1. *Arhitectura aplicației*

Pentru implementarea funcționalităților aplicației web, s-a avut în vedere, pe de o parte, gestionarea eficientă a bazei de date, și pe de altă parte, dezvoltarea unor interfețe prietenoase cu utilizatorul și a unor funcții și operații care să faciliteze interacțiunea cu aceasta. Putem spune, astfel, că aplicația web poate fi considerată o ***aplicației client/server***.

Modelul client/server reprezintă o arhitectură de aplicație distribuită prin intermediul căreia sunt partajate resurse între server (furnizorul de servicii) și clienți (entitățile care solicită servicii). Comunicarea între server și client se realizează cu ajutorul unei rețele de calculatoare care prezintă specificații hardware diferite. De asemenea, orice calculator din această rețea poate reprezenta fie serverul, fie clientul. În cazul actual, clientul este dispozitivul de pe care utilizatorul are acces la aplicație și la datele necesare pentru a realiza anumite funcții, operații pe care le realizează prin interfața aplicației. Aceste date sunt recepționate de pe server.

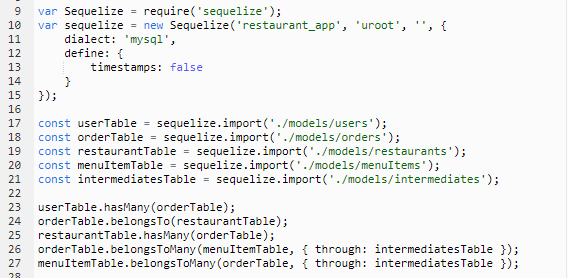
În cadrul aplicației, conexiunea dintre client și server se stabilește prin intermediul protocolului HTTP. Odată stabilită conexiunea, clientul trimite o cerere către server, sub formă de JSON. În urma preluării cererii, serverul se ocupă de rezolvarea acesteia și în final trimite un răspuns clientului cu datele corespunzătoare.

Implementarea ***serviciilor REST***. REST – Representational State Transfer – reprezintă un standard web care utilizează protocoale HTTP pentru furnizarea accesului la date. În cadrul serviciilor REST, datele sunt considerate resurse care pot fi accesate prin identificatori de resurse (URI – Uniform Resource Identifiers), sub forma unor link-uri. Schimbul de resurse dintre client și server se realizează printr-o interfață și protocoale standardizate.

Datele utilizate în aplicație sunt stocate într-o bază de date SQL - ***MySQL*** Stocarea datelor s-a realizat într-o bază de date relațională, aceasta cuprinzând mai multe tabele (users - pentru stocarea utilizatorilor, menuItems - pentru stocarea catalogului de produse, orders - pentru stocarea comenzilor, etc.).

***Node.js*** este o platformă "open source" JavaScript, disponibil pe mai multe sisteme de operare (Windows, Linux, MacOS) care execută cod JavaScript pe partea de server. De-a lungul timpului, limbajul Javascript a evoluat de la a fi folosit doar pe partea de client, scripturile scrise fiind încorporate în HTML și executate de un motor specific JavaScript în browser, la dezvoltarea unor scripturi care să ruleze pe partea de server. În felul acesta, Node.js este considerat "o paradigmă Javascript peste tot", dezvoltarea aplicațiilor web sau a aplicațiilor mobile hibride realizându-se prin intermediul unui singur limbaj de programare, în loc de limbaje diferite pentru client și server.

Astfel, utilizând Node.js ca tehnologie pentru server și modulul ***sequelize***(reprezentând o soluție rapidă și directă pentru modelarea datelor din aplicație) ca dependență pentru facilitarea conexiunii între server și baza de date, aplicația reușește să acceseze datele din baza de date. Includerea dependenței pe partea de server si configurarea acesteia se realizează ca în figura de mai jos:

***Fig. 1****: Configurarea bazei de date pe partea de server*

Pentru implementarea părții de client s-a utilizat ***Angular***. Spre deosebire de alte framework-uri web, Angular se bucură de o mare popularitate în rândul dezvoltatorilor, fiind tehnologia standard pentru dezvoltarea aplicațiilor web. Este cunoscut faptul că Angular este cel mai popular framework pentru "Single Page Applications", oferind o gamă variată de caracteristici pentru a facilita dezvoltarea aplicațiilor web. Pentru stilizarea componentelor web dezvoltate, s-a utilizat libraria ***Bootstrap***. În final, pentru rularea aplicației, s-a creat un folder ce conține tot codul aferent părții de client, în urma procesului de build. Acest folder este servit pe partea de server, la o anumită cale URL.

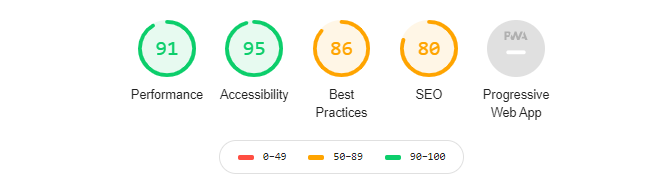
1. *Testarea performanței*

Pentru auditul de performanta al aplicației, a fost utilizat instrumentul ***Lighthouse***. Acesta calculează scoruri în funcție de diferite criterii - Performanță, Accesibilitate, PWA, SEO, ‘Best Practices’. În urma rulării testelor, avem ca rezultat un raport complet împărțit pe criterii, cuprinzând atât testele care au trecut, cât și testele care au picat și sugestii pentru îmbunătățirea scorului. Scorul variază de la 0 la 100, unde 0 este cel mai slab rezultat, iar 100 cel mai bun.

De asemenea, exista o harta pentru interpretarea rapidă a scorurilor, aceasta bazându-se pe culori. Hărțile cu coduri de culori pentru aceste intervale de scoruri de performanță sunt:

* 0 până la 49 (lent): roșu
* 50 până la 89 (medie): portocaliu
* 90 până la 100 (rapid): verde

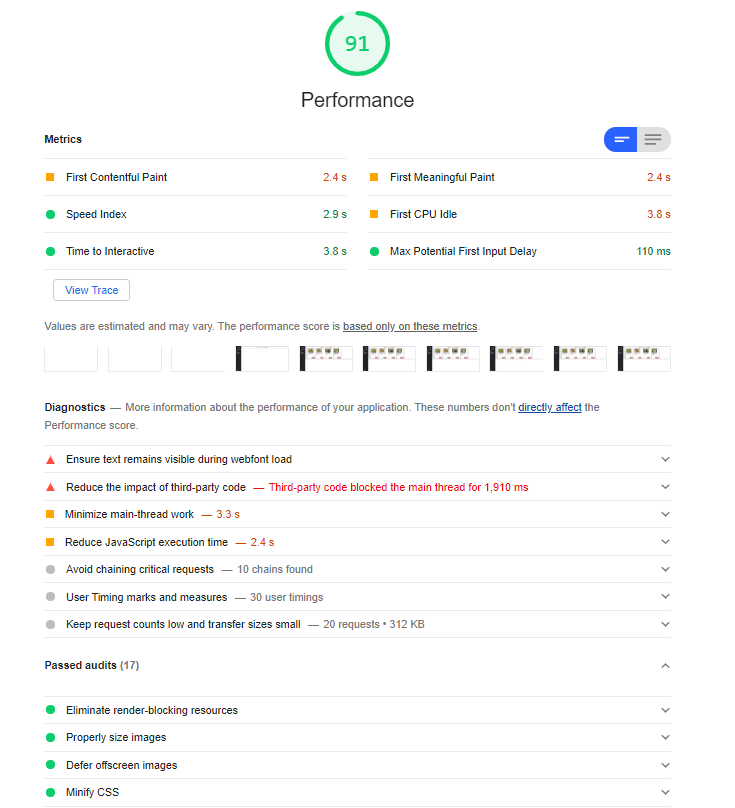
Pentru aplicația web propusă, scorurile sunt următoarele:



*Fig. 2: Scoruri pentru auditul de performanta*

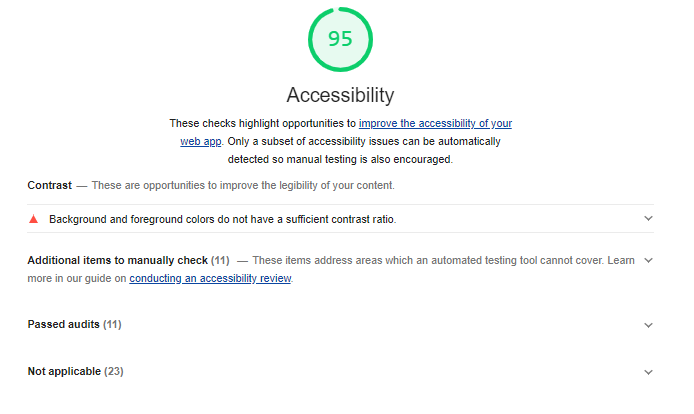
Împărțit pe categorii, avem următoarele rezultate mai detaliate:

Din punct de vedere al performanței, am obținut un punctaj de 91 din 100. Impactul cel mai mare îl are încărcarea librăriilor externe aplicației care în cazul de față au blocat timp de 1.910 ms firul principal de execuție, conducând la o încărcare mai lentă a paginii. De asemenea, afișarea textului în timpul încărcării fonturilor web este un aspect important în cazul acesta.



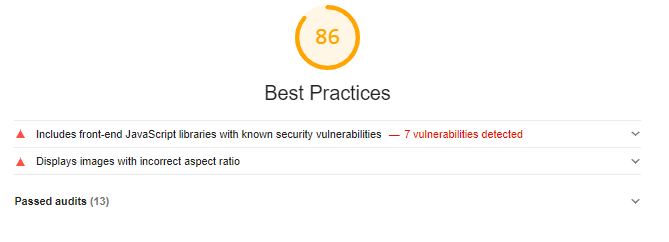
*Fig. 3: Performanța aplicației*

Din punct de vedere al accesibilității, am obținut un scor de 95 din 100. Singurul punct evidențiat pentru criteriul acesta îl reprezintă contrastul paginii. Raportul de contrast pentru fundal și prim-plan nu este cel adecvat, fiind recomandată modificarea acestuia pentru îmbunătățirea lizibilității conținutului.



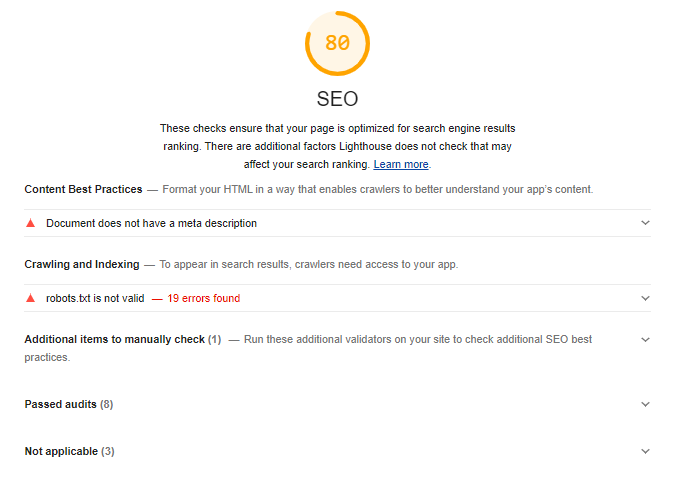
*Fig. 4: Accesibilitatea aplicației*

Analizând cele mai bune practici pentru aplicațiile web, am obținut un scor de 86 din 100. Acest scor poate fi îmbunătățit prin eliminarea librăriilor Javascript care conțin vulnerabilități în securitate și afișarea corectă a imaginilor din punct de vedere al raportului de aspect.



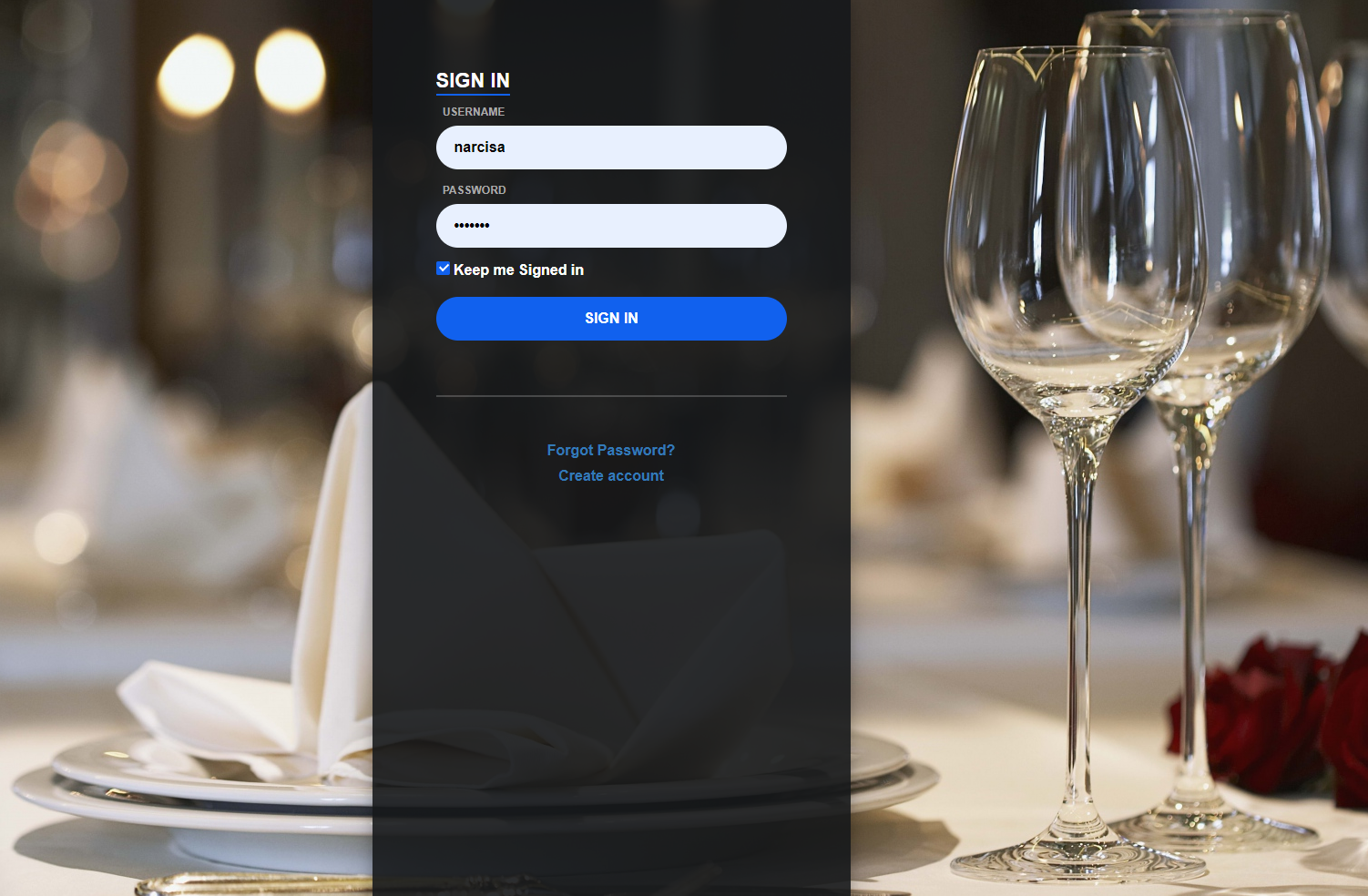
*Fig. 5: Best practices*

Din punct de vedere SEO, am obținut un punctaj de 80 din 100. Sugestiile pentru optimizarea poziției în clasamentul rezultatelor motorului de căutare se referă la adăugarea unei meta descrieri și la înlăturarea erorilor legate de unele fișiere cu format text (robots.txt în cazul de față).

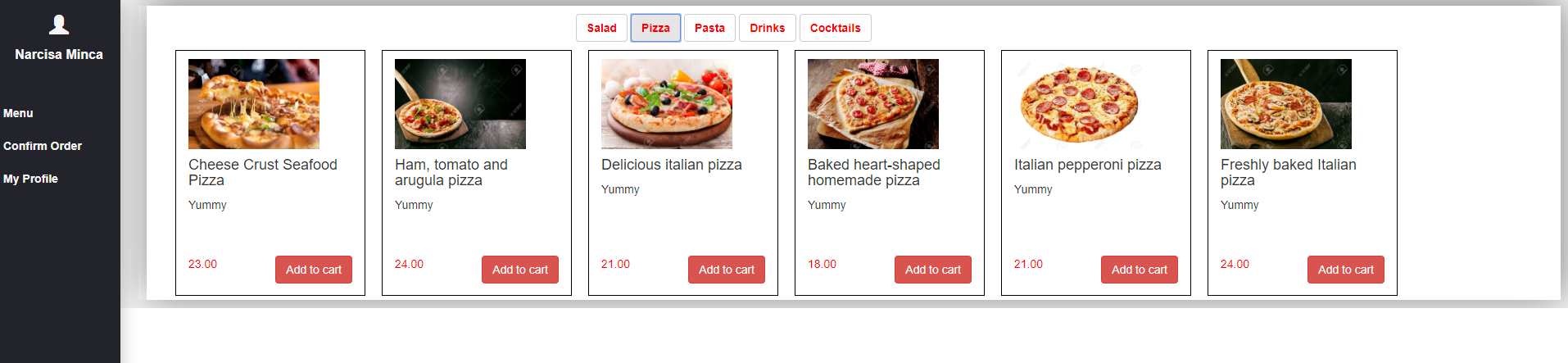


*Fig. 6: SEO*

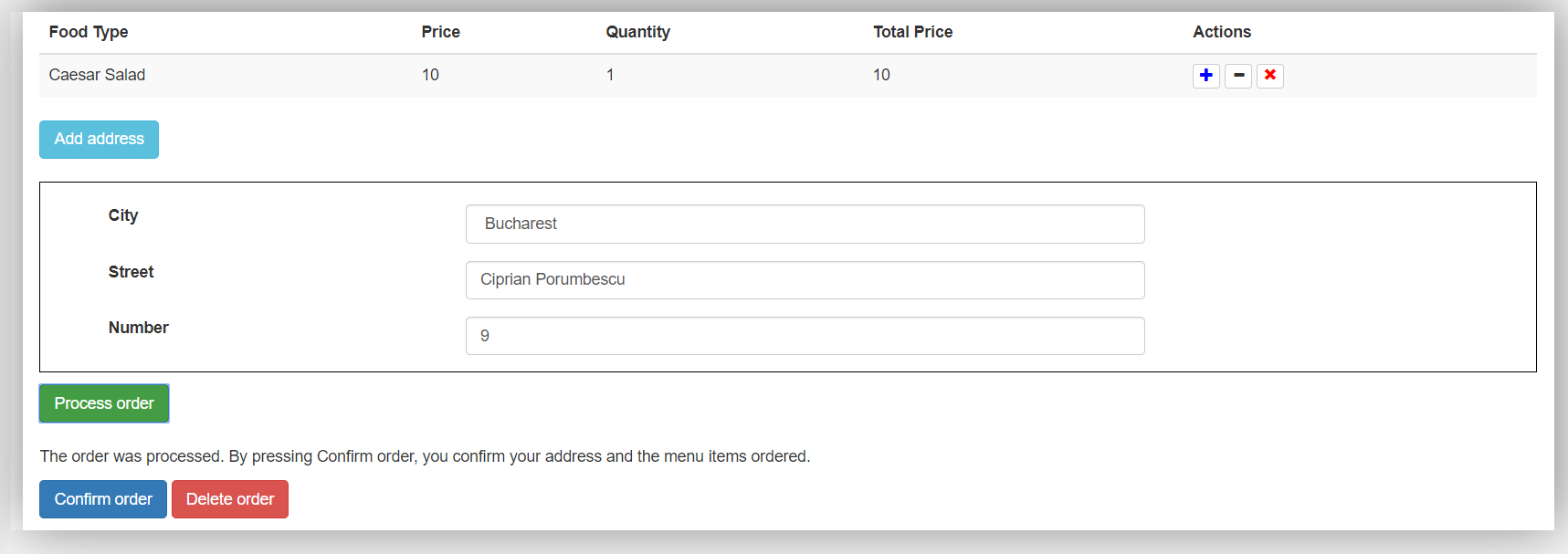
1. Pagini aplicație



*Fig. 7: Pagina de autentificare*



*Fig. 8: Catalog produse pe categorii*



*Fig. 9: Completarea procesului de comandare produse*