Smallest distinct window

Given a string '**s**'. The task is to find the **smallest** window length that contains all the characters of the given string at least one time.
For eg. A = “**aabcbcdbca**”, then the result would be 4 as of the smallest window will be “**dbca**”.

Example 1:

```
Input : "AABBBCBBAC"
Output : 3
Explanation : Sub-string -> "BAC"
```

Example 2:

```
Input : "aaab"
Output : 2
Explanation : Sub-string -> "ab"
```

Example 3:

```
Input : "GEEKSGEEKSFOR"
Output : 8
Explanation : Sub-string -> "GEEKSFOR"
```

Your Task:
You don't need to read input or print anything. Your task is to complete the function **findSubString()** which takes the string **S** as inputs and returns the length of the smallest such string.

Expected Time Complexity: O(256.N)
Expected Auxiliary Space: O(256)

```
def findSubString(self, s):
        # Your code goes here
        target = set(s)
        ans = 0
        i = -1
        j = -1
        freq = {}
        while True:
            f1, f2 = False, False
```

```
        while i < len(s) - 1 and len(freq) < len(target):
            i = i + 1
            ch = s[i]
            freq[ch]=freq.get(ch, 0) + 1
            f1 = True

        while j < i and len(freq) == len(target):
            pAns = i-j
            if ans == 0 or pAns < ans:
                ans = pAns
            j = j+1
            ch = s[j]
            if freq.get(ch) == 1:
                del freq[ch]
            else:
                freq[ch] = freq[ch] - 1
            f2 = True

        if f1 is False and f2 is False:
            break
    return ans
```