CODTECH Internship - Task 4

Code Refactoring and Performance Optimization

Instructions:

1. Take an open-source project and refactor it to improve readability and performance.

2. Deliverable: A report on changes made and their impact on performance.

Below is an example of refactoring and optimization applied to a Python code snippet.

1. Original Code Example (Inefficient):

```
def find_duplicates(arr):
    duplicates = []
    for i in range(len(arr)):
        for j in range(i + 1, len(arr)):
            if arr[i] == arr[j] and arr[i] not in duplicates:
                duplicates.append(arr[i])
    return duplicates

nums = [1, 2, 3, 2, 4, 5, 1]
print(find_duplicates(nums))
```

2. Refactored & Optimized Code:

```
def find_duplicates(arr):
    from collections import Counter
    return [item for item, count in Counter(arr).items() if count > 1]

nums = [1, 2, 3, 2, 4, 5, 1]
print(find_duplicates(nums))
```

3. Refactoring and Optimization Report

Changes Made:

- Replaced nested loops with collections.Counter to count elements efficiently.

- Reduced time complexity from O(n^2) to O(n).

- Improved readability by using list comprehensions.

Impact on Performance:

- For large lists, the refactored code runs significantly faster due to linear-time counting.

- The new approach uses built-in optimized Python libraries.