*Tehnologii web proiect:* Gust Divin

1. Prezentare generala :

Aplicatia GustDivin este o platforma web full-stack dezvoltata pe stiva MERN (MongoDB, Express.js, React.js, Node.js), destinata digitalizarii procesului de comanda si administrare a unei cofetarii. Sistemul implementeaza o arhitectura RESTful, o interfata responsive (pana la o latime minima de 320px) si suporta mai multe roluri de utilizatori (guest, utilizator si administrator), fiecare cu permisiuni specifice.

1. Tehnologii utilizate :

Aplicatia este construita pe o arhitectura JavaScript completa, cu separare clara intre client si server. Tehnologii folosite:

* **MongoDB**: bază de date NoSQL pentru stocarea datelor
* **Express.js**: framework backend pentru construirea unui API RESTful care gestionează cererile și logica aplicației
* **React.js**: bibliotecă frontend pentru crearea intefetei
* **Tailwind CSS**: framework CSS utilitar care faciliteaza dezvoltarea rapida a interfetei prin utilizarea claselor predefinite direct in HTML/JSX
* **Node.js**: mediu de rulare server-side pentru executarea logicii backend
* **Axios**: pentru realizarea cererilor HTTP între frontend și backend
* **CORS**: permite cereri securizate între origini diferite
* **JWT + Cookie-uri HTTP-Only** : pentru gestionarea securizată a sesiunilor și autentificarea utilizatorilor

1. Structura aplicatiei – Frontend :

* **public**
  + acest folder contine resurse statice utilizate de aplicatie, precum imagini
  + initial, am creat un folder assets pentru a stoca aceste imagini, insa, avand in vedere ca pozele folosite nu necesita modificari sau optimizare dinamica, am decis sa le mut in public pentru a imbunatati performanta aplicatiei. Astfel, imaginile sunt accesibile direct prin URL si nu sunt incluse in procesul de build, ceea ce reduce dimensiunea bundle-ului final si accelereaza incarcarea aplicatiei.
* **src/components**
  + contine componente React reutilizabile mai complex (alcatuite din mai multe componente)
* **src/components/ui**
  + stocheaza componente React mai simple care sunt blocuri de constructie reutilizabile pentru interfata utilizator
  + aceste componente sunt importate in componentele „mari” din components sau direct in pagini
* **src/config**
* contine fisiere care stocheaza setarile si constantele aplicatiei, care sunt folosite in mod global pe tot parcursul dezvoltarii aplicatiei
* scopul acestui folder este de a centraliza aceste valori pentru a le putea gestiona si modifica mai usor
* **src/context**
* in implementarea initiala, logica de comunicare cu backend-ul (API REST conectat la baza de date) era functionala, insa interfata nu reflecta in timp real modificarile de stare (ex. actualizarea cosului, adaugarea la favorite), necesitand un refresh manual pentru re-randare
* pentru a remedia aceasta lipsa de sincronizare intre datele persistente si cele afisate, am introdus *context providers React*, care centralizeaza starea aplicatiei si permit propagarea automata a modificarilor catre componentele consumatoare.
* astfel, contextele asigura o actualizare reactiva a UI-ului, fara a reface manual logica de fetch la fiecare interactiune

Obs : in cazul productContext, implementarea nu a fost finalizata in versiunea curenta a aplicatiei. Astfel, functionalitatile precum adaugarea unui review intr-o pagina de produs nu actualizeaza automat informatiile produsului afisate in componenta principala, ci doar in componenta ReviewForm

* **src/hooks**
  + contine hook-uri personalizate (custom hooks) care centralizeaza logica legata de interactiunea cu starea aplicatiei si comportamente comune. Aceste hook-uri simplifica componentele, evitand duplicarea codului si separand clar responsabilitatile
* **src/layouts**
* contine componente de layout reutilizabile care definesc structura vizuala si functionala a paginilor, in functie de contextul utilizatorului (ex. admin vs. utilizator obisnuit). Aceste layout-uri includ elemente comune precum navbar si footer
* **src/pages**
  + contine componentele care reprezinta efectiv paginile aplicatiei, fiecare corespunzand unei rute
* **src/sections**
  + pentru paginile mai complexe din pages, continutul este organizat pe sectiuni distincte, salvate in folderul sections
  + acestea sunt componente mai mici, dedicate unor parti specifice ale unei pagini
* **src/services -** interfata cu backendul aplicatiei
  + acest folder contine logica de comunicare intre aplicatia frontend si API-ul backend
  + rolul sau este sa centralizeze toate cererile HTTP, pastrand componentele React curate si orientate doar spre prezentare si interactiune cu utilizatorul
* **src/services/api/index.js**
  + defineste o instanta Axios folosita in intreaga aplicatie pentru trimiterea cererilor HTTP catre backend
* **src/services/api/endpoints.js**
  + organizeaza toate functiile care trimit cereri HTTP, grupate logic (ex: userEndpoints, cartEndpoints, etc.)
  + fiecare functie corespunde unui endpoint din backend si returneaza rezultatul apelului efectuat de instanta apiClient
* **src/services/\*Service.js**
* module care utilizeaza endpoint-uri pentru a trimite cereri HTTP si a manipula datele primate
* aceste servicii ofera o abstractie pentru logica de comunicare cu backend-ul, permitand componentelor UI sa interactioneze cu datele fara a se ocupa direct de detaliile implementarii cererilor sau erorilor
* **src/App.jsx**
  + acest fisier reprezinta componenta principala a aplicatiei React, care defineste rutele pentru diferitele pagini ale aplicatiei
* **src/index.css**
  + fisierul CSS defineste stilurile globale ale aplicatiei
* **src/main.jsx**
  + acest fisier este punctul de intrare al aplicatiei React

1. Functionalitatile aplicatiei :

* **Guest:**
  + vizualizare lista produselor
  + vizualizare detalii despre produse (pagina produselor)
  + poate adauga produse in cos (date doar din sesiunea curenta) - cat timp e valabil token-ul
  + nu va putea beneficia de lista de favorite (va primi mesaj de eroare la incercarea adaugarii unui produs in lista de favorite)
  + nu va putea lasa review-uri produselor
  + va putea plasa o comanda, insa nu va avea un istoric al comenzilor si nici un profil propriu
* **User:**
  + tot ce poate face un guest
  + poate accesa o pagina a profilului sau unde va putea vizualiza comenzile plasate si detaliile despre acestea si tot aici va putea sa-si modifice unele detalii ale contului
  + poate lasa recenzii si rating-uri pentru produse
  + are acces la lista de favorite
* **Admin:**
  + pagina **Dashboard**
    - pagina cu statistici din ultima luna, cat si de un graphic ce cuprinde evolutia vanzatilor pe o durata de 6luni, respective un an
  + pagina de **Management produse**
    - adminul are control complet asupra postarilor produselor
    - acesta poate vizualiza toate produsele, le poate sorta prin apasarea antetului tabelului in functie de coloanele respective, le poate edita, adauga noi produse sau sterge produse existente
    - de asemenea, adminul poate modifica vizibilitatea produselor pentru utilizatori (facandu-le ascunse sau vizibile) si are posibilitatea de a vizualiza si sterge recenzi.

Obs:

* in urma discutiilor de laborator, s-a decis ca, pentru a preveni inconsistentele in aplicatie, mai ales in paginile de istoric al comenzilor, unde informatiile produsului sunt esentiale, produsele sa nu fie sterse efectiv, ci doar ascunse de utilizatori. Stergerea produselor este posibila, dar se recomanda doar pentru acele produse care nu au fost comandate sau nu sunt utilizate in alte parti ale aplicatiei
  + pagina de **Management comenzi**
    - vizualizarea tuturor comenzile plasate si filtrarea acestora dupa numite criterii (ID comanda, client, status, data etc)
    - sortare comennzi dupa diferite coloane (ex: ID, data, status)
    - prin apasarea butonul din partea drapta pot fi :
      * vizualizate detaliilor despre fiecare comanda
      * editarea campurilor : numar de telefon, adresa de livrare, status comanda ( in cazul in care comanda e in procesare, in orice alt caz doar statusul comenzii poate fi modificat)
  + pagina de **Management utilizatori** 
    - vizualizarea tuturor utilizatorilor, filtrarea si sortarea acestora
    - poate modifica detaliile unui utilizator

1. Configurare si rulare aplicatie:

**Configurare:**

* clonare repository
* in folderul *backend*:
  + copierea fisierului *.env* ( atasat in mail)
  + rularea comenzii : *npm install*
* in folderul *frontend* :
  + rularea comenzii : *npm install vite --save-dev*

**Rulare:**

* atat in folderul *frontend* cat si in folderul *backend : npm run dev*

Accesarea siteului in browser la adresa : [*http://localhost:5173*](http://localhost:5173)

Conturi puse la dispozitie pentru testare :

|  |  |
| --- | --- |
| admin :  email : [larisafilip211@yahoo.com](mailto:larisafilip211@yahoo.com)  parola: lari1234 | user:  email: [daria.andone8@yahoo.com](mailto:daria.andone8@yahoo.com)  parola: daria1234 |