HashAgile Technologies Interview - First Round Program Submission

Question: Longest Palindromic Substring

Write a program to find the longest palindromic substring in a given string without using any built-in substring or reverse functions. For example, for input "babad", the output should be "bab" or "aba".

Code :

Shell Scripting Language (Bash) :

```
palindrome() {
    local str="$1"
    local length=${#str}
    local start=0
    local end=$((length - 1))
    while [ $start -lt $end ]; do
        if [ "${str:start:1}" != "${str:end:1}" ]; then
            return 1
        fi
        start=$((start + 1))
        end=$((end - 1))
    done
    return 0
}
find_longest_palindromic_substring() {
    local input="$1"
    local input_length=${#input}
    local longest_palindrome=""
    for ((i = 0; i < input_length; i++)); do
        for ((j = i + 1; j <= input_length; j++)); do
            substring="${input:i:j-i}"
            if palindrome "$substring"; then
                if [ ${#substring} -gt ${#longest_palindrome} ]; then
                    longest_palindrome="$substring"
                fi
            fi
        done
    done
    echo "Longest Palindromic Substring: $longest_palindrome"
}
echo "Please enter the input string:"
read input_string
if [ -z "$input_string" ]; then
    echo "Input cannot be empty. Please provide a valid string."
    exit 1
fi
find_longest_palindromic_substring "$input_string"
```

Input:

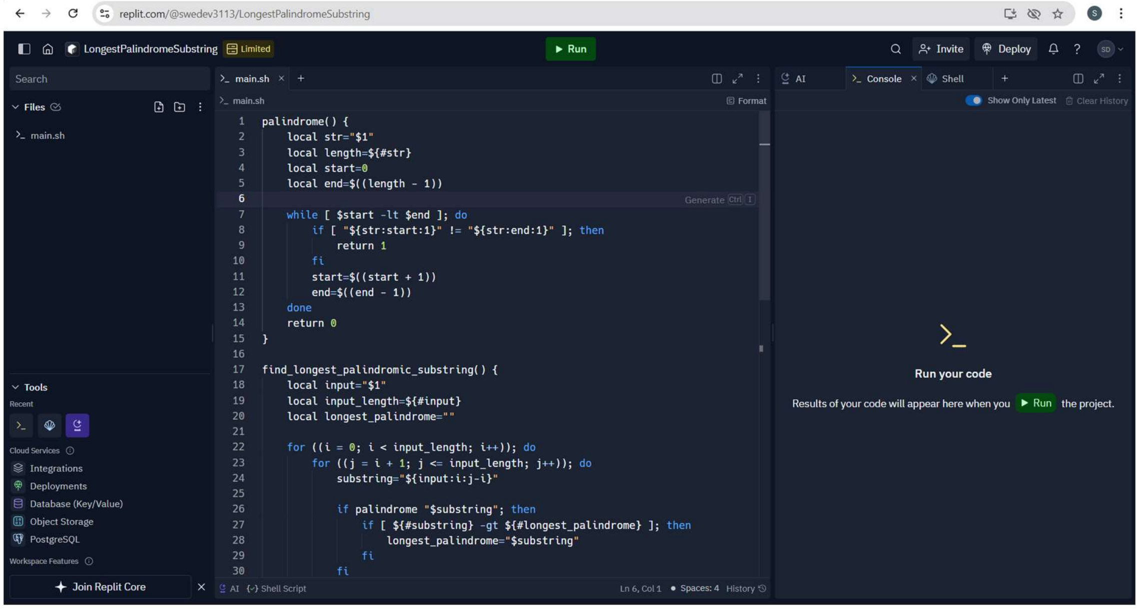
```
Please enter the input string:
babad
```

Output:

```
Longest Palindromic Substring: bab
```

ScreenShots:

Before Running :

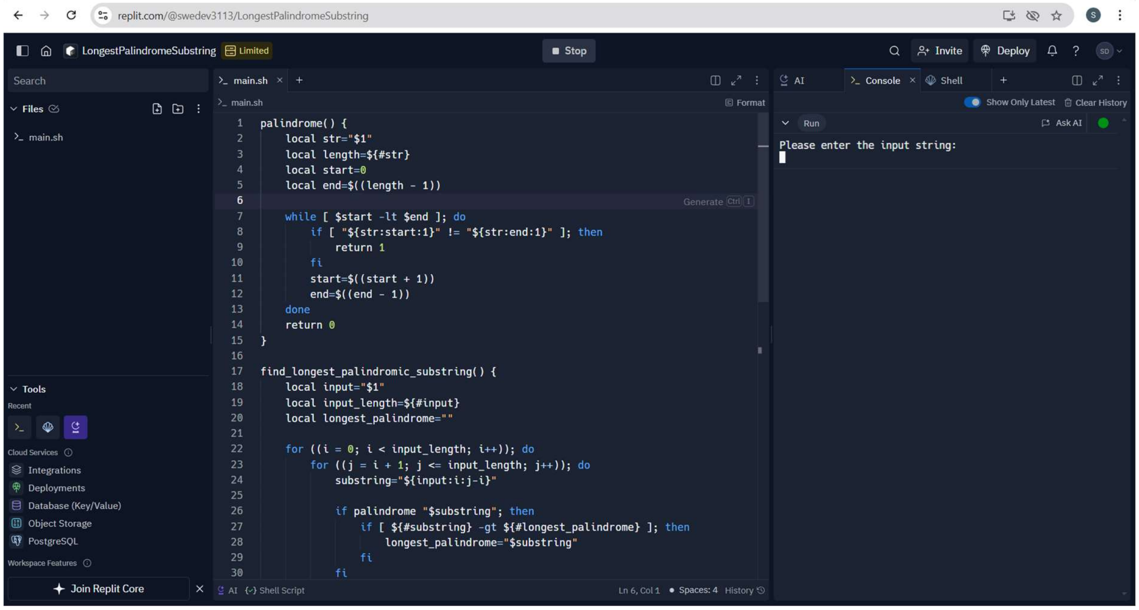


```
palindrome() {
    local str="$1"
    local length=${#str}
    local start=0
    local end=$((length - 1))

    while [ $start -lt $end ]; do
        if [ "${str:start:1}" != "${str:end:1}" ]; then
            return 1
        fi
        start=$((start + 1))
        end=$((end - 1))
    done
    return 0
}

find_longest_palindromic_substring() {
    local input="$1"
    local input_length=${#input}
    local longest_palindrome=""

    for ((i = 0; i < input_length; i++)); do
        for ((j = i + 1; j <= input_length; j++)); do
            substring="${input:i:j-i}"

            if palindrome "$substring"; then
                if [ ${#substring} -gt ${#longest_palindrome} ]; then
                    longest_palindrome="$substring"
                fi
            fi
```

Run your code

Results of your code will appear here when you ▶ Run the project.

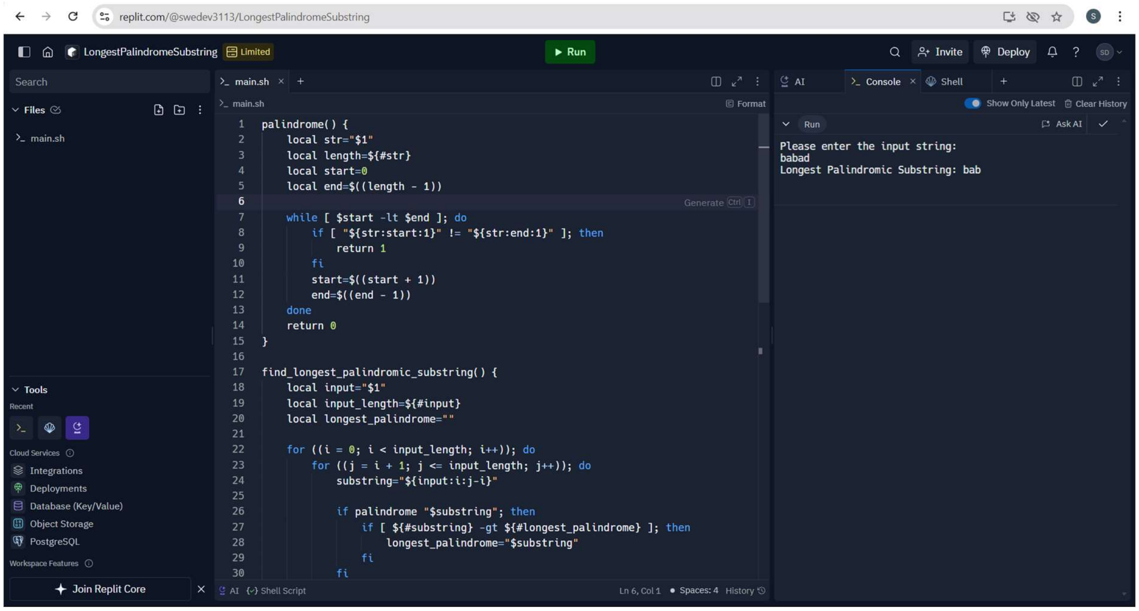
After Running and Waiting for the user to provide the input:



```
palindrome() {
    local str="$1"
    local length=${#str}
    local start=0
    local end=$((length - 1))

    while [ $start -lt $end ]; do
        if [ "${str:start:1}" != "${str:end:1}" ]; then
            return 1
        fi
        start=$((start + 1))
        end=$((end - 1))
    done
    return 0
}

find_longest_palindromic_substring() {
    local input="$1"
    local input_length=${#input}
    local longest_palindrome=""

    for ((i = 0; i < input_length; i++)); do
        for ((j = i + 1; j <= input_length; j++)); do
            substring="${input:i:j-i}"

            if palindrome "$substring"; then
                if [ ${#substring} -gt ${#longest_palindrome} ]; then
                    longest_palindrome="$substring"
                fi
            fi
```

```
Please enter the input string:
```

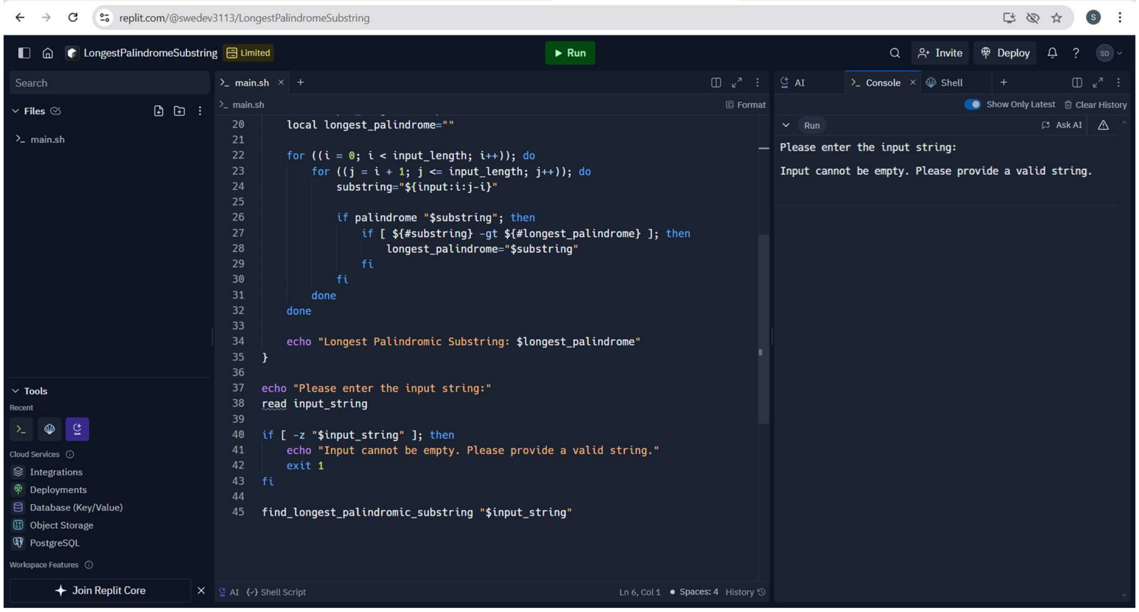
After providing the valid input and the output will be:



```
palindrome() {
    local str="$1"
    local length=${#str}
    local start=0
    local end=$((length - 1))

    while [ $start -lt $end ]; do
        if [ "${str:start:1}" != "${str:end:1}" ]; then
            return 1
        fi
        start=$((start + 1))
        end=$((end - 1))
    done
    return 0
}

find_longest_palindromic_substring() {
    local input="$1"
    local input_length=${#input}
    local longest_palindrome=""

    for ((i = 0; i < input_length; i++)); do
        for ((j = i + 1; j <= input_length; j++)); do
            substring="${input:i:j-i}"

            if palindrome "$substring"; then
                if [ ${#substring} -gt ${#longest_palindrome} ]; then
                    longest_palindrome="$substring"
                fi
            fi
```

```
Please enter the input string:
babad
Longest Palindromic Substring: bab
```

If the user’s input is empty,



```
    local longest_palindrome=""

    for ((i = 0; i < input_length; i++)); do
        for ((j = i + 1; j <= input_length; j++)); do
            substring="${input:i:j-i}"

            if palindrome "$substring"; then
                if [ ${#substring} -gt ${#longest_palindrome} ]; then
                    longest_palindrome="$substring"
                fi
            fi
        done
    done

    echo "Longest Palindromic Substring: $longest_palindrome"
}

echo "Please enter the input string:"
read input_string

if [ -z "$input_string" ]; then
    echo "Input cannot be empty. Please provide a valid string."
    exit 1
fi

find_longest_palindromic_substring "$input_string"
```

```
Please enter the input string:

Input cannot be empty. Please provide a valid string.
```

Java Code:

```java
import java.util.Scanner;
public class Main {
    public static boolean Palindrome(String str, int start, int end) {
        while (start < end) {
            if (str.charAt(start) != str.charAt(end)) {
                return false;
            }
            start++;
            end--;
        }
        return true;
    }

    public static String longestPalindromeSubstring(String s) {
        int max = 0;
        String longestPalindromeSubstring = "";
        for (int i = 0; i < s.length(); i++) {
            for (int j = i; j < s.length(); j++) {
                if (Palindrome(s, i, j)) {
                    int current = j - i + 1;
                    if(current > max) {
                        max = current;
                        longestPalindromeSubstring = s.substring(i, j + 1);
                    }
                }
            }
        }
        return longestPalindromeSubstring;
    }
    public static void main(String[] args) {
        Scanner sc=new Scanner(System.in);
        String input = sc.next();
        System.out.println(longestPalindromeSubstring(input));
    }
}
```

Thank You!