zlib compression in HTTP Server (Server)

Description

In this problem, you need to implement an HTTP server that supports zlib compression using Python's built-in libraries. The server should handle GET requests to the /status endpoint and return a JSON response. When the client includes **Accept-Encoding: deflate** in the request headers, the server must compress the response using **zlib** compression.

Your task is to create both:

1. An HTTP server that listens on localhost:8080 and handles compression
2. An HTTP client that can connect to the server and decompress responses

Server Requirements:

- Listen on localhost port 8080
- Accept only one client connection at a time
- Receive up to 4096 bytes from each client
- Handle GET requests to /status endpoint
- Return JSON object: {"name": "myServer", "status": "online"} (use json library)
- Apply zlib compression using zlib.compress when Accept-Encoding: deflate is present in request headers
- Return 404 for any other endpoints
- Close client connection after each request

Input

The programs should not require any command-line input. Both server and client should be implemented as standalone Python scripts that can be executed directly.

Output (without unit test)

Server Output:

Server listening on ('localhost', 8080)

Connection from ('127.0.0.1', 54172)

```
Request header: ['GET /status HTTP/1.1', 'Host: localhost',
'Accept-Encoding: deflate', 'Connection: close', '', '']


Server shutting down...
```

Output (with unit test)

When running the unit tests for both server and client implementations, the expected output should show successful test assertions:

Server Test Output:

- ✅ Server listening on port 8080
- ✅ Server receives 4096 bytes from client
- ✅ Sent correct JSON response without compression
- ✅ HTTP response includes deflate compression header
- ✅ zlib compression and decompression works correctly