DOCUMENTAZIONE
CARRER AI MAPS

INFORMAZIONI

Corso: Sistemi Intelligenti

Anno Accademico: 2024/2025

Nome progetto: *Carrer AI Maps*

Membri:

- Florin Bran
- Fiordj Ferro

AI classica: *Search - A**

Goal: *Trovare le soluzioni migliori lavorative nel mondo, con il miglior margine di profitto per la posizione lavorativa e per il singolo individuo.*

COME FUNZIONA - LATO UTENTE

Goal utente: Trovare un lavoro

l'utente apre la nostra applicazione e gli viene fornita una mappa dove può mettere la posizione in cui abita. Quando fa click sulla mappa vengono mostrate anche le coordinate in latitudine e longitudine.

Una volta messo dove abita, potrà mettere anche il lavoro che sta cercando.
Inoltre è possibile tramite un pulsante decidere se cercare lavoro in base o meno alla distanza. Ovvero se è disposto a trasferirsi o preferisce rimanere dove è.

Gli verranno forniti una lista di lavori, dal migliore in margine di profitto al peggiore.

IMPLEMENTAZIONE - DATI Offerte lavorative

Abbiamo provato i 3 metodi possibili e abbiamo optato alla fine per una combo tra il primo metodo e il secondo.

Pre - 3° Metodo - Ricerca A* in Tempo Reale:

> "Andremo a usare API per trovare i dati in tempo reale da siti come linkedin per prendere le informazioni e dare veramente soluzioni reali in tempo reale del lavoro migliore."

Post - 3° Metodo - Fail

Dopo aver cercato per tanti API, abbiamo concluso che le API disponibili erano troppo limitate per poter fare progetti di AI search.
Nonostante però avrebbero dato risultati molto più veri e sicuramente rimane un'ottima soluzione se un giorno si volesse investire in questo progetto.

Pre - 2° Metodo - Dataset online:

> "Andremo a scaricare dei dataset con tutte le informazioni lavorative e lavoreremo nel search su quei dati."

Post - 2° Metodo - Succed

Abbiamo purtroppo trovato solo Datasets su dati del 2022.

Di cui:

- Dataset sulle offerte lavorative
- Dataset sui costi di vita

PROBLEMA:

Il dataset sui costi di vita non prendeva anche le città ma solo le nazioni.

Quindi abbiamo creato anche una parte nel sito in cui puoi tu andare ad aggiungere città come se fossero nazioni. Il tutto confonde ma in questo modo si possono fare studi in base alla città, favorendo di molto ricerche più vicine.

Il dataset sulle offerte lavorative invece ha il problema che ha le coordinate di latitudine e longitudine in base alla nazione e non in base alla città. Inoltre sono state tutte date alla città capitale della nazione. Perciò abbiamo creato

anche un sistema per andare a creare offerte lavorative e poter aggiornare così il dataset.

Pre - 1° Metodo :

Andremo a scrivere dei dati creati da noi come esempio e troveremo la soluzione migliore per il lavoro.

Post - 1° Metodo - Succed

Creando dei metodi siamo riusciti ad aggiornare i datasets.

Unico problema è che nel caso del dataset sui job, essendo molto grande ci mette davvero tanto ad essere aggiornato o essere letto. Quindi l'utente deve aspettare molto.

Invece per quello sulle città, essendoci solo le nazioni, non superavano le 130 righe. Sono già state messe però delle città per poter fare dei test. E l'utente quando vuole può aggiungerne di nuove.

AI – Search:

A*

- **Stato iniziale:**

  Posizione attuale dell'utente (lat, lon), lavoro desiderato

- **Stati:**

  Ogni stato rappresenta una città nel mondo con almeno un'offerta lavorativa corrispondente al lavoro desiderato.

  Ed ogni città ha i relativi costi di vita e di tasse.

- **Azioni:**

  Spostarsi da una città all'altra (in realtà simulato solo come salto diretto dal nodo iniziale a un possibile nodo finale).

- **Costo g(n):**

  Calcolato come:

  distanza * 1000 * w

(così che le distanze più alte diano una priorità alla priorityQueue che viene usata per l'A* di dare minore priorità alle offerte con distanza più grande).

1000 è stato deciso dopo vari test, perché dava i risultati migliori

w può essere 0 o 1.

- w = 1: Se la persona vuole cercare lavoro da dove abita
- w = 0 Se la persona sta cercando proprio di cambiare vita ed è disposto a trasferirsi

- **Euristica h(n):**

Funzione che tiene conto di: salario, costo della vita, affitto.

Formula usata:

$$h(n) = (affitto + costo\_vita) - stipendio$$

In questo modo più lo stipendio sarà grande e più piccolo sarà l'heuristica.
Di cui la prioriry Queue può gestire anche i numeri negativi.

- **Funzione di valutazione:**

$$f(n) = g(n) + h(n)$$

Grazie per l'attenzione.

Team Carrer AI Maps

Florin Bran, Fiordj Ferro