Sistem expert pentru recomandarea destinaţiilor turistice

*Cosma Patricia-Giorgiana și Damian Melisa-Dana*

1. **Introducere**

Proiectul curent este o aplicaţie web dezvoltată folosind limbajul de programare Python și framework-ul Django. Aplicația oferă recomandări de destinații turistice pe baza unui set de întrebări adresate dinamic utilizatorului. Motivația din spatele alegerii acestei teme este dorința de a ușura procesul relativ obositor de selecție a unei locații de vacanță datorat numeroșilor factori ce sunt luați în considerare.

1. **Design și implementare**

Django este un framework open-source bazat pe Python, care folosește tiparul arhitectural model-view-template (MVT) și al cărui principal scop este facilitarea dezvoltării de site-uri web complexe ce folosesc baze de date.

Models

În crearea aplicației, s-a folosit o baza de date relațională în SQLite. Aceasta conține trei tabele, una dedicată întrebărilor – Questions, una dedicată răspunsurilor posibile pentru o întrebare – Choices, maparea făcându-se prin intermediul cheilor secundare, care sunt id-urile numerice ale întrebărilor, și ultima tabelă, Results, ce conține toate destinațiile posibile împreună cu câte o scurtă descriere pentru fiecare. Aceste trei tabele definesc modelele ce vor fi folosite mai departe pentru a popula interfața cu utilizatorul.

Views

Aici se prelucrează acțiunile utilizatorului în paginile HTML (în momentul în care se face un salt la următoarea pagină HTML, se transmite o cerere de tip POST, în urma căreia se preiau răspunsurile din pagina respectivă) și se fac redirecționări către alte pagini. Redirecționările se fac dinamic, în sensul că saltul la o pagină se realizează ținând cont de răspunsul la întrebarea din pagina curentă. De asemenea, în Views se regăsesc și celelalte funcționalități din backend, adică partea de parsare și de mașină de inferență.

Templates

Partea de templates din tiparul MVC se refera la crearea paginilor HTML. Aplicația încorporează trei pagini HTML importante, una este pagina rădăcină – index, care este chiar începutul propriu-zis al sistemului expert, una este pagina dedicată unei întrebări oarecare, iar cea de-a treia este pagina care redă rezultatele obținute prin inferență de către sistemul expert.

Parsarea regulilor și a concluziilor

Regulile și concluziile se regăsesc într-un fișier text astfel: pe prima linie se află șirul concluziilor, iar pe următoarele rânduri se află regulile sub forma “IF premisa1 AND premisa2 AND … AND premisaN TEHN concluzie parțială/finală”. Din fișierul text, se face extragerea concluziilor finale într-o listă *final\_conclusions*, apoi se prelucrează regulile și se extrag premisele în altă listă *premises* și concluziile parțiale (acele concluzii ce nu se regăsesc în lista *final\_conclusions*) în lista *partial\_conclusions*. Fiecare regulă este salvată într-un dicționar *rules* care are drept cheie, o concluzie, indiferent de tipul ei (parţială/finală), și drept valori, lista de premise.

Mașina de inferență

Mecanismul de inferență ales este *forward chaining*. Se pornește de la un răspuns al utilizatorului ce reprezintă o premisă și se caută o regulă din care să facă parte. Apoi, când s-au găsit toate premisele care activează regula, se verifică dacă concluzia este una finală, caz în care se salvează în lista de rezultate finale ale utilizatorului *final\_results*,sau dacă este parțială, caz în care se adaugă în lista *user\_partial\_conclusions*. Pornind de la concluziile parțiale obținute, se încearcă activarea altor reguli care să le cuprindă în partea de premise.

După oferirea recomandărilor, când se reia procesul pentru alt utilizator, listele populate cu date corespunzătoare utilizatorului anterior sunt golite.

1. **Utilizare**

Pentru utilizarea aplicației, este necesară instalarea distribuției *Anaconda*, care conține, în mod implicit, *Jupyter Notebook*, un mediu computațional interactiv web-based pentru programarea în *Python*, acesta fiind și limbajul de programare utilizat.

În ceea ce privește interfața cu utilizatorul, a fost folosit *Django*,  un framework pentru dezvoltarea aplicațiilor web, scris în Python, care are propriul server local de dezvoltare. Ca bază de date, Django vine la pachet cu *SQLite* și, mai mult decât atât, oferă o interfață, în secțiunea admin, pentru a ușura adăugarea/modificarea/ștergerea datelor din tabele. Django trebuie, de asemenea, instalat înainte de prima utilizare a aplicației.

Primul pas, după ce toate cerințele tehnice au fost îndeplinite, este rularea server-ului din Anaconda Prompt, prin intermediul comenzii: *python manage.py runserver*, dar nu înainte de a naviga în folder-ul aplicației web. În continuare, trebuie deschis server-ul local *127.0.0.1:8000/expertsystem/*, unde *expertsystem* este numele aplicației.

Interfața cu utilizatorul conține 11 întrebări care alcătuiesc un chestionar, pe baza căruia acesta va primi o recomandare, cu privire la destinațiile de vacanță potrivite. Fiecare pagină web cuprinde câte o întrebare cu variantele de răspuns corespunzătoare si un buton pentru trecerea la următoarea întrebare. Anumite întrebări au variante multiple de răspuns (*check-boxes*), pe când altele au o singură variantă de răspuns (*radio buttons*). Procedura de trecere de la o interogație la alta se realizează în mod dinamic, în sensul că utilizatorului nu i se vor mai adresa întrebări care nu au legatură cu dorințele pe care le-a exprimat până la acel moment. Prin urmare, interogația i+1 depinde de răspunsul dat la întrebările 0, 1, …, i.

Pagina web care conține ultima interogație cuprinde și un buton care face o redirecționare către pagina cu rezultatele sistemului de recomandare, adică acele destinații turistice care sunt pe gustul utilizatorului. Fiecare destinație de vacanță include câte o scurtă descriere a locului, în acest fel oferindu-se o motivaţie a recomandării acelui loc. La final, exista un buton care permite reluarea chestionarului.





1. **Concluzii**

În acest proiect s-a realizat un sistem expert bazat pe mecanismul de inferență de tip *forward chaining* folosind framework-ul Django. Scopul lui este să ajute utilizatorul să găsească destinații turistice care îi îndeplinesc cerințele. Ulterior se pot muta regulile în baza de date, se pot aduce optimizări de performanță și îmbunătățiri de design în cazul interfeței cu utilizatorul.