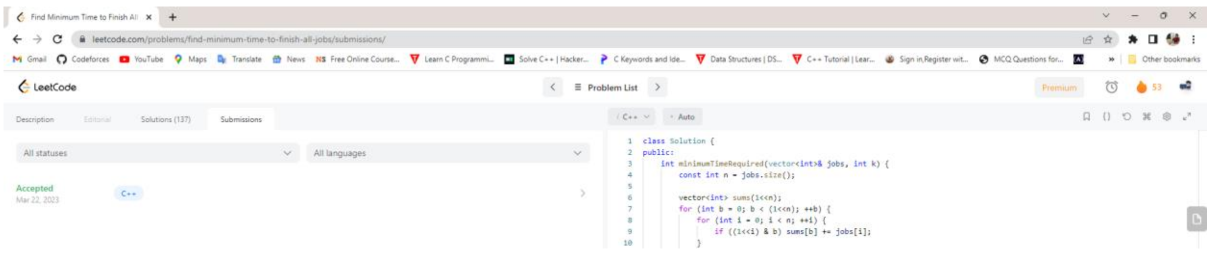
Code-

```
class Solution {
public:
    int minimumTimeRequired(vector<int>& jobs, int k) {
        const int n = jobs.size();

        vector<int> sums(1<<n);
        for (int b = 0; b < (1<<n); ++b) {
            for (int i = 0; i < n; ++i) {
                if ((1<<i) & b) sums[b] += jobs[i];
            }
        }

        vector<vector<int>> dp(k+1, vector<int>(1<<n));
        for (int b = 0; b < (1<<n); ++b) dp[1][b] = sums[b];
        for (int i = 2; i <= k; ++i) {
            for (int b = 1; b < (1<<n); ++b) {
                dp[i][b] = dp[i-1][b];
                for (int tb = b; tb; tb = (tb-1)&b) {
                    dp[i][b] = min(dp[i][b], max(sums[tb], dp[i-1][b-tb]));
                }
            }
        }
        return dp[k][(1<<n)-1];
    }
};
```

Time complexity – O(3^n*n)
Space complexity – O(N)



leetcode.com/problems/find-minimum-time-to-finish-all-jobs/submissions/

Accepted
Mar 22, 2023
C++

```
class Solution {
public:
    int minimumTimeRequired(vector<int>& jobs, int k) {
        const int n = jobs.size();

        vector<int> sums(1<<n);
        for (int b = 0; b < (1<<n); ++b) {
            for (int i = 0; i < n; ++i) {
                if ((1<<i) & b) sums[b] += jobs[i];
            }
```