Componentele proiectului UIA

= cerințe de complexitate pentru nota maximă =

Componenta A (4 puncte). O bază de grafuri construită în RDF4J.

- Aceasta trebuie trimisă pentru analizare, cu 24 de ore înainte de data examenului, sub forma unui fişier unic scris în sintaxa .trig,
- Codul fişierului TriG să fie îngrijit pentru o citire cât mai uşoară (cu comentarii care să indice unde începe fiecare secțiune: declaraţii de clase, de proprietăţi, de etichete etc.).

Componenta B (6 puncte). O aplicaţie Web scrisă în limbajul de programare pe care îl stăpâniţi cel mai bine. Va trebui să investigaţi cum vă puteți conecta la RDF4J din limbajul de programare ales. Exemplele din seminar sunt în PHP sau Python dar aceleași principii se regăsesc în orice limbaj - aveți următoarele căi la dispoziție:

  - Prin cereri HTTP configurate (conform serviciului REST oferit de RDF4J)
  - Prin librării dedicate (precum EasyRDF din PHP)
  - Pentru Java, RDF4J oferă și un folder LIB cu fișiere .jar integrabile

Cerințe de pentru baza de grafuri

- Să includă atât instanțe cât și o terminologie (vocabular RDF) inventată de voi și construită cu RDF Schema
- Ierarhia de clase a vocabularului să ofere cel puţin două utilizări ale relației **rdfs:subClassOf** între clase propuse de voi (inventate de voi sau preluate din Schema.org)
- Ierarhia de proprietăți să ofere cel puțin două utilizări ale relației **rdfs:subPropertyOf** între clase propuse de voi (inventate de voi sau preluate din Schema.org)
- Fiecare proprietate propusă de voi să aibă declarat domeniul (**rdfs:domain**) şi codomeniul (**rdfs:range**)
- Fiecare proprietate propusă de voi să aibă o descriere (**rdfs:comment**) care să explice în limbaj natural ce reprezintă (ex.: "reprezintă relația dintre ....")
- Fiecare clasă propusă de voi să aibă la rândul său o descriere în limbaj natural (de forma "reprezintă mulțimea tuturor ...)
- Pentru orice termin X să existe minim un răspuns diferit de rdfs:Resource la interogarea **Ce este X?** (Ce tip are X?)
- Baza de grafuri să conțină minim 2 grafuri distincte (gruparea în grafuri o faceți după ce criterii doriți)

Cerințe de complexitate pentru aplicația Web

- Partea front-end (codul HTML) să fie distilabilă, deci cu descrieri încorporate prin una dintre sintaxele ***microdata*** **sau** ***RDFa*** (la alegere). În grafurile distilate să se regăsească **instanțe ale minim unui tip** (clasă) dintre cele propuse de voi cu minim **două dintre proprietățile** propuse de voi
- Să realizeze atât operaţii de **citire** cât şi de **scriere** în baza de grafuri
- La operaţiile de citire:
  - Să nu se vadă termeni URI în interfaţa cu utilizatorul
- La operaţiile de scriere:
  - Afirmațiile introduse/modificate/şterse să fie determinate de acţiunea utilizatorului în front-end (selecţia unor opţiuni, introducerea unor date). Cu alte cuvinte, să nu se uploadeze direct, fără nici un fel de procesări, un fişier RDF brut citit de pe disc sau de pe Internet.

Alte remarci

- Nu trebuie să fie un site complet => nu e nevoie să aveți login, sesiune, cookie etc.
- Nu e obligatoriu să aibă mai multe pagini => poate fi un single-page front-end de tip AJAX care să acopere toate operațiile cerute (plus scripturile server-side care să gestioneze interogările)
- Nu e obligatoriu nici măcar să aveți scripturi server-side => grafurile pot fi aduse direct în clientul JavaScript, dar s-ar putea să considerați mai dificil acest mod de lucru căci...
  - ...necesită configurarea Tomcat (pe care e RDF4J) pentru a permite cereri cross-domain CORS dinspre serverul pe care găzduiți aplicația voastră
  - ...librăriile JavaScript pentru procesarea grafurilor sunt mai puțin prietenoase decât procesarea JSON (totuși, pentru cine se încumetă, aveți și o listă de librării JS la https://www.w3.org/community/rdfjs/wiki/Comparison_of_RDFJS_libraries)