MoMo SMS Data Hub – Secure REST API Implementation and DSA Integration

1. Introduction

The mobile money services bring vast amounts of SMS messages to execute transactions like deposits, withdrawals, transfers, and payments. Though these messages are useful, they are not structured making them difficult to analyze or adapt in the modern applications.

The current project uses a secure REST API to make mobile money SMS data available in structured and queryable format. The data set (`modified_sms_v2.xml`) was converted to a JSON format and is being made available through a Python-based Authenticated, CRUD enabled, and Documented REST API.

Data Structures & Algorithms (DSA) are also implemented in the project to compare the efficiency of the search using linear search and dictionary look-up to support the relevance of algorithmic thought in API development.

2. API Implementation

2.1 Endpoints

- GET /transactions — Returns all transactions
- GET /transactions/{id} — Returns one transaction by ID
- POST /transactions — Adds a new transaction
- PUT /transactions/{id} — Updates a transaction
- DELETE /transactions/{id} — Deletes a transaction

2.2 Sample Request & Response

Example GET /transactions request and response:

Request:
GET http://localhost:8000/transactions
Authorization: Basic <base64(admin:password)>

Response:

[{ "id": 1, "type": "DEPOSIT", "amount": 5000, "sender": "0780001111", "receiver": "MOMO_AGENCY", "timestamp": "2023-08-01T12:34:56" }]

2.3 Error Codes

Status	Meaning
400 Bad Request	Invalid JSON or missing required fields
401 Unauthorized	Invalid or missing auth credentials

404 Not Found	Resource or transaction not found
500 Internal Server Error	Unexpected server error

3. Authentication & Security

The API uses Basic Authentication, where the client sends a Base64-encoded username:password in the HTTP Authorization header.

Example:
Authorization: Basic YWRtaW46cGFzc3dvcmQ=

Reflection: Basic Auth is insecure since the credentials will be sent with each request, tokens are not expired, and the authentication cannot be used without HTTPS. The more robust options are JWT, OAuth2 or TLS API keys.

4. Data Structures & Algorithms (DSA)

Two methods were implemented for transaction lookups:

- Linear Search (O(n))
- Dictionary Lookup (O(1))

Benchmark results:

Method	Average Lookup Time (ms)
Linear Search	1.45
Dictionary Lookup	0.02

Reflection: The dictionary look up is much quicker. In the case of large datasets, indexed databases or balanced trees are more appropriate with respect to complex queries.

5. Testing & Validation

Testing was done using curl and Postman. Example commands:

- GET /transactions (valid auth): curl -u admin:password http://localhost:8000/transactions
- Unauthorized GET: curl -u admin:wrong http://localhost:8000/transactions
- POST /transactions: curl -u admin:password -X POST http://localhost:8000/transactions

```
-H "Content-Type: application/json" -d
'{"type":"PAYMENT","amount":2000,"sender":"0781112222","receiver":"SHOP_123","timesta
mp":"2023-08-05T14:00:00"}'
```

6. Conclusion & Reflection

This project successfully:
- Parsed SMS data to structured JSON
- Built a REST API with CRUD functionality
- Secured endpoints with Basic Auth
- Documented API usage
- Compared lookup strategies via DSA

Improvements in the future involve the migration to JWT or OAuth2, the use of a database, the inclusion of TLS, and the addition of the analytical endpoints.