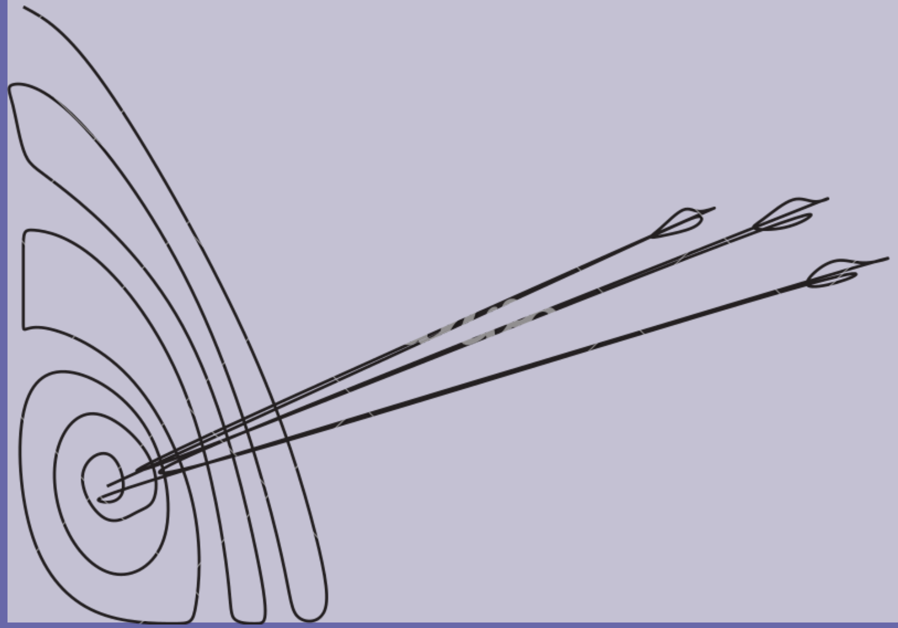
BOA CODE TO CONNECT - GMOT

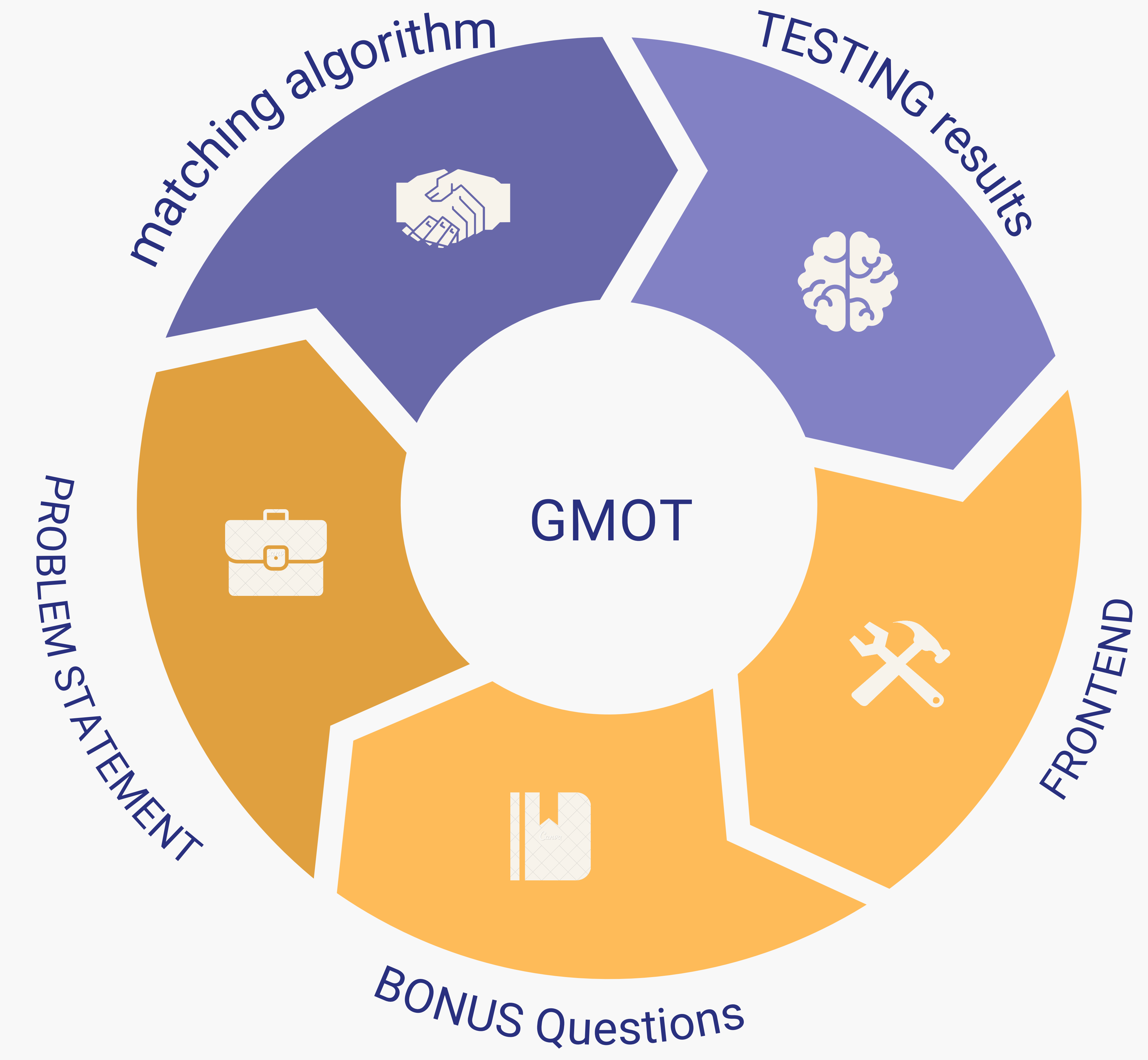
JAYATI PARWANI

BANK OF AMERICA

Client Assets Protection and Reconciliations



PRESENTATION CONTENTS



Breakdown of Problem Statement

1

READ AND PARSE SOURCE FILES

- Parse Schematic Ledger and Swift Statement fast
- Extract Important & Necessary Information for comparison

2

WRITE A MATCHING ALGORITHM

Implement

- One-to-One Matching [1:1]
- One-to-Many Matching [1:N]
- Many-to-Many Matching [M:N]

3

ACCURATELY IDENTIFY CLIENT MONEY BREAKS

- Full Integrity Check : Identify Root Cause of Balance Break
- Proofing Check: reconciliation in Proof

Design Considerations

Assumptions and Design Considerations

Client Friendly Interface

Simple, concise UI that **segregates** balance and transaction reconciliation results in **tables, not overtly technical**

Detailed Information

Provide **details** on where the break is, the **difference** in the money and number of **entries** in **batch** transactions.

Efficient, Accurate and Safe Algorithm

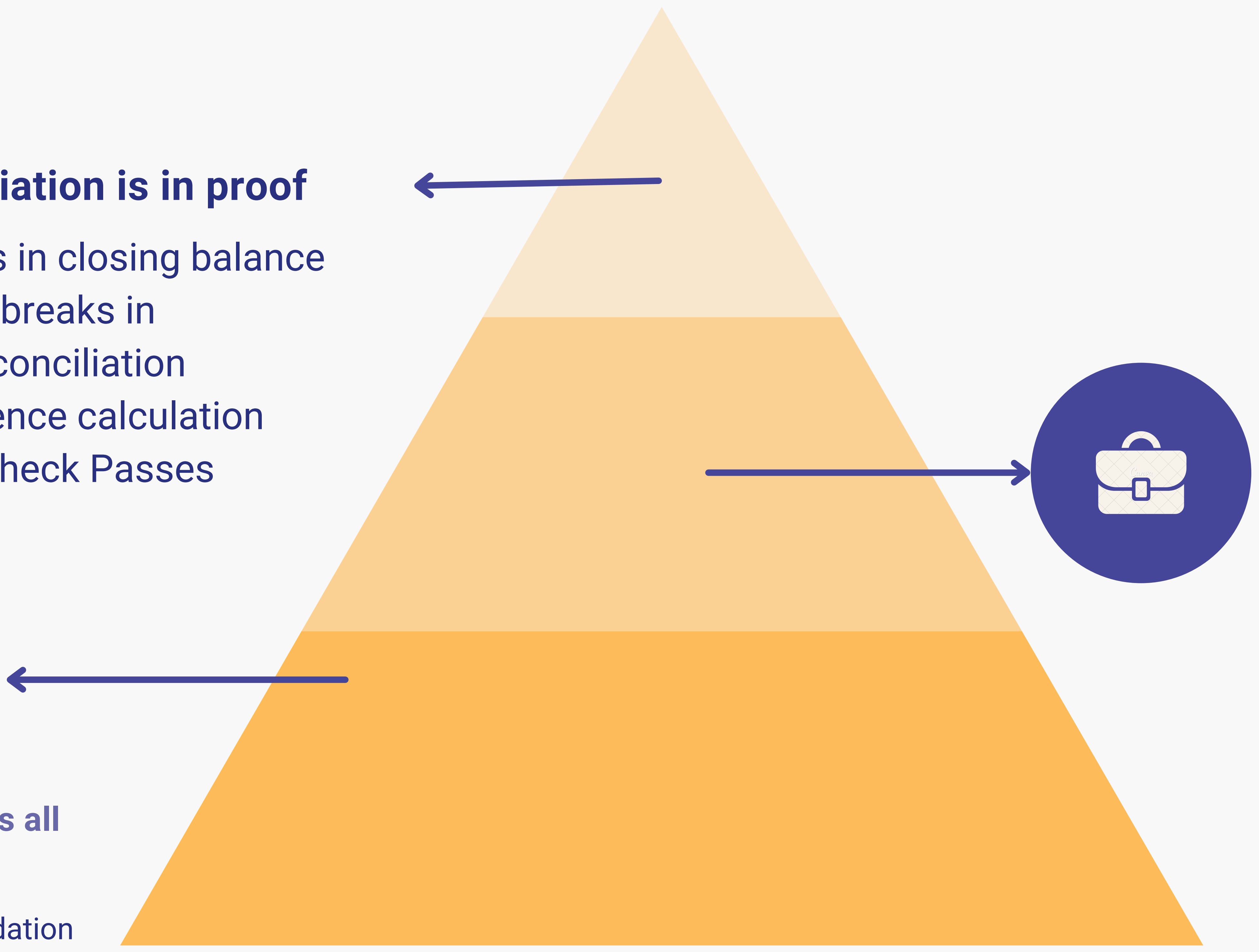
SWIFT sample given has **all necessary data** corresponding to the Ledger (account number/date) , and break is in reconciliation. Algorithm doesn't use any cloud-database service and is hosted locally for added **safety**

Modular Programming that handles exceptions

Code can be **applied** to any SWIFT Messages and Ledger Data
Code segregated into functions and handle exception cases

Assumptions:
the XML is **valid**, **well-formed**, and **conforms to the SWIFT ISO20022 XML Schema Definition (XSD)**
Dates are **consistent** in ledger and SWIFT XML

Proof Checking Methodology



Check Reconciliation is in proof

- Verify if breaks in closing balance correspond to breaks in transaction reconciliation through difference calculation
- If true, Proof Check Passes

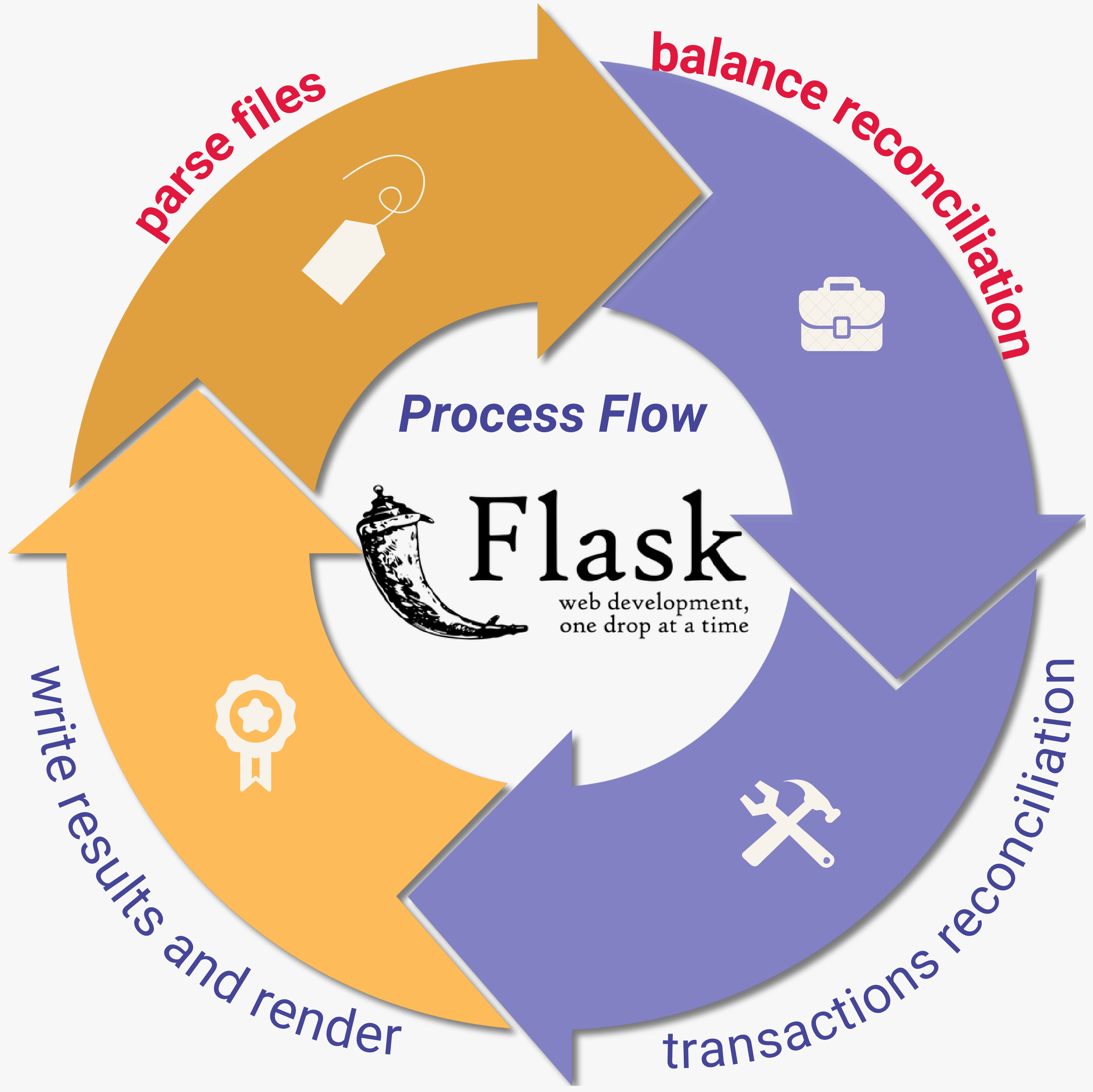
Swift Metadata corresponds to Ledger

- Verify if account number is the same in SWIFT and Ledger
- Verify currency and type of balance(debit, credit) corresponds in Ledger and Swift

Check if Swift Data contains all necessary parameters

- Sender and Receiver BIC Validation
- Statement ID
- Account Number
- Grp Header: Message, Message ID, Creation Date Time

Full Integrity Check Methodology- I



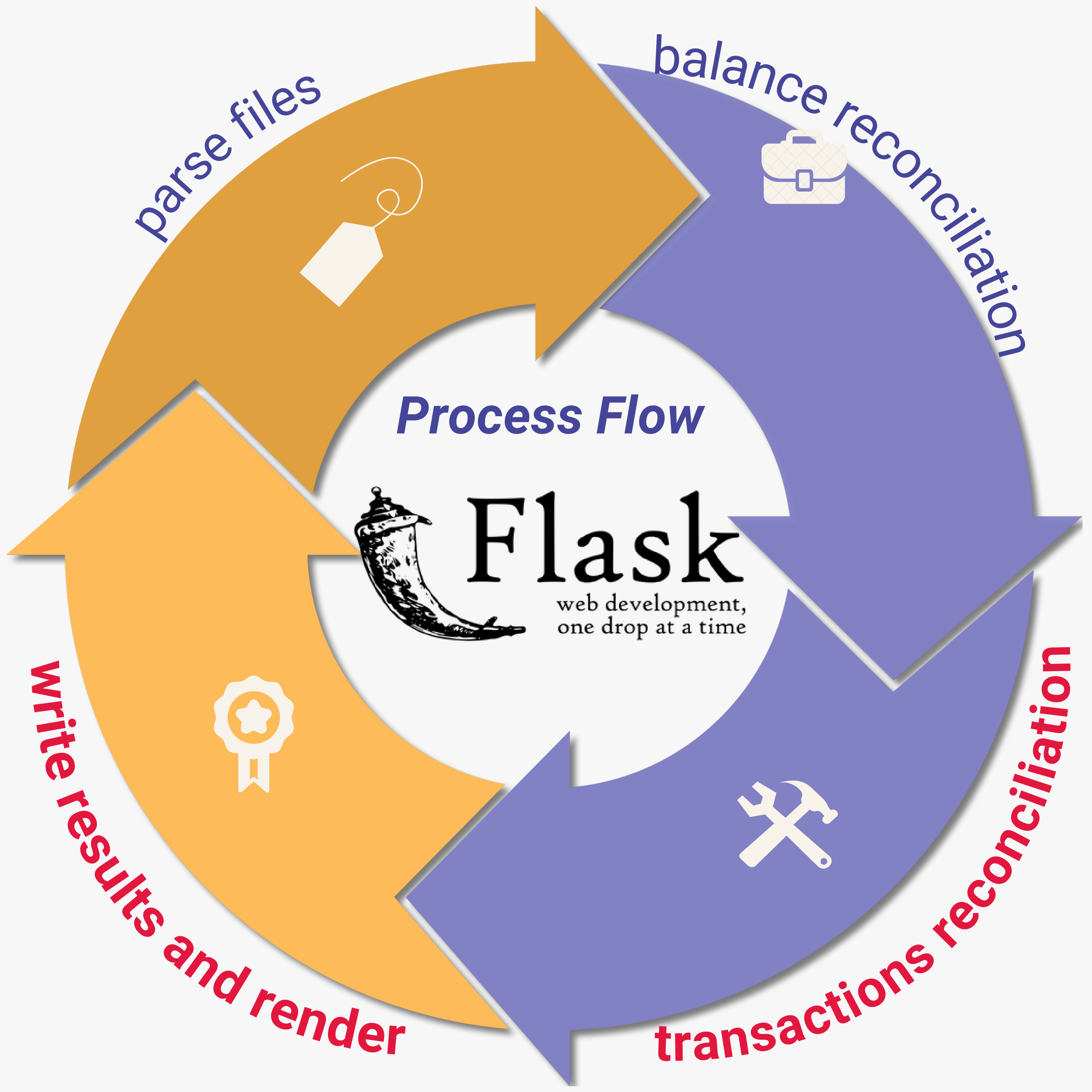
Parse Ledger and SWIFT XML

- Parse Ledger Data in 2 separate pandas data frames for: balance reconciliation and transaction reconciliation
- Parse SWIFT data in 3 separate dictionaries for clarity: balances(opening and closing), all transaction entries, batch entries (for efficient aggregate testing)

Matching Algorithm Balance Reconciliation

- Compare opening and closing balance in ledger and SWIFT Data.
- If breaks, create a pandas data-frame to record the break and record difference in balance due to inconsistency

Full Integrity Check Methodology- II



Matching Algorithm Transactions

- Run 1:1 Matching on all transaction entries.
- Record Results
- On batch Entries, run aggregate matching. Compute number of entries per batch transaction in ledger and SWIFT. Report differences if any in a pandas data-frame
- Analyse if difference is consistent with balance reconciliation results

Write results directly to HTML from Python script
Render using Flask Framework

Bonus Question- Matching Algorithm

Bonus Question 3: Characteristics and Considerations of the Matching Algorithm

Additional Proofing Implementation
Validation of Swift XML Data against the Ledger Data such as currency, credit/debit

Effective Time and Space Complexity

- Fast Parsing of Relevant Information
- Functions Aggregate Matching and 1:1 matching for **modularity** and **decomposition**

Breaks explained in detail and Proof Checking through Difference Computation

- Number of Entries calculated in batch transaction to identify root cause of breaks
- Neatly displayed

Easy to Run, Organised Information Presentation, Flexible

- Use of Pandas DataFrame and Flask Framework
- Matching Algorithm can be **applied** to **any** SWIFT XML and Ledger Data

Results- Balance Reconciliation

	Account	Currency	Balance	Date	Type	Source	Difference
0	5491-00003011-MULT	SGD	12,052,345.27	2022-07-04T11:00:00+00:00	CLBD	Ledger	NaN
1	5491-00003011-MULT	SGD	15,052,345.27	2022-07-04	CLBD	SWIFT	3,000,000

- **Opening Balance is the same**

Opening Balance is the same for both ledger and SWIFT

- **Closing Balance has a break**

Ledger Closing Balance is not the same as SWIFT statement

- **There is a difference of SGD $3,000,000 in closing balance**

- **SWIFT Closing Balance exceeds by $3,000,000**

Results- Transaction Reconciliation

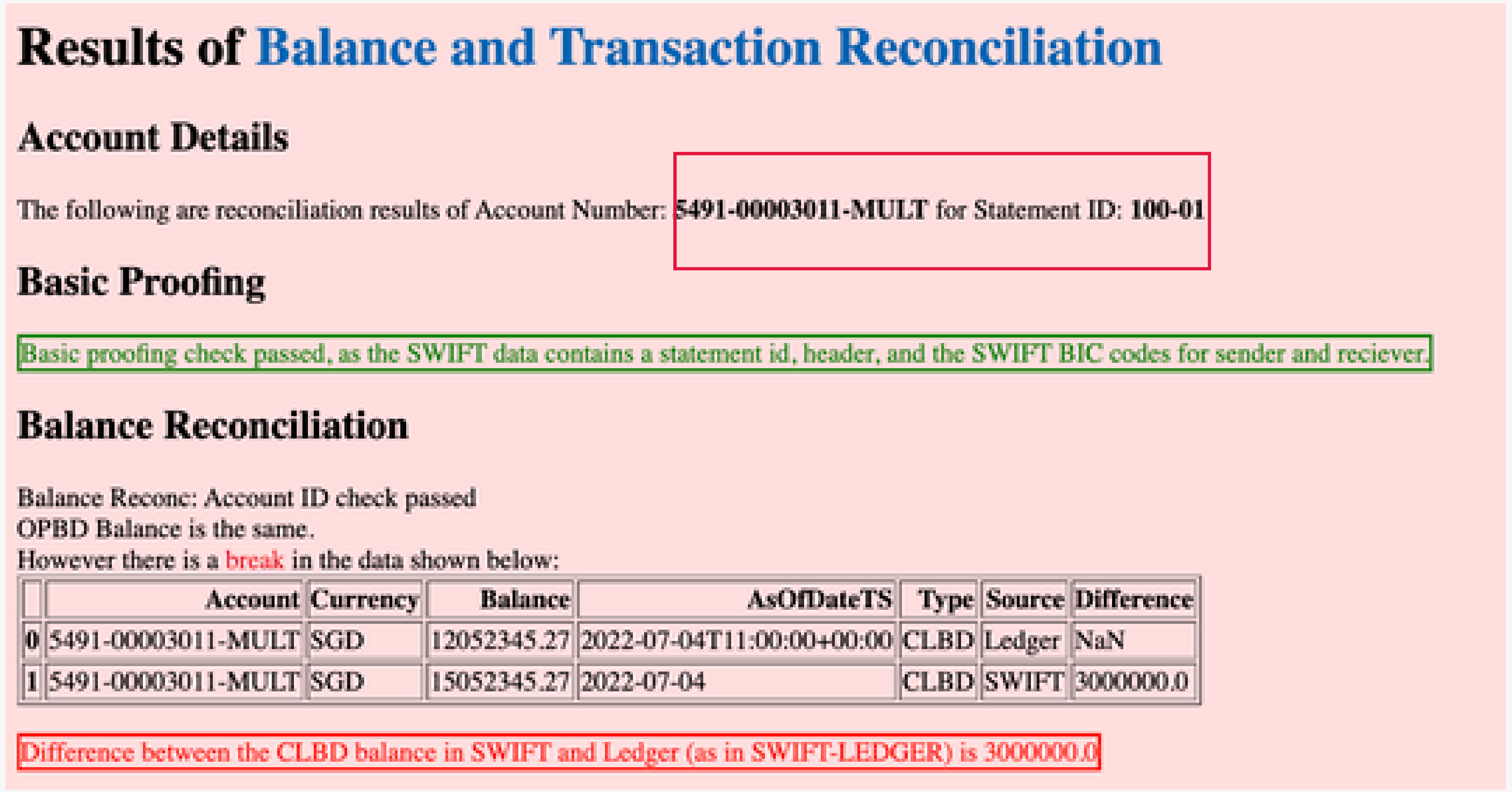
	Account	Balance	TotalAmount	Enteries	Type	MessageID	PaymentID	Difference
0	5491-00003011-MULT	1000000	11000000.0	11	Ledger	None	None	NaN
1	5491-00003011-MULT	1000000	10000000.0	10	SWIFT	FINP-0055/002	FINP-0055/002	-1000000.0
0	5491-00003011-MULT	2000000	12000000.0	6	Ledger	None	None	NaN
1	5491-00003011-MULT	2000000	10000000.0	5	SWIFT	FINP-0055/003	FINP-0055/003	-2000000.

- **All transactions passed 1:1 Matching**
- **For batch transactions there were two entries with breaks**

For the entry of $1 million, there were 10 such transactions in SWIFT but 11 such transactions in the ledger. Similarly, for the entry of $2 million, there were 5 such transactions in SWIFT but 6 such transactions in the ledger.
This explains why the closing balance of the ledger was $3 million less compared to SWIFT, as SWIFT data missed 2 transactions worth this difference

- **This results in a difference of $3,000,000 in transaction data.**
- **Reconciliation is in proof as transaction and balance breaks are consistent.**

Frontend Display- Balance Reconciliation



Results of Balance and Transaction Reconciliation

Account Details

The following are reconciliation results of Account Number: 5491-00003011-MULT for Statement ID: 100-01

Basic Proofing

Basic proofing check passed, as the SWIFT data contains a statement id, header, and the SWIFT BIC codes for sender and reciever.

Balance Reconciliation

Balance Reconc: Account ID check passed
OPBD Balance is the same.
However there is a break in the data shown below:

	Account	Currency	Balance	AsOfDateTS	Type	Source	Difference
0	5491-00003011-MULT	SGD	12052345.27	2022-07-04T11:00:00+00:00	CLBD	Ledger	NaN
1	5491-00003011-MULT	SGD	15052345.27	2022-07-04	CLBD	SWIFT	3000000.0

Difference between the CLBD balance in SWIFT and Ledger (as in SWIFT-LEDGER) is 3000000.0

Display Account Details

Fetch Statement ID and Account ID from SWIFT XML

Basic Proofing

Validation results of checking statement id, group-header, validation of SWIFT BIC codes

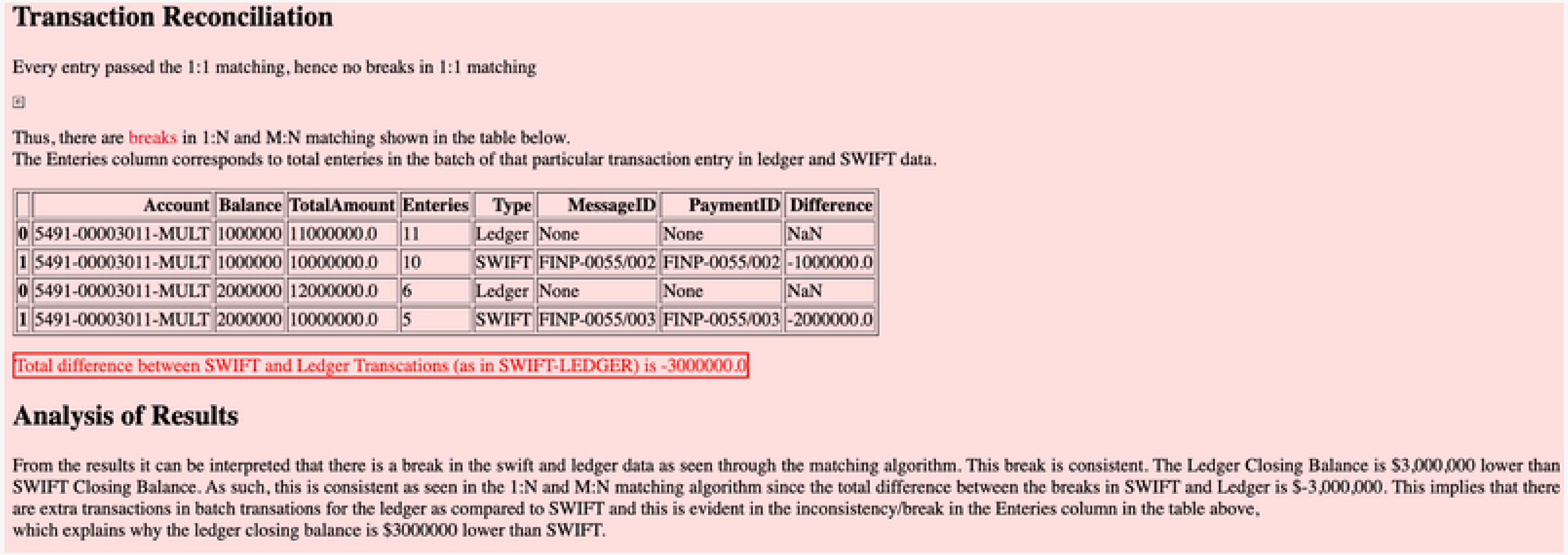
Balance Reconciliation

Check Account ID is consistent.
Report if there are breaks in opening/closing bala

Tabular Display

Table to identify root cause of breaks. Compute and display the difference in case of breaks

Frontend Display- Transaction Reconciliation



Transaction Reconciliation

Every entry passed the 1:1 matching, hence no breaks in 1:1 matching

Thus, there are breaks in 1:N and M:N matching shown in the table below.
The Enteries column corresponds to total enteries in the batch of that particular transaction entry in ledger and SWIFT data.

	Account	Balance	TotalAmount	Enteries	Type	MessageID	PaymentID	Difference
0	5491-00003011-MULT	1000000	11000000.0	11	Ledger	None	None	NaN
1	5491-00003011-MULT	1000000	10000000.0	10	SWIFT	FINP-0055/002	FINP-0055/002	-1000000.0
0	5491-00003011-MULT	2000000	12000000.0	6	Ledger	None	None	NaN
1	5491-00003011-MULT	2000000	10000000.0	5	SWIFT	FINP-0055/003	FINP-0055/003	-2000000.0

Total difference between SWIFT and Ledger Transcations (as in SWIFT-LEDGER) is -3000000.0

Analysis of Results

From the results it can be interpreted that there is a break in the swift and ledger data as seen through the matching algorithm. This break is consistent. The Ledger Closing Balance is $3,000,000 lower than SWIFT Closing Balance. As such, this is consistent as seen in the 1:N and M:N matching algorithm since the total difference between the breaks in SWIFT and Ledger is $-3,000,000. This implies that there are extra transactions in batch transations for the ledger as compared to SWIFT and this is evident in the inconsistency/break in the Enteries column in the table above,
which explains why the ledger closing balance is $3000000 lower than SWIFT.

1:1 Matching Result

Report result of 1:1 matching, if breaks display exceptions

Aggregate Matching

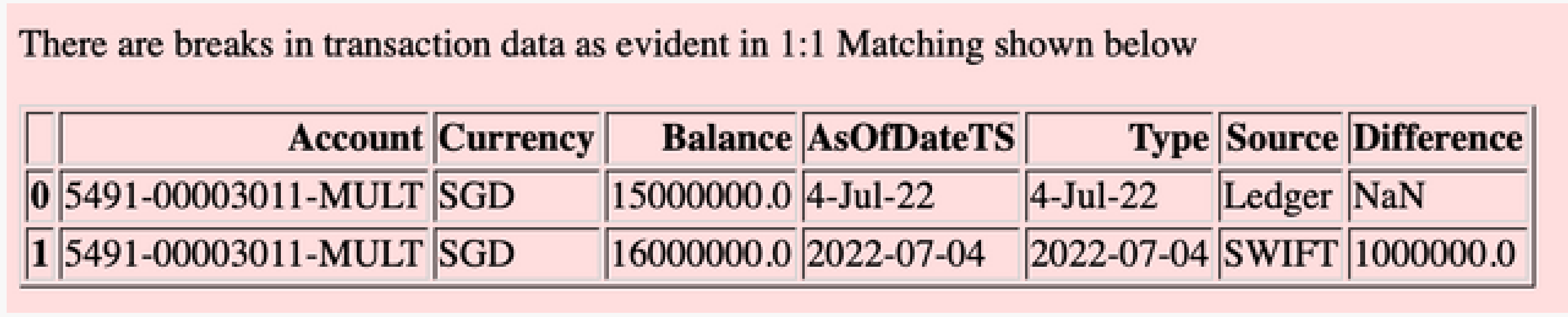
Report result of 1:1 matching, if breaks display exceptions and difference

Analysis of Results

Report an Analysis of Results and whether reconciliation is in proof and consistent with balance and transaction results

Testing- Exception Cases

- Exceptions have been handled in the code and test cases passed have been displayed on the interface.
- In our results breaks were displayed for balance reconciliation and transaction reconciliation (but not for 1:1 matching as there were no breaks)
- A hypothetical case where there is a break in one of the entries of 1:1 matching has been tested as shown



There are breaks in transaction data as evident in 1:1 Matching shown below

	Account	Currency	Balance	AsOfDateTS	Type	Source	Difference
0	5491-00003011-MULT	SGD	15000000.0	4-Jul-22	4-Jul-22	Ledger	NaN
1	5491-00003011-MULT	SGD	16000000.0	2022-07-04	2022-07-04	SWIFT	1000000.0

Hence, 1:1 Matching can handle breaks too

Further Improvements and Considerations

Use of Cloud Hosted Database and Encryption

To save local storage and ensure security of transactions, encryption-decryption and could database can be used

Interactive User Interface

The analysis and explanation of results is currently hardcoded. Analysis can be generalised in the future.

Dynamic Programming and Testing for Further Optimisation

Algorithm can be further optimised using memoization or recursion for aggregate and 1:1 matching in future implementation
Unit Testing to be carried

Bonus Questions 1 and 2

Swift Data Validations

Validate Swift BIC Code (implemented)

Validate other parameters than balance amount, such as currency, credit or debit, date, account details (implemented)

Encryption and Decryption of sensitive data such as transaction entries to allow validation as both sender and receiving counter-party. (Future Consideration)

Prevent Duplicate Swift Messages

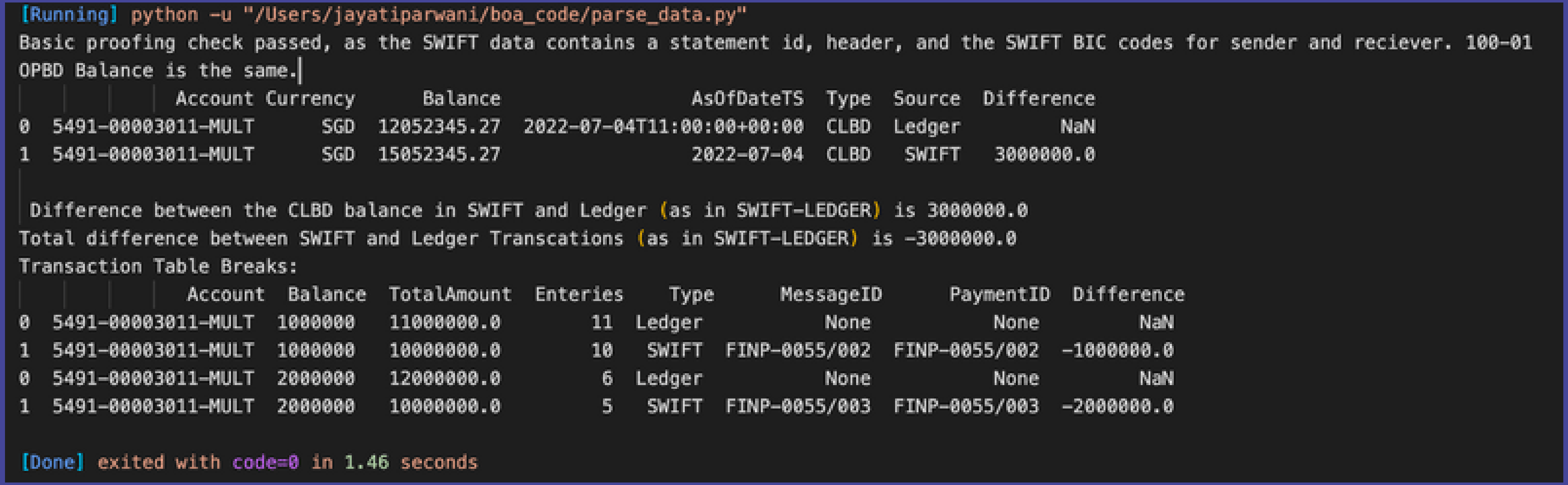
Track and record a unique **Transaction Entry ID** for each transaction along with the Statement and Message ID.

Alert Admin in case **duplicate entries** of the above **ids** and **remove** them.

Code Access Guide- Python Script

Initial Steps

- Ensure python3 is installed in your desktop
- cd to boa_code_jayati folder
- compile and run parse_data.py. Output console should display the results and an index.html is created/updated in templates folder.
- You should see this on your output console:



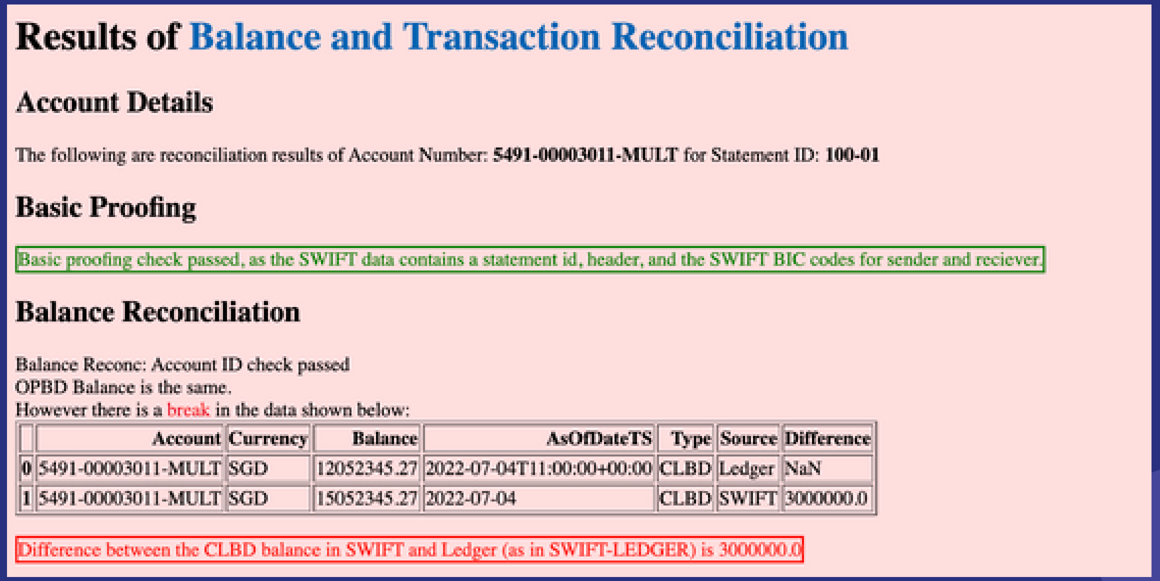
```
[Running] python -u "/Users/jayatiparwani/boa_code/parse_data.py"
Basic proofing check passed, as the SWIFT data contains a statement id, header, and the SWIFT BIC codes for sender and reciever. 100-01
OPBD Balance is the same.
              Account Currency     Balance               AsOfDateTS  Type  Source  Difference
0  5491-00003011-MULT      SGD  12052345.27  2022-07-04T11:00:00+00:00  CLBD  Ledger         NaN
1  5491-00003011-MULT      SGD  15052345.27               2022-07-04  CLBD   SWIFT   3000000.0

 Difference between the CLBD balance in SWIFT and Ledger (as in SWIFT-LEDGER) is 3000000.0
Total difference between SWIFT and Ledger Transcations (as in SWIFT-LEDGER) is -3000000.0
Transaction Table Breaks:
               Account  Balance  TotalAmount  Enteries    Type      MessageID      PaymentID  Difference
0  5491-00003011-MULT  1000000   11000000.0        11  Ledger           None           None         NaN
1  5491-00003011-MULT  1000000   10000000.0        10   SWIFT  FINP-0055/002  FINP-0055/002  -1000000.0
0  5491-00003011-MULT  2000000   12000000.0         6  Ledger           None           None         NaN
1  5491-00003011-MULT  2000000   10000000.0         5   SWIFT  FINP-0055/003  FINP-0055/003  -2000000.0

[Done] exited with code=0 in 1.46 seconds
```

Code Access Guide- Flask

Expected Results



Results of Balance and Transaction Reconciliation

Account Details

The following are reconciliation results of Account Number: **5491-00003011-MULT** for Statement ID: **100-01**

Basic Proofing

Basic proofing check passed, as the SWIFT data contains a statement id, header, and the SWIFT BIC codes for sender and reciever.

Balance Reconciliation

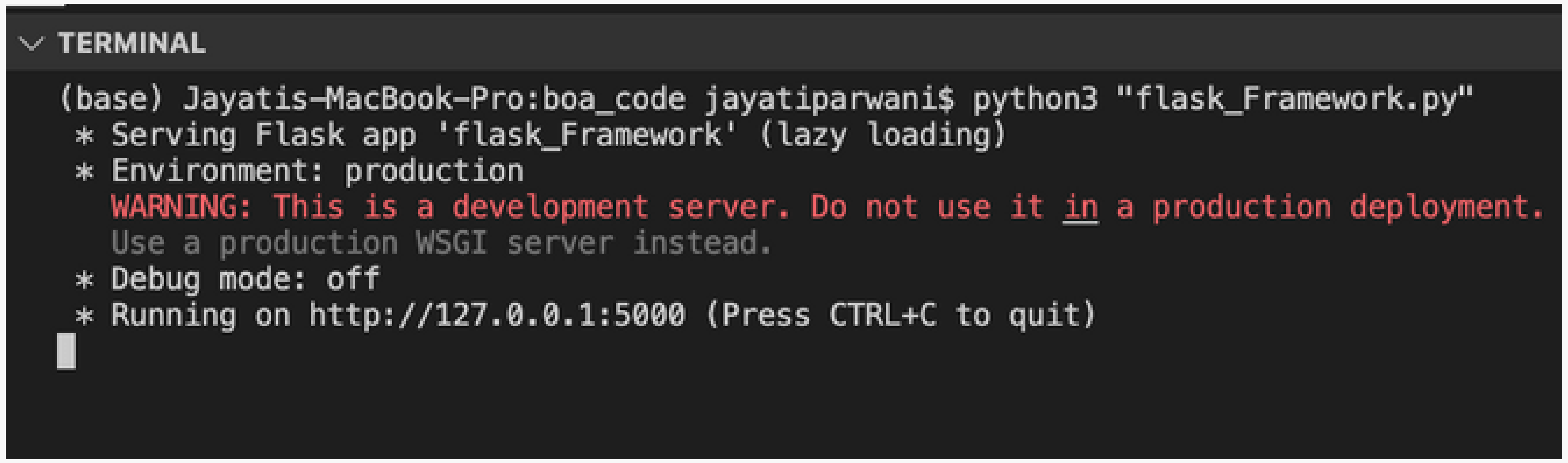
Balance Reconc: Account ID check passed
OPBD Balance is the same.
However there is a break in the data shown below:

	Account	Currency	Balance	AsOfDateTS	Type	Source	Difference
0	5491-00003011-MULT	SGD	12052345.27	2022-07-04T11:00:00+00:00	CLBD	Ledger	NaN
1	5491-00003011-MULT	SGD	15052345.27	2022-07-04	CLBD	SWIFT	3000000.0

Difference between the CLBD balance in SWIFT and Ledger (as in SWIFT-LEDGER) is 3000000.0

Run Flask

1. Start a new terminal session.
2. cd to boa_code_jayati directory. (You may not need to do this if you are running vs code terminal)
3. run --> $ python3 "flask_framework.py" in your terminal <refer to image>
4. If you see the link: http://127.0.0.1:5000 in your terminal, copy paste this link in your browser (preferred: Chrome/Safari) and you should be able to see the results in the expected results screenshot.



```
(base) Jayatis-MacBook-Pro:boa_code jayatiparwani$ python3 "flask_Framework.py"
 * Serving Flask app 'flask_Framework' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5000 (Press CTRL+C to quit)
```

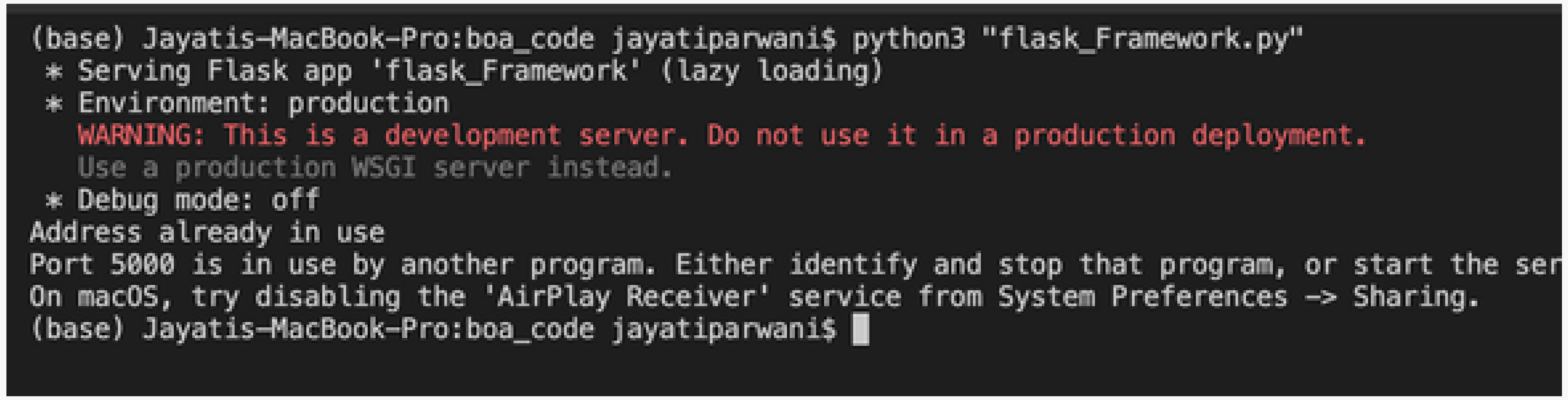
Code Access Guide- Case of Errors

In case of you are unable to view expected results and get the errors:

1. If flask is not installed, install flask using the command:

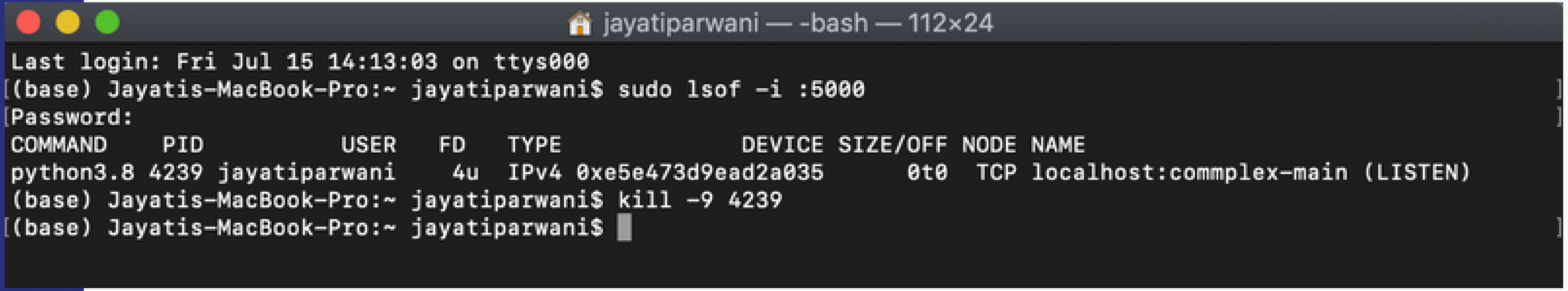
   run --> $ pip3 install flask after you cd to boa_code_directory

2. If you get the error below, your port 5000 is already in use.



```
(base) Jayatis-MacBook-Pro:boa_code jayatiparwani$ python3 "flask_Framework.py"
 * Serving Flask app 'flask_Framework' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
Address already in use
Port 5000 is in use by another program. Either identify and stop that program, or start the ser
On macOS, try disabling the 'AirPlay Receiver' service from System Preferences -> Sharing.
(base) Jayatis-MacBook-Pro:boa_code jayatiparwani$ 
```

- If you get the error below, your port 5000 is already in use.
- kill this terminal session, start a new terminal session.
- run --> $ sudo lsof -i :3000
- enter your laptop Password
- run --> $ kill -9 <PID>
- PID above is the number you see in the table. Refer to screenshot of steps below
- Re-run steps in the last slide.
- In case, there are still errors, please open index.html directly from templates folder in boa_code_jayati folder and I have also attached a pdf of expected results called “results_on_webpage” pdf



```
Last login: Fri Jul 15 14:13:03 on ttys000
(base) Jayatis-MacBook-Pro:~ jayatiparwani$ sudo lsof -i :5000
Password:
COMMAND     PID           USER    FD    TYPE               DEVICE SIZE/OFF NODE NAME
python3.8 4239 jayatiparwani    4u  IPv4 0xe5e473d9ead2a035      0t0  TCP localhost:commplex-main (LISTEN)
(base) Jayatis-MacBook-Pro:~ jayatiparwani$ kill -9 4239
(base) Jayatis-MacBook-Pro:~ jayatiparwani$ 
```