Project Report: MoMo Transaction Management System

1. Introduction to API Security

API security is a critical component of modern software development, especially for financial systems like Mobile Money (MoMo). It involves protecting the integrity, confidentiality, and availability of data as it travels between clients and servers. Without proper security, APIs are vulnerable to unauthorized access, data breaches, and man-in-the-middle attacks.

In this project, we implemented **Basic Authentication** to ensure that only authorized agents can interact with the transaction records. This serves as the first line of defense by requiring a valid username and password for every request.

2. Documentation of Endpoints

The API provides a full suite of CRUD (Create, Read, Update, Delete) operations to manage the 1,693 transactions parsed from the source XML data.

Endpoint	**Method**	**Description**	**Auth Required**
/transactions	GET	Retrieve all active transactions.	Yes
/transactions/<id>	GET	Retrieve a specific transaction by its ID.	Yes
/transactions	POST	Add a new MoMo transaction record.	Yes
/transactions/<id>	PUT	Update fields of an existing transaction.	Yes
/transactions/<id>	DELETE	Perform a soft delete (mark is_deleted = TRUE).	Yes

Example Data Structure

Transactions are modeled based on MoMo SMS structures:

- **transaction_ref**: Unique ID (e.g., 76662021700)
- **amount**: The value transferred (e.g., 2000 RWF)
- **fee**: Service charges (e.g., 100 RWF)

- **status**: Transaction state (Success/Failed).

---

3. Results of DSA Comparison

To optimize performance, we compared two searching algorithms using 20 sample records from the dataset.

Performance Data

- **Linear Search O(n):** Scanned the list sequentially. On average, searching for the last record took significantly longer as the list grew.
- **Dictionary Lookup O(1):** Used a hash map to jump directly to the transaction. The time remained constant regardless of the search target.

Conclusion: For a high-volume system (1,600+ records), the Dictionary Lookup is the superior choice for real-time API performance.

---

4. Reflection on Basic Auth Limitations

While Basic Authentication was successfully implemented to return **401 Unauthorized** for invalid credentials, it has several critical weaknesses:

1. **Credential Encoding:** Basic Auth sends credentials as a Base64 string. This is not encryption; if an attacker intercepts the traffic, they can easily decode the password.
2. **No Expiration:** Unlike tokens (JWT), Basic Auth credentials do not expire, posing a long-term risk if a device is compromised.
3. **Repetitive Transmission:** Sending the password with *every* request increases the "attack surface" or the chances of the password being stolen during transit.

Recommended Upgrade: In a production environment, this system should transition to **OAuth2 or JWT (JSON Web Tokens)** to provide better security through short-lived, revocable access tokens.