\*\*\*\*\*\*\*\*\*\*\*\*Tech stack\*\*\*\*\*\*\*\*\*\*\*\*

.Net Core 3

Angular 11

node 12

npm 8

\*\*\*\*\*\*\*\*\*\*\*\*\*IDE Tool\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

VS 2019

VS Code

\*\*\*\*\*\*\*\*\*\*\*INSTRUCTION\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1- Open CandidateMatcher.UI in VS code

2- Run 'npm install' in console

3- Run 'npm run start' in console

4- Open JobAdder.Microservices in VS 2019

5- Build Solution

6- Run CandidateMatcher.API

\*\*\*\*\*\*\*\*\*\*\*Algorithm Used\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1- First we consider a weight for each skill in job skill set from highest to lowest. Weight simply starts from the length of skill set to 1

Job: S1, S2,S3, S4, S5, S6: 6, 5, 4, 3, 2, 1

2- Then, we wight each candidate skill in their skill set. The weight starts from the maximum number of skills a candidate has among all candidates

candiateA: S1, S2 => 5, 4

candiateB: S1, S2, S3, S4, S5 => 5, 4, 3, 2, 1

candiateC: S1, S2, S3, S4 => 5, 4, 3, 2

candiateD: S1, => 5

Please Note: we cannot weight candiate skills simply based on the index, because the candidate with higher number of skills will always have a better chance to win

3- Multiply the skill wieght for each candidate by job skill weight and add them up. The candidate with highest number is the most suitable canidate for the job

CandidateA Score: (5\*6)+(4\*5) = 50

CandidateB Score = (5\*6) + (4\*5) + (3\*4) + (2\*3) + (1\*2) = 70

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Test results\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Graphical user interface, application, Word

Description automatically generated