# System Requirements Specification Index

For

# String Indexing and Concatenation

Version 1.0

IIHT Pvt. Ltd.
fullstack@iiht.com

# Table of Contents

# USE CASE DESCRIPTION

## System Requirements Specification

## 1 Project Abstract

You will be required to demonstrate the use of string indexing techniques and various string concatenation methods like:

- String Indexing: Accessing characters from specific positions and finding positions of substrings.
- String Concatenation: Combining multiple strings using different techniques.

## 2 Assessment Tasks

**Task 1:**

1. **Declare a variable:**
   - A variable named `str` of `String` datatype, initialized with the value `"Hello, Java World!"`.
2. **Perform String Indexing Operations:**
   Use the string variable `str` to perform the following indexing operations:

   - **Character at Index (`charAt()`)**:
     1) Retrieve the character at index `7` from `str` using `charAt(7)`.
     2) Store the result in a variable named `charAt` of `char` datatype.
   - **Find Index of Substring (`indexOf()`)**:
     1) Find the starting index of `"Java"` in `str` using `indexOf("Java")`.
     2) Store the result in a variable named `indexOfJava` of `integer` datatype.
   - **Find Last Index of Character (`lastIndexOf()`)**:
     1) Find the last occurrence of the character `'l'` in `str` using `lastIndexOf("l")`.
     2) Store the result in a variable named `lastIndexOfL` of `integer` datatype.
   - **Extract Substring (`substring()`)**:
     1) Extract a substring from index `7` to `11` using `substring(7, 11)`.
     2) Store the result in a variable named `subStr` of `String` datatype.

   **Print the Results:**
   - Print the results of each string indexing operation i.e, `charAt`, `indexOfJava`, `lastIndexOfL` and `subStr` with appropriate labels in separate lines as shown in the expected output.

## Task 2:

3. **Declare a new string variable:**
   - A variable named `str1` of `String` datatype, initialized with the value `"Hello"`.
   - A variable named `str2` of `String` datatype, initialized with the value `"Java"`.
4. **Perform String Concatenation Operations:**
   Use the string variables `str1` and `str2` of type String with values “Hello” and “Java” respectively to perform the following concatenation operations:

   - **Using `concat()` Method**:
     1) Concatenate `str1` and `str2` using the `concat()` method like concat(" " + str2).
     2) Store the result in a variable named `concatStr` of `String` datatype.
   - **Using + Operator**:
     1) Concatenate `str1` and `str2` using the + operator like str1 + “ “ + str2.
     2) Store the result in a variable named `combined` of `String` datatype.
   - **Using `StringBuilder`**:
     1) Use a `StringBuilder` object named `sb` to append `str1` and `str2` efficiently.
     2) Use the `append` method of `StringBuilder` to add `str1`, followed by a space `" "`, and then append `str2`.
     3) Convert the `StringBuilder` object to a `String` using the `toString` method and store the result in a variable named `builderStr` of `String` datatype.
5. **Print the Results:**
   - Print the results of each string concatenation operation i.e, `concatStr`, `combined`, and `builderStr` with appropriate labels in separate lines as shown in the expected output.

## Expected Output:

```
Character at index 7: J
Index of 'Java': 7
Last index of 'l': 15
Substring from index 7 to 11: Java
Concatenated with concat: Hello Java
Concatenated with + operator: Hello Java
Concatenated with StringBuilder: Hello Java
```

# 3 Template Code Structure

### 3.1 Package: com.yaksha.assignment.StringIndexingConcatenationAssignment

**Resources**

| Class/Interface | Description | Status |
|---|---|---|
| **StringIndexingConcatenationAssignment (class)** | • Main class demonstrating string indexing operations like: `charAt`, `indexOf`, `lastIndexOf`, `substring`.<br>• And string concatenation operations using: `concat`, `+` operator, and `StringBuilder`. | Need to be implemented. |

# 4 Execution Steps to Follow

1. **All actions like build, compile, running application, running test cases will be through Command Terminal.**
2. **To open the command terminal the test takers, need to go to Application menu (Three horizontal lines at left top) Terminal New Terminal.**
3. **This editor Auto Saves the code.**
4. **If you want to exit(logout) and continue the coding later anytime (using Save & Exit option on Assessment Landing Page) then you need to use CTRL+Shift+B-command compulsorily on code IDE. This will push or save the updated contents in the internal git/repository. Else the code will not be available in the next login.**
5. **These are time bound assessments the timer would stop if you logout and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.**
6. **To run your project use command:**
   **mvn compile exec:java**
   **-Dexec.mainClass="com.yaksha.assignment.StringIndexingConcatenationAssignment"**
7. **To test your project test cases, use the command**
   **mvn test**

8. **You need to use CTRL+Shift+B - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.**

9. **You need to use CTRL+Shift+B - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.**