https://integration.test.tyro.com/terminals/850/437/transactions/4740e5643d8cf24370bbc22b2fc05196677e","httpVerb":"GET","headers":{"X-Tyro-Agent":"iClient/1.0"},"data":"key=93__89"}
12. Thu Jan 28 2021 14:43:01 GMT+1100 (Australian Eastern Daylight Time): {"type":"IS response","statusCode":200}
```

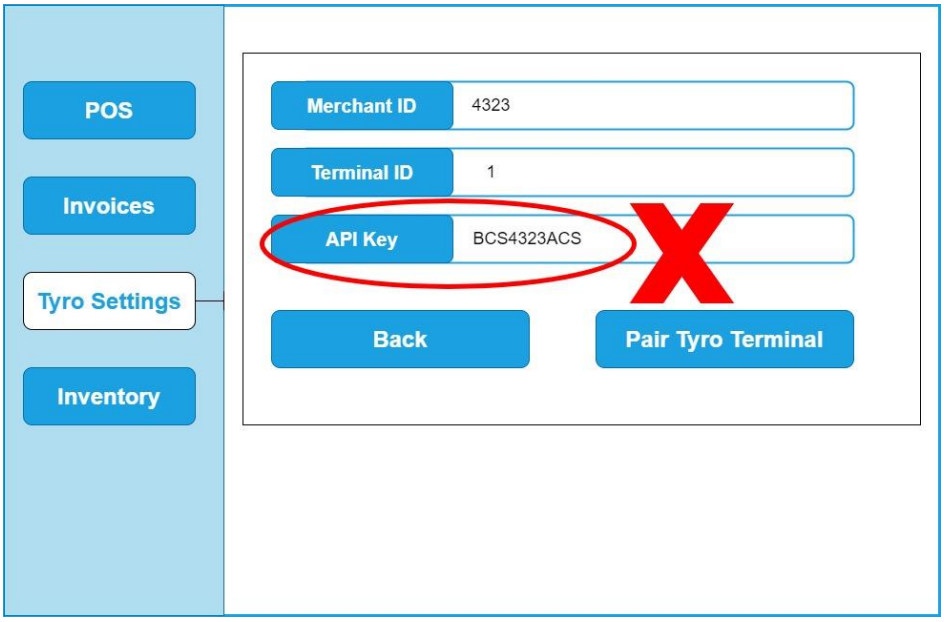
13. Thu Jan 28 2021 14:43:01 GMT+1100 (Australian Eastern Daylight Time): {"type":"UI: statusMessageChanged","args":["Would you like to leave a tip?"]}
14. Thu Jan 28 2021 14:43:01 GMT+1100 (Australian Eastern Daylight Time): {"type":"IS request","isUrl":"https://integration.test.tyro.com/terminals/850/437/transactions/4740e5643d8cf24370bbc22b2fc05196677e","httpVerb":"GET","headers":{"X-Tyro-Agent":"iClient/1.0"},"data":"key=93__89"}
15. Thu Jan 28 2021 14:43:02 GMT+1100 (Australian Eastern Daylight Time): {"type":"IS response","statusCode":200}
16. Thu Jan 28 2021 14:43:02 GMT+1100 (Australian Eastern Daylight Time): {"type":"UI: statusMessageChanged","args":["Swipe / Insert Card. Purchase $100.00"]}
17. Thu Jan 28 2021 14:43:02 GMT+1100 (Australian Eastern Daylight Time): {"type":"IS request","isUrl":"https://integration.test.tyro.com/terminals/850/437/transactions/4740e5643d8cf24370bbc22b2fc05196677e","httpVerb":"GET","headers":{"X-Tyro-Agent":"iClient/1.0"},"data":"key=93__89"}
18. Thu Jan 28 2021 14:43:06 GMT+1100 (Australian Eastern Daylight Time): {"type":"IS response","statusCode":200}
19. Thu Jan 28 2021 14:43:06 GMT+1100 (Australian Eastern Daylight Time): {"type":"UI: statusMessageChanged","args":["Enter PIN"]}
20. Thu Jan 28 2021 14:43:06 GMT+1100 (Australian Eastern Daylight Time): {"type":"IS request","isUrl":"https://integration.test.tyro.com/terminals/850/437/transactions/4740e5643d8cf24370bbc22b2fc05196677e","httpVerb":"GET","headers":{"X-Tyro-Agent":"iClient/1.0"},"data":"key=93__89"}
21. Thu Jan 28 2021 14:43:08 GMT+1100 (Australian Eastern Daylight Time): {"type":"IS response","statusCode":200}
22. Thu Jan 28 2021 14:43:08 GMT+1100 (Australian Eastern Daylight Time): {"type":"UI: statusMessageChanged","args":["Processing transaction - please wait"]}
23. Thu Jan 28 2021 14:43:08 GMT+1100 (Australian Eastern Daylight Time): {"type":"IS request","isUrl":"https://integration.test.tyro.com/terminals/850/437/transactions/4740e5643d8cf24370bbc22b2fc05196677e","httpVerb":"GET","headers":{"X-Tyro-Agent":"iClient/1.0"},"data":"key=93__89"}
24. Thu Jan 28 2021 14:43:11 GMT+1100 (Australian Eastern Daylight Time): {"type":"IS response","statusCode":200}
25. Thu Jan 28 2021 14:43:11 GMT+1100 (Australian Eastern Daylight Time): {"type":"Resp Interface: receiptReceived","args":[{"merchantReceipt":" MERCHANT COPY \n\n Pay@table test \n 123 test street \n Sydney NSW 2000 \n\n Tyro Payments EFTPOS \n\nNAB Visa Credit\nAID: A0000000031010\nCard: xxxxxxxxxxxx9521(t)\n\nPurchase AUD $100.00\nSurcharge AUD $0.50\n ----------\nTotal AUD $100.50\n\nAPPROVED 00\n\nTerminal ID: 437\nTransaction Ref: 460042\nAuthorisation No: 000007\n28 Jan 2021 at 02:43 PM\n","signatureRequired":false}]}
26. Thu Jan 28 2021 14:43:11 GMT+1100 (Australian Eastern Daylight Time): {"type":"UI: statusMessageChanged","args":["APPROVED"]}
27. Thu Jan 28 2021 14:43:11 GMT+1100 (Australian Eastern Daylight Time): {"type":"IS request","isUrl":"https://integration.test.tyro.com/terminals/850/437/transactions/4740e5643d8cf24370bbc22b2fc05196677e","httpVerb":"GET","headers":{"X-Tyro-Agent":"iClient/1.0"},"data":"key=93__89"}
28. Thu Jan 28 2021 14:43:11 GMT+1100 (Australian Eastern Daylight Time): {"type":"IS response","statusCode":200}

29. Thu Jan 28 2021 14:43:11 GMT+1100 (Australian Eastern Daylight Time): {"type":"UI: statusMessageChanged","args":["APPROVED"]}
30. Thu Jan 28 2021 14:43:11 GMT+1100 (Australian Eastern Daylight Time): {"type":"Resp Interface: transactionComplete","args":[{"result":"APPROVED","transactionId":"4740e5643d8cf24370bbc22b2fc05196677e","cardType":"Visa","transactionReference":"460042","authorisationCode":"000007","issuerActionCode":"00","elidedPan":"xxxxxxxxxxxx9521","rrn":"102814460042","baseAmount":"100.00","transactionAmount":"100.00","customerReceipt":" CUSTOMER COPY \n\n Pay@table test \n 123 test street \n Sydney NSW 2000 \n\n Tyro Payments EFTPOS \n\nNAB Visa Credit\nAID: A0000000031010\nCard: xxxxxxxxxxxx9521(t)\n\nPurchase AUD $100.00\nSurcharge AUD $0.50\n ----------\nTotal AUD $100.50\n\nAPPROVED 00\n\nTerminal ID: 437\nTransaction Ref: 460042\nAuthorisation No: 000007\n28 Jan 2021 at 02:43 PM\n"}]}
31. Thu Jan 28 2021 14:43:11 GMT+1100 (Australian Eastern Daylight Time): {"type":"window.onerror","error":"Uncaught TypeError: Cannot set property 'completed' of null","url":"https://iclient.test.tyro.com/js/d62c25bd.tta-brain.min.js","lineNumber":8}

API Key configuration

iClient integration uses an API key that is specific to the POS product to authenticate requests against the integration server, upon being certified an API key is allocated to the POS product and sent to the relevant points-of-contact for the respective POS partner to allow the configuration of mutual merchants in the production environment.
The criteria for API Key configuration is as given below:

1. The API Key must not be visible, configurable, or editable by any merchants, through any front-end user interface component, an example of this is given below where it can be seen that the POS is showing the API key in the front-end interface and allowing the merchants to edit and configure the API key through a free-text field, which must never be done.



2. The API key must not be disclosed externally and must be used by the POS partners to configure the POS for the production environment in the back-end of their internal system before deployment to the merchant/s.
3. The POS Partner must only use a valid production API key which is communicated by Tyro to the POS partner once the POS product has been officially certified.
4. The API key must be unique for each integration product, this means that a POS partner must not reuse an API keys given out for a previously certified POS product to try and use a newly-developed integration product in the production environment.