MoWise API Report

1. Introduction

MoWise is a REST API that processes mobile money (MoMo) SMS transactions. The API provides endpoints to create, read, update, and delete transaction records. This report documents the API endpoints, compares the performance of two data structure implementations, and discusses authentication security.

API Security Overview

The API uses Basic Authentication to protect endpoints. Authentication requires a username and password sent with each request. While suitable for development, Basic Auth has security limitations that make it inappropriate for production environments handling financial data.

---

2. API Endpoints Documentation

Base URL: `http://127.0.0.1:8000`
Authentication: Basic Auth
Credentials: `admin:admin123`, `user:password`, `guest:guest123`

Available Endpoints

Method	Endpoint	Description
GET	/	Get API information
GET	/transactions	Get all transactions
GET	/transactions/{id}	Get single transaction
POST	/transactions	Create new transaction
PUT	/transactions/{id}	Update transaction
DELETE	/transactions/{id}	Delete transaction

Transaction Structure

{
"id": 3,
"transaction_type": "payment",
"amount": "600",
"sender": "You",
"receiver": "Samuel Carter",
"timestamp": "2024-05-10 21:32:32",
"balance": "400",
"fee": "0",
"date": "10 May 2024 9:32:40 PM",
"raw_body": "TxId: 51732411227. Your payment of 600 RWF to Samuel Carter 95464 has been completed at 2024-05-10 21:32:32. Your new balance: 400 RWF. Fee was 0 RWF.Kanda*182*16# wiyandikishe muri poromosiyo ya BivaMoMotima, ugire amahirwe yo gutsindira ibihembo bishimishije."
}

Testing with Postman

Setting up Authentication:

1. Open Postman and create a new request
2. Go to the **Authorization** tab
3. Select **Basic Auth** from the dropdown
4. Enter Username: `admin` and Password: `admin123`

Get all transactions:

- Method: `GET`
- URL: `http://127.0.0.1:8000/transactions`
- Click **Send**

Create transaction:

- Method: `POST`
- URL: `http://127.0.0.1:8000/transactions`
- Go to **Body** tab, select **raw**, select **JSON**
- Enter:
- {
- "transaction_type": "transfer",
- "amount": "10000",
- "sender": "You",
- "receiver": "Aime"
- }

- Click **Send**

Get by ID:

- Method: `GET`
- URL: `http://127.0.0.1:8000/transactions/1`
- Click **Send**

---

3. Data Structure Comparison

Version 1.0: Linear Search

Storage Format:

- [
- {"id": 1, "amount": "5000"....},
- {"id": 2, "amount": "2000"...}
-]

Algorithm: Loops through the entire list to find matching ID
Time Complexity: O(n) - gets slower as data grows
How it works: Checks each transaction one by one until it finds the right ID

Version 2.0: Dictionary Lookup

Storage Format:

- {
- "1": {"amount": "5000",...},
- "2": {"amount": "2000",...}
- }

Algorithm: Direct access using ID as dictionary key
Time Complexity: O(1) - constant speed regardless of data size
How it works: Uses hash table to jump directly to the transaction

Performance Results

Tested with 1000 transactions, running 1000 lookups:

ID	Linear Search	Dictionary Lookup	Speedup
ID 1	0.001250s	0.000012s	104x faster
ID 2	0.001550s	0.000011s	
ALL IDs	33ms	35ms	Same range

Key Findings:

- Due to working with local projects with a few rows, the difference is minimal, but general tests and the implementation of the two algorithms prove Dictionary Lookup to be incredibly faster than linear search due to the constant O(1) performance makes it essential for production use.

---

4. Basic Authentication Limitations

MoWise currently uses Basic Authentication, which has significant security problems:

How Basic Auth Works

Sends username:password encoded in Base64 with every request:

Why It's Problematic

Not Actually Encrypted: Base64 is encoding, not encryption. Anyone can decode it instantly to see the password.

Password Sent Every Time: Unlike tokens, your actual password travels with every single API request, increasing exposure.

No Expiration: Credentials work forever until manually changed. If compromised, attackers have unlimited access.

Can't Revoke Sessions: No way to invalidate specific sessions - must change password globally.

Requires HTTPS: Without HTTPS, passwords are sent in plaintext over the network.

Better Alternatives

JWT (JSON Web Tokens):

- Password only sent once during login
- Tokens expire automatically
- Can be revoked individually
- Can contain user roles and permissions