Building and Securing a REST API for MoMo SMS Transactions

1. Introduction to API Security

An API exposes application data and functionality to external clients. This makes security a core concern. Without proper controls, APIs can be misused, abused, or compromised.API security focuses on three main goals:

- Authentication ensures that the client is who they claim to be.
- Authorization ensures that the client is allowed to perform a specific action.
- Validation ensures that only correct and safe data is processed.

In this project, security was addressed at a basic level using HTTP Basic Authentication and strict request validation. Even though Basic Authentication is simple, it demonstrates how access control and error handling should be applied consistently across endpoints.

2. API Endpoint Documentation

This API manages transaction records stored in memory as a list of dictionaries. Each transaction has a fixed structure and strict validation rules.

Common Rules for All Endpoints

- All responses use JSON format.
- Unsupported HTTP methods return 405 Method Not Allowed.
- Invalid routes return 404 Not Found.
- Server errors return 500 Internal Server Error.
- Authentication is required for protected endpoints.
- Invalid credentials return 401 Unauthorized.

Base URL - http://localhost:8000

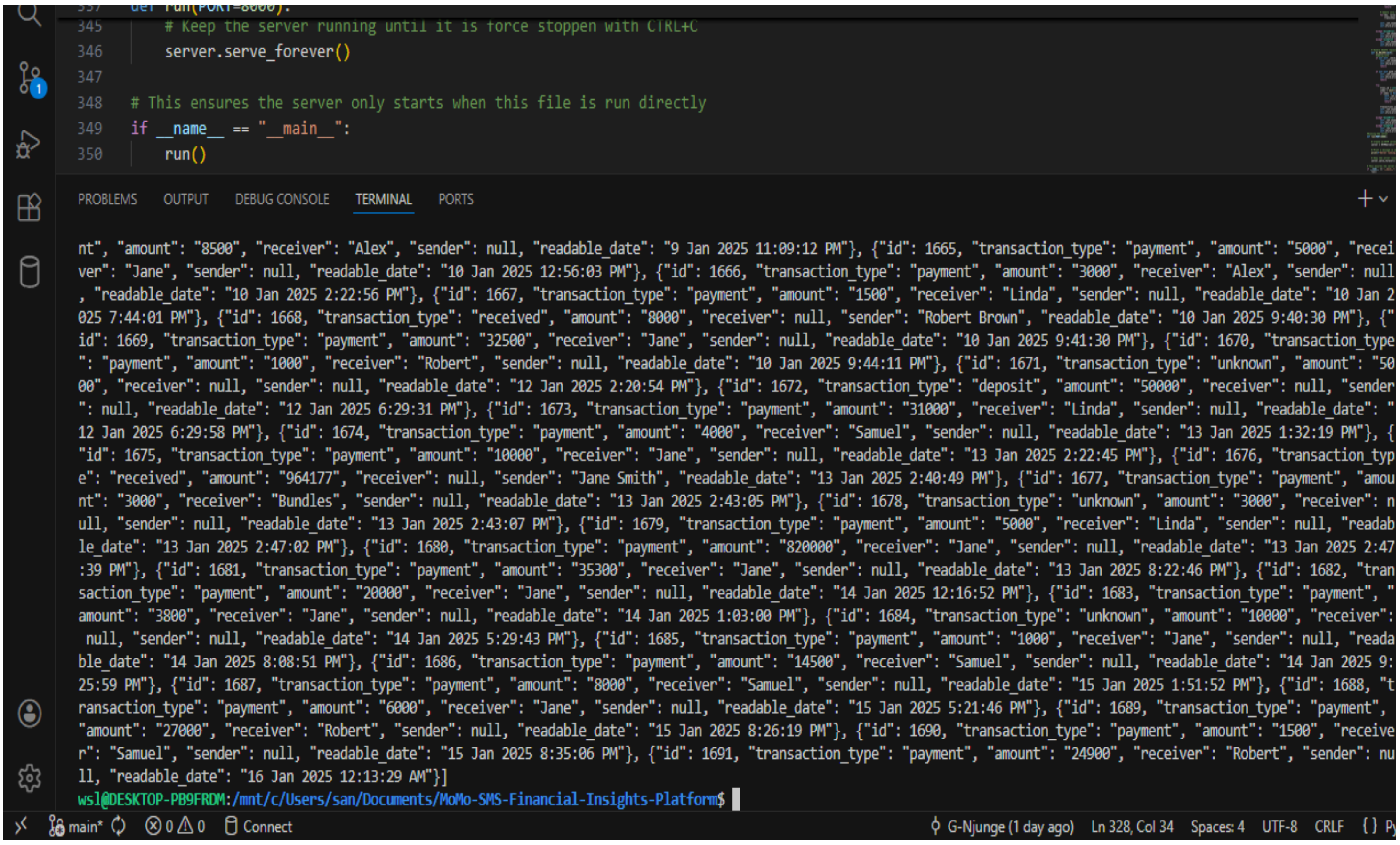
2.1 GET /transactions

Returns a list of all SMS transaction records.

Request Example:

```
curl -u admin:admin123 http://localhost:8000/transactions
```

Successful Response (200 OK):

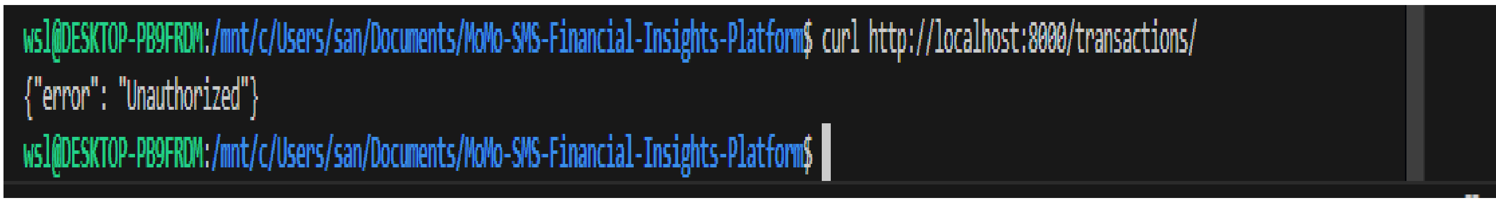


```
def run(PORT=8000):
    # Keep the server running until it is force stoppen with CTRL+C
    server.serve_forever()

# This ensures the server only starts when this file is run directly
if __name__ == "__main__":
    run()
```

```
nt", "amount": "8500", "receiver": "Alex", "sender": null, "readable_date": "9 Jan 2025 11:09:12 PM"}, {"id": 1665, "transaction_type": "payment", "amount": "5000", "recei
ver": "Jane", "sender": null, "readable_date": "10 Jan 2025 12:56:03 PM"}, {"id": 1666, "transaction_type": "payment", "amount": "3000", "receiver": "Alex", "sender": null
, "readable_date": "10 Jan 2025 2:22:56 PM"}, {"id": 1667, "transaction_type": "payment", "amount": "1500", "receiver": "Linda", "sender": null, "readable_date": "10 Jan 2
025 7:44:01 PM"}, {"id": 1668, "transaction_type": "received", "amount": "8000", "receiver": null, "sender": "Robert Brown", "readable_date": "10 Jan 2025 9:40:30 PM"}, {"
id": 1669, "transaction_type": "payment", "amount": "32500", "receiver": "Jane", "sender": null, "readable_date": "10 Jan 2025 9:41:30 PM"}, {"id": 1670, "transaction_type
": "payment", "amount": "1000", "receiver": "Robert", "sender": null, "readable_date": "10 Jan 2025 9:44:11 PM"}, {"id": 1671, "transaction_type": "unknown", "amount": "50
00", "receiver": null, "sender": null, "readable_date": "12 Jan 2025 2:20:54 PM"}, {"id": 1672, "transaction_type": "deposit", "amount": "50000", "receiver": null, "sender
": null, "readable_date": "12 Jan 2025 6:29:31 PM"}, {"id": 1673, "transaction_type": "payment", "amount": "31000", "receiver": "Linda", "sender": null, "readable_date": "
12 Jan 2025 6:29:58 PM"}, {"id": 1674, "transaction_type": "payment", "amount": "4000", "receiver": "Samuel", "sender": null, "readable_date": "13 Jan 2025 1:32:19 PM"}, {
"id": 1675, "transaction_type": "payment", "amount": "10000", "receiver": "Jane", "sender": null, "readable_date": "13 Jan 2025 2:22:45 PM"}, {"id": 1676, "transaction_typ
e": "received", "amount": "964177", "receiver": null, "sender": "Jane Smith", "readable_date": "13 Jan 2025 2:40:49 PM"}, {"id": 1677, "transaction_type": "payment", "amou
nt": "3000", "receiver": "Bundles", "sender": null, "readable_date": "13 Jan 2025 2:43:05 PM"}, {"id": 1678, "transaction_type": "unknown", "amount": "3000", "receiver": n
ull, "sender": null, "readable_date": "13 Jan 2025 2:43:07 PM"}, {"id": 1679, "transaction_type": "payment", "amount": "5000", "receiver": "Linda", "sender": null, "readab
le_date": "13 Jan 2025 2:47:02 PM"}, {"id": 1680, "transaction_type": "payment", "amount": "820000", "receiver": "Jane", "sender": null, "readable_date": "13 Jan 2025 2:47
:39 PM"}, {"id": 1681, "transaction_type": "payment", "amount": "35300", "receiver": "Jane", "sender": null, "readable_date": "13 Jan 2025 8:22:46 PM"}, {"id": 1682, "tran
saction_type": "payment", "amount": "20000", "receiver": "Jane", "sender": null, "readable_date": "14 Jan 2025 12:16:52 PM"}, {"id": 1683, "transaction_type": "payment", "
amount": "3800", "receiver": "Jane", "sender": null, "readable_date": "14 Jan 2025 1:03:00 PM"}, {"id": 1684, "transaction_type": "unknown", "amount": "10000", "receiver":
 null, "sender": null, "readable_date": "14 Jan 2025 5:29:43 PM"}, {"id": 1685, "transaction_type": "payment", "amount": "1000", "receiver": "Jane", "sender": null, "reada
ble_date": "14 Jan 2025 8:08:51 PM"}, {"id": 1686, "transaction_type": "payment", "amount": "14500", "receiver": "Samuel", "sender": null, "readable_date": "14 Jan 2025 9:
25:59 PM"}, {"id": 1687, "transaction_type": "payment", "amount": "8000", "receiver": "Samuel", "sender": null, "readable_date": "15 Jan 2025 1:51:52 PM"}, {"id": 1688, "t
ransaction_type": "payment", "amount": "6000", "receiver": "Jane", "sender": null, "readable_date": "15 Jan 2025 5:21:46 PM"}, {"id": 1689, "transaction_type": "payment",
"amount": "27000", "receiver": "Robert", "sender": null, "readable_date": "15 Jan 2025 8:26:19 PM"}, {"id": 1690, "transaction_type": "payment", "amount": "1500", "receive
r": "Samuel", "sender": null, "readable_date": "15 Jan 2025 8:35:06 PM"}, {"id": 1691, "transaction_type": "payment", "amount": "24900", "receiver": "Robert", "sender": nu
ll, "readable_date": "16 Jan 2025 12:13:29 AM"}]
wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$
```

Error response: 401 Unauthorized

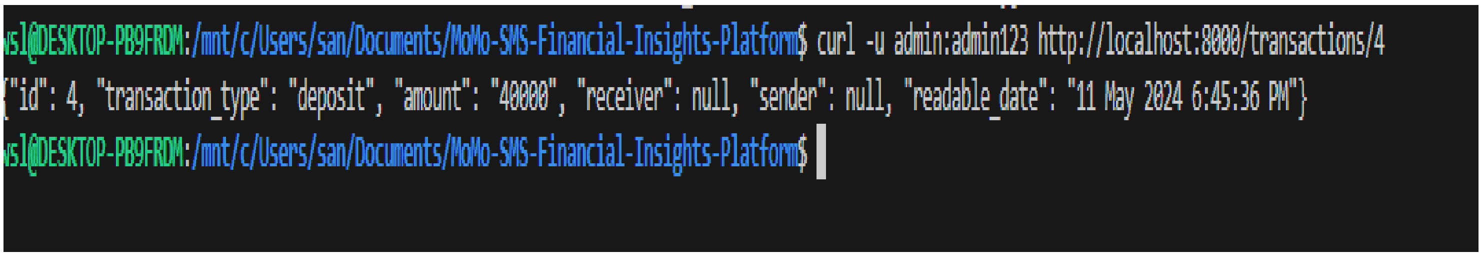


```
wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$ curl http://localhost:8000/transactions/
{"error": "Unauthorized"}
wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$
```

2.2 GET /transactions/{id}

Description:
Returns a single transaction based on its ID.

Successful Response (200 OK):



```
wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$ curl -u admin:admin123 http://localhost:8000/transactions/4
{"id": 4, "transaction_type": "deposit", "amount": "40000", "receiver": null, "sender": null, "readable_date": "11 May 2024 6:45:36 PM"}
wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$
```

2.3 POST /transactions

Description:
Adds a new transaction to the dataset.
Accepted Data Types and Fields

Only the following fields are allowed:

- amount integer
- type string

No extra fields are accepted.

Validation Rules

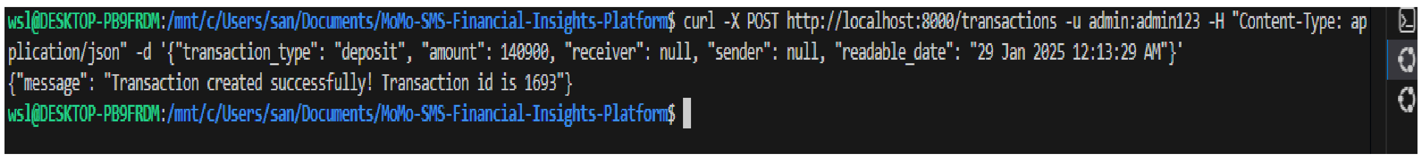
- Request body must be valid JSON.
- amount must be an integer.
- Missing fields result in rejection.
- Extra fields result in rejection.
- IDs supplied by the client are ignored.

Reason for Restrictions
This prevents inconsistent data, schema drift, and accidental overwriting of values generates by the system

Request Example:

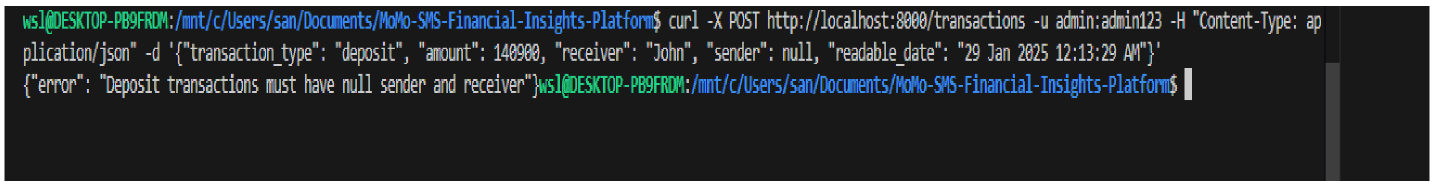
Successful response (201)

(view at 125% zoom to see the screenshots clearly)

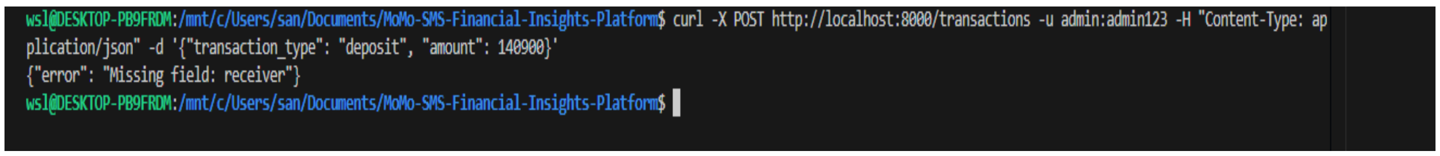


```
wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$ curl -X POST http://localhost:8000/transactions -u admin:admin123 -H "Content-Type: ap
plication/json" -d '{"transaction_type": "deposit", "amount": 140900, "receiver": null, "sender": null, "readable_date": "29 Jan 2025 12:13:29 AM"}'
{"message": "Transaction created successfully! Transaction id is 1693"}
wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$
```

Error Responses:(Due to the validation rules put in place)



```
wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$ curl -X POST http://localhost:8000/transactions -u admin:admin123 -H "Content-Type: ap
plication/json" -d '{"transaction_type": "deposit", "amount": 140900, "receiver": "John", "sender": null, "readable_date": "29 Jan 2025 12:13:29 AM"}'
{"error": "Deposit transactions must have null sender and receiver"}wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$
```



```
wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$ curl -X POST http://localhost:8000/transactions -u admin:admin123 -H "Content-Type: ap
plication/json" -d '{"transaction_type": "deposit", "amount": 140900}'
{"error": "Missing field: receiver"}
wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$
```

2.4 PUT /transactions/{id}

Description:
Updates an existing transaction.

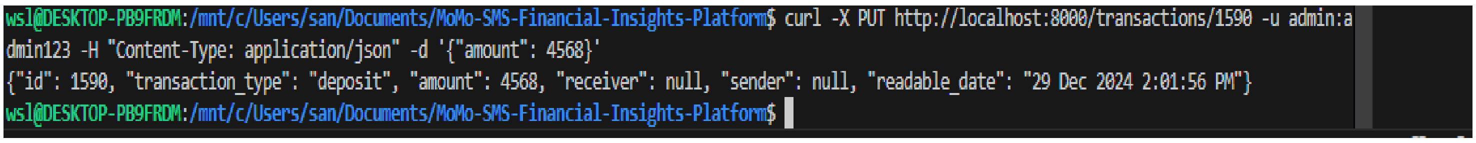
Accepted Fields

- amount integer
- type string

Validation Rules

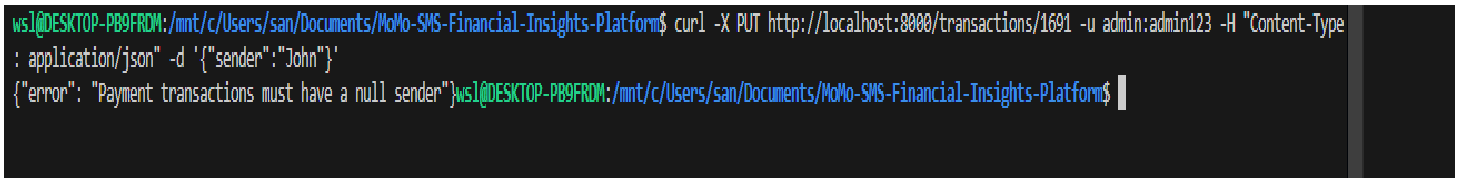
- Only existing transaction IDs can be updated.
- Partial updates are not allowed.
- All required fields must be present.
- Extra fields are rejected.
- Data types must match the schema.

Successful Response (200 OK):



```
wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$ curl -X PUT http://localhost:8000/transactions/1590 -u admin:a
dmin123 -H "Content-Type: application/json" -d '{"amount": 4568}'
{"id": 1590, "transaction_type": "deposit", "amount": 4568, "receiver": null, "sender": null, "readable_date": "29 Dec 2024 2:01:56 PM"}
wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$
```

Error Response:



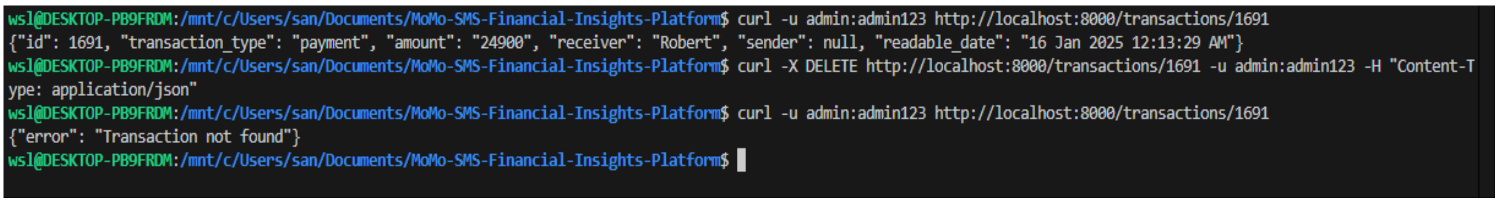
```
wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$ curl -X PUT http://localhost:8000/transactions/1691 -u admin:admin123 -H "Content-Type
: application/json" -d '{"sender":"John"}'
{"error": "Payment transactions must have a null sender"}wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$
```

2.5 DELETE /transactions/{id}

Description:
Deletes a transaction by ID.

Validation Rules
- ID must exist.
- No request body is accepted.

Successful Response - 204 No Content**:**



```
wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$ curl -u admin:admin123 http://localhost:8000/transactions/1691
{"id": 1691, "transaction_type": "payment", "amount": "24900", "receiver": "Robert", "sender": null, "readable_date": "16 Jan 2025 12:13:29 AM"}
wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$ curl -X DELETE http://localhost:8000/transactions/1691 -u admin:admin123 -H "Content-T
ype: application/json"
wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$ curl -u admin:admin123 http://localhost:8000/transactions/1691
{"error": "Transaction not found"}
wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$
```

3. Data Structures & Algorithms (DSA) Comparison

Two approaches were implemented to retrieve transactions by ID:

3.1 Linear Search

- Transactions stored in a list
- Each search scans records one by one
- Time complexity: O(n)

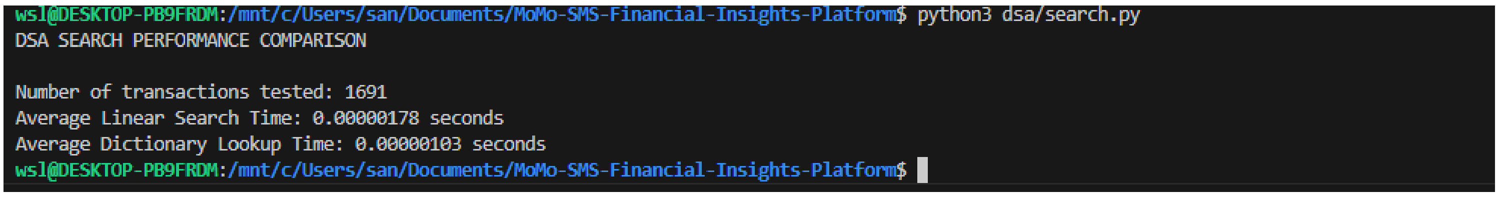
Observation
Performance decreases as the number of transactions grows.

3.2 Dictionary Lookup

- Transactions stored in a dictionary using id as the key
- Direct access to records
- Time complexity: O(1)

Observation
Lookup time remains constant regardless of the number of transactions.

Results (1690+ Records)



```
wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$ python3 dsa/search.py
DSA SEARCH PERFORMANCE COMPARISON

Number of transactions tested: 1691
Average Linear Search Time: 0.00000178 seconds
Average Dictionary Lookup Time: 0.00000103 seconds
wsl@DESKTOP-PB9FRDM:/mnt/c/Users/san/Documents/MoMo-SMS-Financial-Insights-Platform$
```

Conclusion:
Dictionary lookup is significantly faster because it avoids scanning the entire dataset. The hash-based structure allows direct access to records using unique keys.

Suggested Improvement:
Using a database index (e.g., B-tree index in SQL) would further improve performance and scalability for large datasets.

4. Reflection on Basic Authentication Limitations

While Basic Authentication was suitable for this academic project due to its simplicity, it has several weaknesses:

- Credentials are sent with every request
- Base64 encoding is not encryption
- No session expiration or token revocation
- Vulnerable without HTTPS

Stronger Alternatives

- JWT (JSON Web Tokens):
  Stateless, time-limited tokens with better scalability
- OAuth2:
  Secure delegated access without exposing passwords
- API Keys with HTTPS:
  Easier to rotate and revoke than passwords

In a production system handling financial data, Basic Authentication should only be used in combination with HTTPS or replaced entirely by token-based authentication.