Proiect Web Semnatic

Nume membri echipă:

- Filep Rebeca, grupa 3

- Marcu Andreea-Cosmina, grupa 4

Tematică proiect: Restaurante

Tabele:

- tabel 1: Restaurants (One)

- tabel 2: Menu items (Many)

Componente realizate în proiect:

- **Buton 1**: a fost realizat prin intermediul serverului RDF4J și funcționalitatea sa este ilustrată prin Tabel 1

- **Buton 2**: a fost realizat prin intermediul serverului JSON 1, folosind tehnici REST (tipuri de cerere GET pentru verificare și POST pentru adăugare în server), funcționalitatea sa este testată prin accesarea:

<http://localhost:4000/restaurants> (acesta se ocupă de verificarea restaurantelor afișate în Tabel 1)

<http://localhost:4000/menuitems> (acesta se ocupă de verificarea felurilor de mâncare din meniurile restaurantelor, afișate la rândul lor, în Tabel 1)

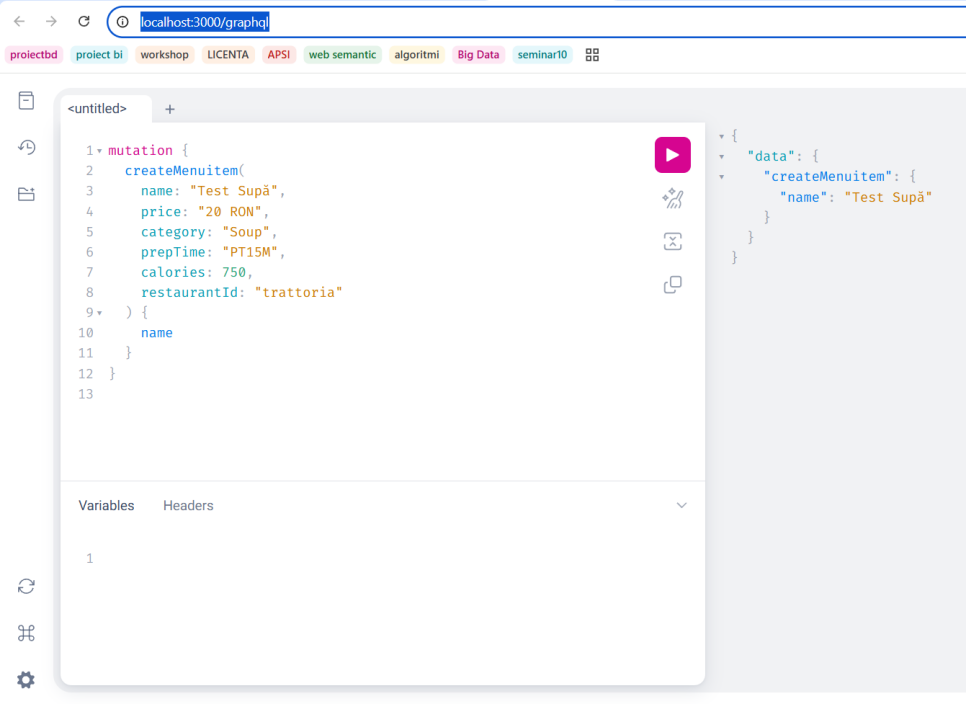
- datele sunt introduse automat, fără existența unor duplicate, în fișierul db.json

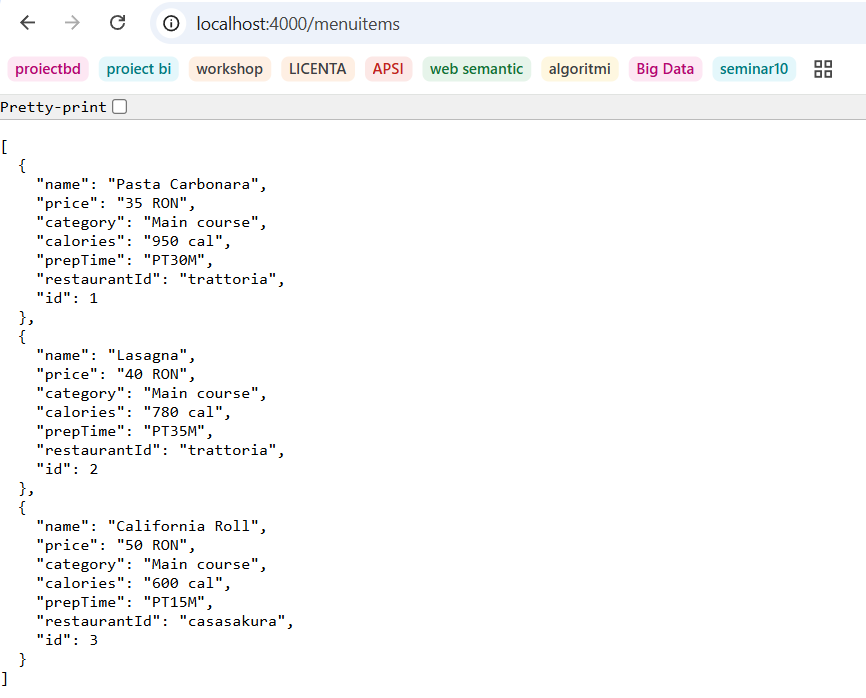
- **Buton 3**: a fost realizat prin intermediul serverului JSON 1, folosind tehnici REST (interogări - GET), funcționalitatea sa este afișată în Tabel 2

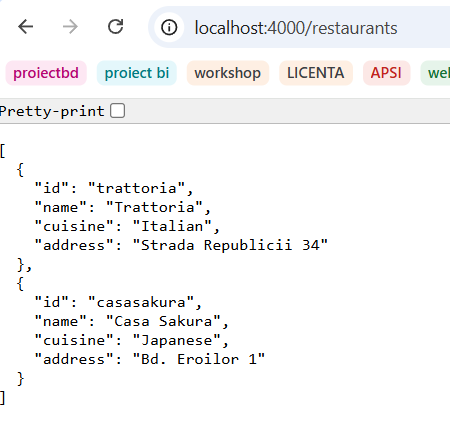
Totodată, în urma apăsării butonului 3, se populează dropdown-ul și se completează formularul pentru funcționarea butonului 4

- **Buton 4**: a fost realizat prin intermediul serverului JSON 2, folosind tehnici GraphQL prin construirea unei mutații (mutation) si trimiterea datelor din formular cu fetch() la <http://localhost:3000/graphql>

- **Buton 5**: a fost realizat prin intermediul serverului JSON 2, folosind tehnici GraphQL (query), funcționalitatea este afișată în Tabel 3, prin interogarea datelor din server







Fișier db.json - afișare când se populează în urma apăsării butonului 2

{

  "restaurants": [

    {

      "id": "trattoria",

      "name": "Trattoria",

      "cuisine": "Italian",

      "address": "Strada Republicii 34"

    },

    {

      "id": "casasakura",

      "name": "Casa Sakura",

      "cuisine": "Japanese",

      "address": "Bd. Eroilor 1"

    }

  ],

  "menuitems": [

    {

      "name": "Pasta Carbonara",

      "price": "35 RON",

      "category": "Main course",

      "calories": "950 cal",

      "prepTime": "PT30M",

      "restaurantId": "trattoria",

      "id": 1

    },

    {

      "name": "Lasagna",

      "price": "40 RON",

      "category": "Main course",

      "calories": "780 cal",

      "prepTime": "PT35M",

      "restaurantId": "trattoria",

      "id": 2

    },

    {

      "name": "California Roll",

      "price": "50 RON",

      "category": "Main course",

      "calories": "600 cal",

      "prepTime": "PT15M",

      "restaurantId": "casasakura",

      "id": 3

    }

  ]

}

Filtre aplicate în proiect:

Nu au fost accesate rutele de descărcare completă, ceea ce înseamnă că filtrele au fost aplicate direct la pasul de descărcare a datelor

- Buton 1: filtrul folosit în cadrul acestui buton, ia doar datele care au la categorie „Main course”

„FILTER(?category = "Main course")”

- Buton 2: filtrul folosit în cadrul acestui buton ia doar preparatele care durează mai puțin sau exact 30 de minute pentru a fi gătite

„menuitems?prepTime\_lte=PT30M”

- Buton 3: filtrul folosit în cadrul acestui buton ia doar preparatele care au mai mult sau exact 700 de calorii

„(filter: { calories\_gte: 700 })”

Bonus aplicat:

- la butonul 1, pe lângă afișarea datelor în tabelul HTML, am generat și un fișier JSON-LD. Acest fișier conține date semantice despre fiecare restaurant și preparat din restaurant, în format Schema.org

Tehbologii foloste:

- pentru partea de Front-end, am folosit HTML și JavaScript

- pentru partea de Back-end am folosit JavaScrpit

-pentru a trimite cereri SPARQL către RDF4j am folosit proxy.php

-JSON Server 1(REST): <http://localhost:4000/>

-JSON Server 2(GraphQL): <http://localhost:3000/>

Instrucțiuni de rulare:

1. Serverul RDF4J va fi pornit din „startup.bat”- din fișierul Tomcat din XAMPP. De asemenea Apache din XAMPP rulează pe portul 8080. Pentru afișarea site-ului în browser-ul Chrome, se va folosi index.html („http://localhost:8080/proiectweb/index.html”).
2. Serverul RDF4J rulează pe portul 8081. În acest server se va crea un repository cu id=grafexamen, titlu=poiectweb. În acest repository se va încărca fișierul „restaurant.trig” și va fi selectat „Data Format = TriG”.

Acestea sunt necesare pentru buna funcționare a butonului 1

1. Serverul JSON Server se pornește cu comanda ”json-server db.json --port 4000”.

Acesta este necesar pentru buna funcționare a butoanelor 2 și 3.

1. Serverul GraphQL se pornește cu comanda „json-graphql-server db-graphql.json --p 3000”.

Acesta este necesar pentru buna funcționare a butoanelor 4 și 5.

1. În Notepad++, au fost scrise fișierele script.js, index.html, restaurant.trig și proxy.php

- script.js, index.html și proxy.php sunt salvate în C:\XAMPP\htdocs\proiectweb

- restaurant.trig poate fi salvat oriunde

1. Pentru funcționarea serverului rdf4j, a fost descărcat fișierul „rdf4j-server”, și a fost copiat in C:\XAMPP\tomcat\webapps
2. Fisierele db.json și db-graphql.json au fost create în Visual Code și pot fi salvate oriunde(acolo unde sunt salvate serverele)

Din cauza unor probleme neașteptate, am fost nevoite să schimbăm porturile default pentru Apache XAMPP și RDF4J, de la 80 la 8080, și de la 8080 la 8081.