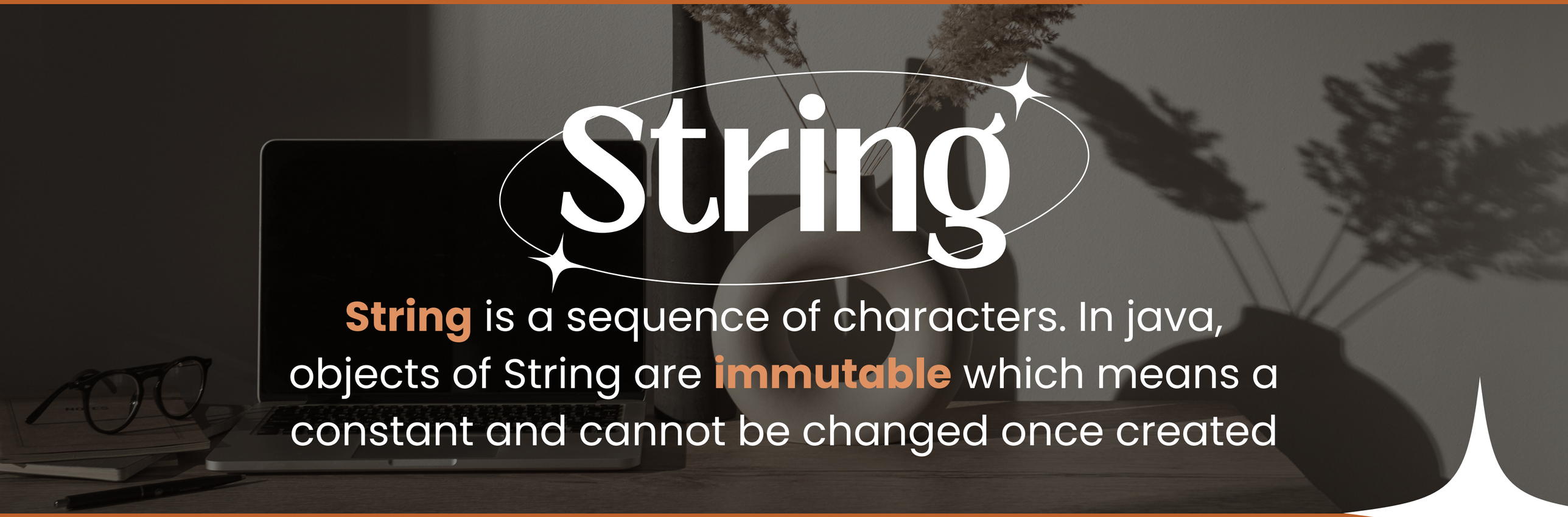
Passerelles Numeriques

String Class

Presentation by **Stephanie Micah**

Java | 2023

Passerelles Numeriques



String

String is a sequence of characters. In java, objects of String are **immutable** which means a constant and cannot be changed once created

Java | 2023

Creating a String

String literal

Using new keyword

String literal

String s = “HelloWorld”;

Using new keyword

String s = new String (“HelloWorld”);

String Methods

- length()
- concat()
- substring()
- replace()
- toUpperCase()
- toLowerCase()
- equals()
- equalsIgnoreCase()
- indexOf()
- lastIndexOf()
- split()

Java | 2023

length()

The length() method is used to find the length (number of characters) of a string.

```
String message = "Hello, Java!";
int length = message.length();
System.out.println("The length of the string is: " + length);
```

```
The length of the string is: 12
```

concat()

The concat() method is used to concatenate (combine) two strings.

```
String firstName = "John";
String lastName = "Doe";
String fullName = firstName.concat(" ").concat(lastName);
System.out.println("Full Name: " + fullName);
```

```
Full Name: John Doe
```

substring()

The substring() method is used to extract a substring from a given string.

```
String originalString = "Hello, World!";
String substring1 = originalString.substring(7);
String substring2 = originalString.substring(0, 5);

System.out.println("Substring 1: " + substring1);
System.out.println("Substring 2: " + substring2);
```

substring()

The substring() method is used to extract a substring from a given string.

```
Substring 1: World!
Substring 2: Hello
```

replace()

The replace() method is used to replace characters or substrings within a string.

```
String originalString = "Java is awesome!";
String replacedString = originalString.replace("awesome", "amazing");

System.out.println("Original String: " + originalString);
System.out.println("Replaced String: " + replacedString);
```

replace()

The replace() method is used to replace characters or substrings within a string.

```
Original String: Java is awesome!
Replaced String: Java is amazing!
```

Conversion & Comparison

toLowerCase() equals()

toUpperCase() equalsIgnoreCase()

toLowerCase() & toUpperCase()

Convert the case of characters in a string.

```
String message = "Hello, Java!";
String lowerCaseMessage = message.toLowerCase();
String upperCaseMessage = message.toUpperCase();


System.out.println("Original Message: " + message);
System.out.println("Lowercase Message: " + lowerCaseMessage);
System.out.println("Uppercase Message: " + upperCaseMessage);
```

toLowerCase() & toUpperCase()

Convert the case of characters in a string.

```
Original Message: Hello, Java!
Lowercase Message: hello, java!
Uppercase Message: HELLO, JAVA!
```

equals()
&
equalsIgnoreCase()

The equals() method is used to check if two strings are equal, while equalsIgnoreCase() ignores the case during the comparison.

equals()
&
equalsIgnoreCase()

```java
String str1 = "Java";
String str2 = "java";
String str3 = "Java";

boolean isEqual1 = str1.equals(str2);
boolean isEqual2 = str1.equals(str3);
boolean isEqualIgnoreCase = str1.equalsIgnoreCase(str2);

System.out.println("str1 equals str2: " + isEqual1);
System.out.println("str1 equals str3: " + isEqual2);
System.out.println("str1 equalsIgnoreCase str2: " + isEqualIgnoreCase);
```

equals()
&
equalsIgnoreCase()

```
str1 equals str2: false
str1 equals str3: true
str1 equalsIgnoreCase str2: true
```

Searching & Splitting

indexOf()

lastIndexOf()

split()

indexOf() & lastIndexOf

The indexOf() method is used to find the index of a specified character or substring within a string. The lastIndexOf() method is used to find the last occurrence of a specified character or substring within a string.

indexOf() & lastIndexOf

```java
String text = "Hello, Java Programming!";
int indexOfJava = text.indexOf("Java");
int lastIndexOfo = text.lastIndexOf("o");


System.out.println("Text: " + text);
System.out.println("Index of 'Java': " + indexOfJava);
System.out.println("Last Index of 'o': " + lastIndexOfo);
```

indexOf() & lastIndexOf

```
Text: Hello, Java Programming!
Index of 'Java': 7
Last Index of 'o': 18
```

split()

```
String names = "Alice,Bob,Carol,Dave";
String[] nameArray = names.split(",");

System.out.println("Names: " + names);
System.out.print("Separated Names: ");
for (String name : nameArray) {
    System.out.print(name + " ");
}
```

The split() method is used to split a string into an array of substrings based on a delimiter.

split()

The split() method is used to split a string into an array of substrings based on a delimiter.

```
Names: Alice,Bob,Carol,Dave
Separated Names: Alice Bob Carol Dave
```

Immutability & Memory Optimization

Strings in Java are immutable, meaning once created, their values cannot be changed. This immutability has memory optimization benefits and impacts how strings are manipulated.

Immutability

Once a string is created, its contents cannot be changed. Any operation that appears to modify a string actually creates a new string.

Memory Optimization

Strings in Java are stored in the string pool, which helps to conserve memory by reusing strings with the same value.

```java
String str1 = "Hello";
String str2 = str1.concat(", Java!");


System.out.println("str1: " + str1);
System.out.println("str2: " + str2);
System.out.println("str1 after concat: " + str1);
```

```
str1: Hello
str2: Hello, Java!
str1 after concat: Hello
```

The memory optimization benefits of the string pool allow the JVM to reuse existing strings with the same value, which helps conserve memory in Java applications.

StringBuilder & StringBuffer

When you need to modify strings frequently, the StringBuilder and StringBuffer classes provide more efficient ways to handle string manipulation compared to the standard String class.

StringBuilder

It is a mutable sequence of characters that can be modified without creating new instances. It is not thread-safe, suitable for single-threaded environments.

StringBuffer

Similar to StringBuilder, but it is thread-safe, making it suitable for multi-threaded environments.

StringBuilder

```
StringBuilder sb = new StringBuilder();
sb.append("Java");
sb.append(" is");
sb.append(" awesome!");

System.out.println(sb.toString());
```

StringBuffer

```java
StringBuffer stringBuffer = new StringBuffer("Hello");

// Task 1: Append " Java" to the existing string
stringBuffer.append(" Java");

// Task 2: Insert " World" at index 5
stringBuffer.insert(5, " World");

// Task 3: Reverse the string
stringBuffer.reverse();

System.out.println("Modified StringBuffer: " + stringBuffer.toString());
```

Instructions

1. Create a Java program that takes a user-input *string* and performs the following tasks:
    a. *Calculate* and *display* the *length* of the input string.
    b . Check and display if the string is empty.
    c. Check and display if the string contains the word "Java".
    d. Convert the input string to uppercase and display the result.
    e. Replace all occurrences of the letter 'e' with 'o' and display the modified string.
    f. Extract and display a substring from the input string.
    g. Concatenate the input string with another string and display the final result.
    h. Create a StringBuilder and append the input string to it, then display the result.
    i. Create a StringBuffer and append the input string to it, then display the result.
2. Use the Scanner class to read the user input.
3. Use the String class methods for tasks a to g, and the StringBuilder and StringBuffer classes for tasks h and i.

Instructions

1. Create a Java program that takes a user-input *string* and performs the following tasks:

   a. *Calculate* and *display* the *length* of the input string.

   b . Check and display if the string is empty.

   c. Check and display if the string contains the word "Java".

   d. Convert the input string to uppercase and display the result.

   e. Replace all occurrences of the letter 'e' with 'o' and display the modified string.

Instructions

f. Extract and display a substring from the input string.

g. Concatenate the input string with another string and display the final result.

h. Create a StringBuilder and append the input string to it, then display the result.

i. Create a StringBuffer and append the input string to it, then display the result

2. Use the Scanner class to read the user input.

3. Use the String class methods for tasks a to g, and the StringBuilder and StringBuffer classes for tasks h and i.

Expected Output

```
Length of the string: 24
Is the string empty? false
Contains 'Java': true
Uppercase string: HELLO, JAVA IS AWESOME!
Modified string: Hollo, Java is awosomo!
Substring: Java
Concatenated string: Hello, Java is awesome! - Welcome!
StringBuilder result: Hello, Java is awesome! - StringBuilder
StringBuffer result: Hello, Java is awesome! - StringBuffer
```

Passerelles Numeriques

Thank You!

Presentation by **Stephanie Micah**

Java | 2023

Presentation by **Juliana Silva**

Literary Review

01

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus sed vestibulum nunc, eget aliquam.

02

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus sed vestibulum nunc, eget aliquam.

03

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus sed vestibulum nunc, eget aliquam.

04

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus sed vestibulum nunc, eget aliquam.

Theoretical Framework

Overview

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus sed vestibulum nunc, eget aliquam felis.

Proponents

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus sed vestibulum nunc, eget aliquam felis.

Methodology

01 Quantitative Method

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus sed vestibulum nunc, eget aliquam felis.

02 Quantitative Method

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus sed vestibulum nunc, eget aliquam felis.

Presentation by **Juliana Silva**

Implementation

Phase 1

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus sed vestibulum nunc, eget aliquam felis.

Phase 2

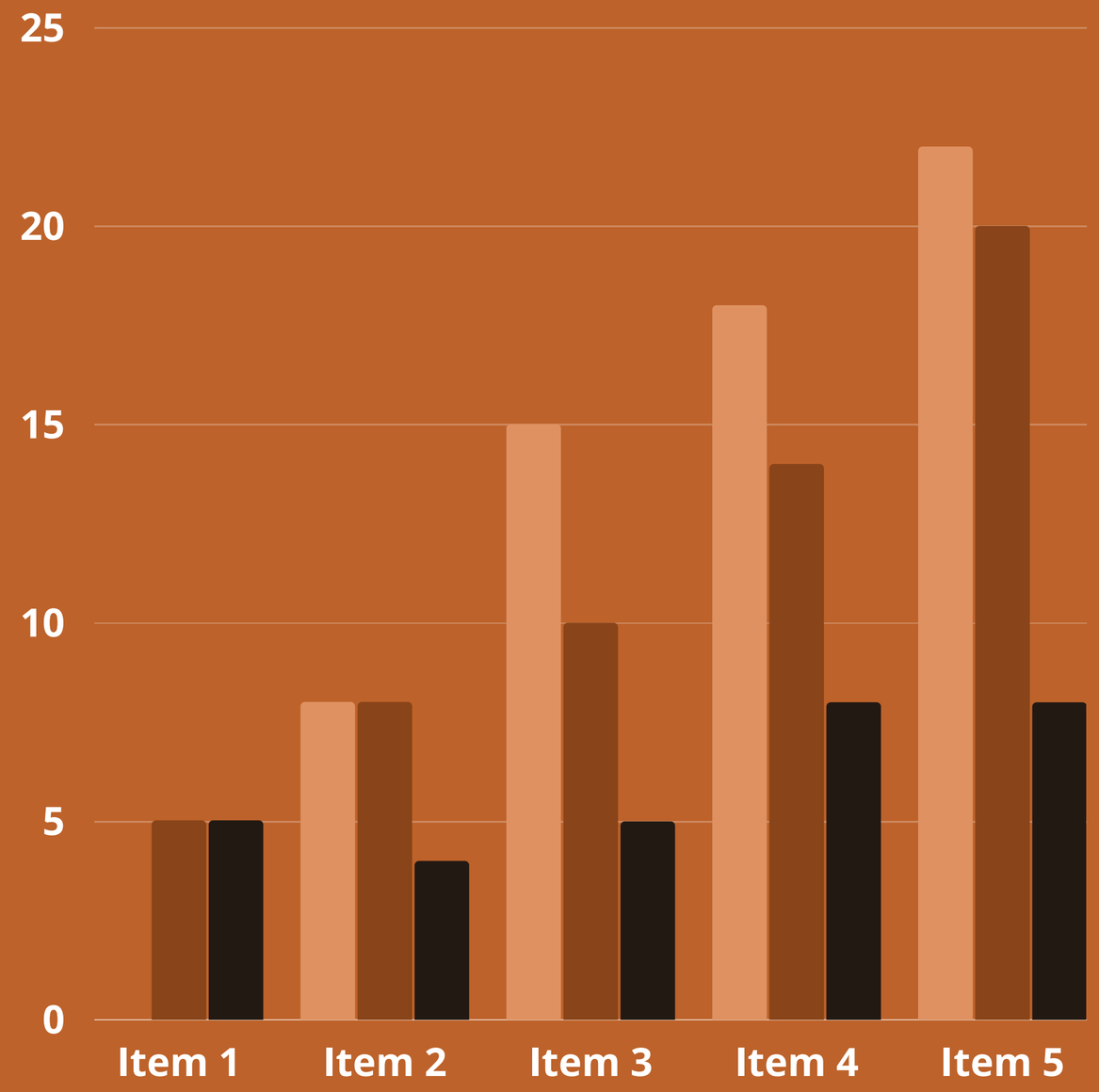
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus sed vestibulum nunc, eget aliquam felis.

Phase 3

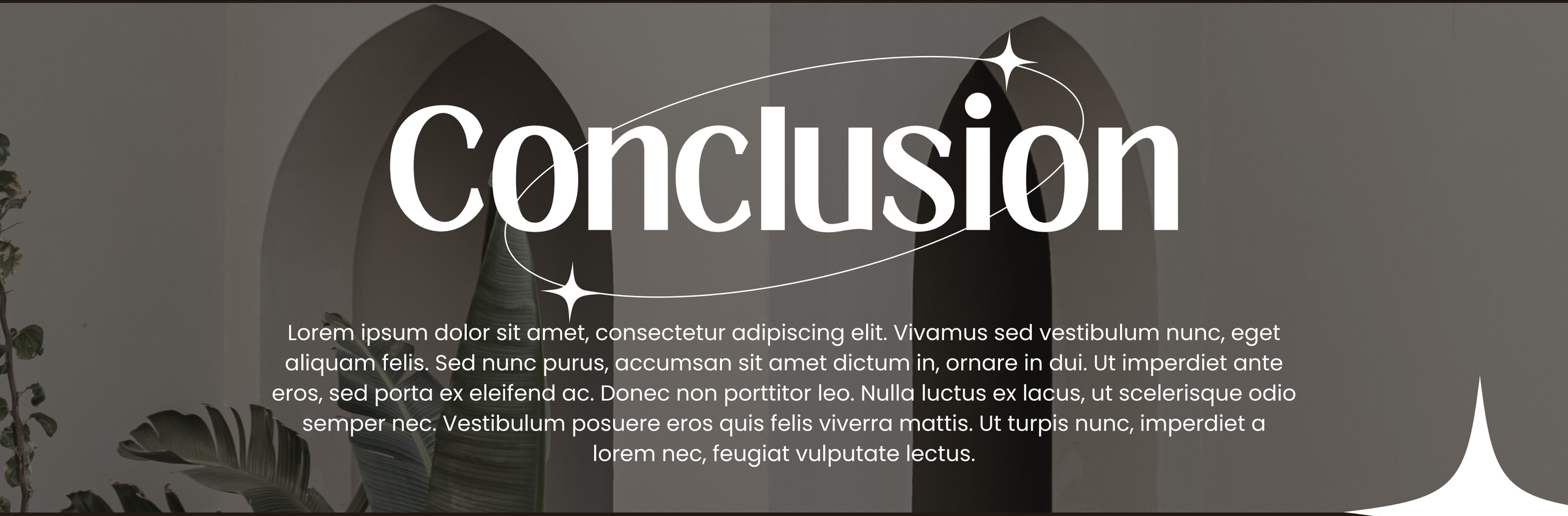
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus sed vestibulum nunc, eget aliquam felis.

Result

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus sed vestibulum nunc, eget aliquam felis. Sed nunc purus, accumsan sit amet dictum in, ornare in dui. Ut imperdiet ante eros, sed porta ex eleifend ac. Donec non porttitor leo. Nulla luctus ex lacus, ut scelerisque odio semper nec. Vestibulum posuere eros quis felis viverra mattis. Ut turpis nunc, imperdiet a lorem nec, feugiat vulputate lectus.



Presentation by **Juliana Silva**



Conclusion

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus sed vestibulum nunc, eget aliquam felis. Sed nunc purus, accumsan sit amet dictum in, ornare in dui. Ut imperdiet ante eros, sed porta ex eleifend ac. Donec non porttitor leo. Nulla luctus ex lacus, ut scelerisque odio semper nec. Vestibulum posuere eros quis felis viverra mattis. Ut turpis nunc, imperdiet a lorem nec, feugiat vulputate lectus.