Brute force solution: -

TC->O(n^2), SC->O(1)

```
public class IsUnique {
    public static boolean isUnique(String input) {
        char[] charArray = input.toCharArray();
        for (int i = 0; i < charArray.length; i++) {
            for (int j = i + 1; j < charArray.length; j++) {
                if (charArray[j] == charArray[i]) {
                    return false;
                }
            }
        }
        return true;
    }
    public static void main(String[] args) {
        String input = "Helo Hai";
        System.out.println(input+"  "+isUnique(input));
    }
}
```

Efficient Solution: -

TC->O(n), SC->O(1)

```
public class IsUniqueEfficient {
    public static boolean isUnique(String input) {
        boolean[] chars = new boolean[256];
        char[] charArray = input.toCharArray();
        for (int i = 0; i < charArray.length; i++) {
            int value = charArray[i];
            if (chars[value]) {
                return false;
            }
            chars[value] = true;
        }
        return true;
    }
    public static void main(String[] args) {
        String input = "Helo hai";
        System.out.println(input+"  "+isUnique(input));
    }
}
```