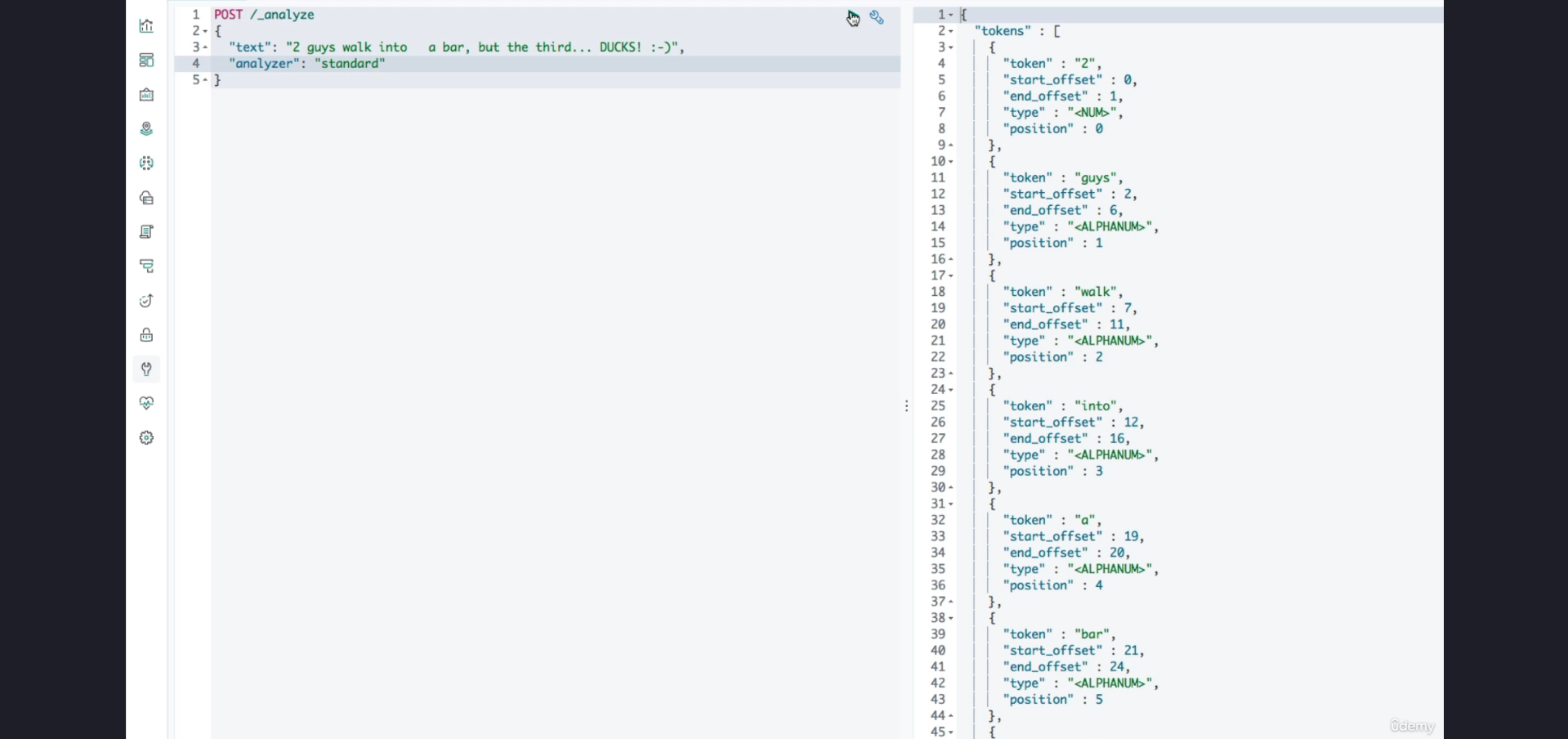
# Using the Analyze API in Elasticsearch

## 1. Introduction

In this lecture, we explore the Analyze API in Elasticsearch, a powerful tool for testing and understanding how strings are analyzed. This API is invaluable for building custom analyzers and experimenting with the analysis process.

## 2. Testing the Standard Analyzer



1. The endpoint for the Analyze API is **'\_analyze'**, and the HTTP verb used is POST.

2. To analyze a string, the 'text' parameter is used to supply the text.

3. Example Query:

POST /\_analyze  
{  
 "analyzer": "standard",  
 "text": "Why did the third duck cross the road? To prove he wasn’t a chicken!"  
}

4. Result Explanation:

- Tokens: The analyzer emits tokens representing words or meaningful units.

- Metadata: Each token includes metadata such as:

- Type: Usually 'alphanumeric', but can vary (e.g., 'numeric').

- Start and End Offsets: Character positions of the token in the original text.

- Example: A number in the text results in a numeric token type.

5. Tokenization Details:

- The standard tokenizer splits text at word boundaries using an advanced algorithm.

- Whitespace and symbols (e.g., commas, periods, exclamation marks) are removed as they add no value for full-text search.

- The 'lowercase' token filter converts tokens to lowercase.

## 3. Replicating the Standard Analyzer

1. Instead of specifying the analyzer directly, its components can be defined explicitly: character filters, tokenizer, and token filters.

This one used above {**"analyzer": "standard"**} and blow one both are same.   
{  
 **"char\_filter": [],  
 "tokenizer": "standard",  
 "filter": ["lowercase"]**  
}

2. Example Query:

POST /\_analyze  
{  
 "char\_filter": [],  
 "tokenizer": "standard",  
 "filter": ["lowercase"],  
 "text": "Why did the third duck cross the road? To prove he wasn’t a chicken!"  
}

3. Explanation of Parameters:

- 'char\_filter': Defines character filters. Left empty as the standard analyzer does not use them.

- 'tokenizer': Specifies the tokenizer, in this case, 'standard'.

- 'filter': Lists token filters, such as 'lowercase'.

4. Result: The output matches the result from the standard analyzer.

## 4. Use Cases for the Analyze API

1. Building Custom Analyzers: Experiment with components to create and test custom analyzers.

2. Understanding Built-in Analyzers: Test built-in analyzers to understand their behavior and results.

3. Debugging and Optimization: Use the API to debug and refine analysis configurations before applying them to documents.

## 5. Summary

The Analyze API is a versatile tool for testing how Elasticsearch processes text. It allows you to experiment with built-in analyzers or define custom analyzer components explicitly.

Understanding how the standard analyzer works lays the foundation for building custom solutions. This API is essential for experimenting with and optimizing text analysis in Elasticsearch.