UNIVERSITATEA “LUCIAN BLAGA” DIN SIBIU

FACULTATEA DE INGINERIE

DEPARTAMENTUL DE CALCULATOARE ŞI INGINERIE ELECTRICĂ

**PROIECT DE DIPLOMĂ**

Îndrumător: c.d. ce s-a ocupat de student

Absolvent:

Șufană Adrian-Daniel

Specializarea Tehnologia Informației

* Sibiu, 2023 –

UNIVERSITATEA “LUCIAN BLAGA” DIN SIBIU

FACULTATEA DE INGINERIE

DEPARTAMENTUL DE CALCULATOARE ŞI INGINERIE ELECTRICĂ

# APLICAȚIE ORGANIZARE PORTOFOLIU

Îndrumător: c.d. ce s-a ocupat de student

Absolvent:

Șufană Adrian-Daniel

Specializarea Tehnologia Informației

**Cuprins**

[APLICAȚIE ORGANIZARE PORTOFOLIU 2](#_Toc131850159)

[**1.** **Prezentare** 6](#_Toc131850160)

[**1.1** **Noțiuni introductive** 6](#_Toc131850161)

[**1.2** **Motivația alegerii temei** 6](#_Toc131850162)

[**1.3** **Scop și obiective** 7](#_Toc131850163)

[**1.4** **Realizări similare acestei teme** 7](#_Toc131850164)

[**2** **Teorie și tehnologii utilizate** 8](#_Toc131850165)

[**2.1** **Visual Studio Code** 8](#_Toc131850166)

[**2.2** **Arhitectura MERN** 9](#_Toc131850167)

[**2.2.1** **MongoDB** 10](#_Toc131850168)

[**2.2.2** **Express.js** 10](#_Toc131850169)

[**2.2.3** **React** 10](#_Toc131850170)

[**2.2.4** **Node.js** 11](#_Toc131850171)

[**2.3** **API-uri** 11](#_Toc131850172)

[**2.3.1** **React Router și Express API** 11](#_Toc131850173)

[**2.3.2** **Mongoose** 12](#_Toc131850174)

[**2.3.3** **Multer** 12](#_Toc131850175)

[**2.3.4** **JWT** 13](#_Toc131850176)

[**2.3.5** **EXIF** 13](#_Toc131850177)

[**2.3.6** **Adauga si restu api urilor aici** 13](#_Toc131850178)

[**3** **Realizarea aplicației** 13](#_Toc131850179)

[**3.1** **Mediul de dezvoltare** 13](#_Toc131850180)

[**3.2** **Dezvoltarea propriu-zisă a aplicației** 13](#_Toc131850181)

[**3.2.1** **Register/Login** 13](#_Toc131850182)

[**3.2.2** **Pagina Social Media** 13](#_Toc131850183)

[**3.2.3** **Pagina Galerie** 13](#_Toc131850184)

[**3.2.3.1** **Încărcare imagini** 13](#_Toc131850185)

[**3.2.3.2** **Extragerea metadata din imagini folosind exif api** 13](#_Toc131850186)

[**3.2.3.3** **Filtrare utilizând metadata** 13](#_Toc131850187)

[**3.2.3.4** **Sortare** 13](#_Toc131850188)

[**3.2.3.5** **Creearea modurilor de vizualizare grid a galeriei** 13](#_Toc131850189)

[**3.2.4** **Funcții aplicabile pe imaginea postată** 13](#_Toc131850190)

[**3.2.5** **Modul rapoarte** 13](#_Toc131850191)

[**4** **Concluzii și dezvoltări ulterioare** 14](#_Toc131850192)

[**4.1** **Concluzii** 14](#_Toc131850193)

[**4.2** **Dezvoltări ulterioare** 14](#_Toc131850194)

[**5** **Bibliografie** 14](#_Toc131850195)

[**5.1** **Webliografie** 14](#_Toc131850196)

1. **Prezentare**
   1. **Noțiuni introductive**

În viața de zi cu zi, peisajele și arhitectura din jurul nostru continuă să uimească prin modul în care sunt percepute de către noi, oamenii. Dar aceste imagini capturate de ochii noștrii rămân doar amintiri in momentul în care nu mai sunt subiectul principal vizual. Din fericire există modalități prin care aceste cadre pot fi păstrate fizic sau digital. Acest lucru este realizat prin fotografie.

Fotografia este un concept vechi, prima fotografie fiind realizată în anul 1827 de către fizicianul francez Joseph-Nicéphore Niépce. El a pus temelia acestei arte care până în prezent a evoluat la un nivel foarte înalt atât din punct de vedere calitativ dar și cantitativ. Dacă prima fotogafie realizată nu era decât o imagine foarte neclară, în momentul de față imaginile sunt tot mai detaliate, pline de culoare și de emoții transpuse prin intermediul unui obiectiv.

Fotografia a schimbat percepția oamenilor asupra acestor peisaje. Fotografia reprezintă arta și tehnica de a captura momente din viața noastră, folosindu-se de lumină. Prin intermediul fotografiei putem captura o varietate de peisaje, momente unice și speciale care pot fi partajate și apreciate de public, sau păstrate în propria galerie.

* 1. **Motivația alegerii temei**

Alegerea temei denumită “Aplicație de organizare portofoliu” a pornit de la o pasiune personală pentru fotografie și de la o necesitate pentru cunoașterea în detaliu a tehnicilor complexe necesare pentru realizarea unor fotografii corecte. Pe scurt, aplicația a fost creată pentru a oferii o platformă online unde fotografii amatori și profesioniști pot să își expună lucrările și să le împărtășească publicului.

În plus, galeria oferă o oportunitate fotografilor de a-și dezvolta abilitățile utilizând detaliile extrase din spatele fiecărei fotografii expuse în aplicație, astfel își pot perfecționa munca. Utilizând această idee, fotografii sunt îndemnați automat la aprofundarea cunoștințelor legate de tehnica fotografiei și la realizarea de imagini cât mai corecte și impresionante

Un ultim motiv, dar de o importanță la fel de mare ca celelalte, este interacțiunea între oameni pasionați de același domeniu. Astfel se păstrează o colectivitate unită prin intermediul căreia oricine poate să fie îndemnat să își imbunătățească tehnicile prin intemediul feedback-ului primit. Totodată fotografii pot colabora în mod facil, împărtășind experiențele. Această interacțiune poate să comduca la dezvoltarea unei comunități mari, active, unde membrii se pot suscine reciproc și se pot aprecia unii pe altții.

În concluzie, motivul pentru care aplicația a fost creeată este legat de pasiunea pentru fotografie și de dorința de a creea o colectivitate în intermediul căreia această pasiune poate fi împărtășita.

* 1. **Scop și obiective**

Scopul principal este de a creea un mediu online unde fotografii profesioniști sau amatori pot să își împărtășească creațiile cu publicul interesat de acest domeniu. Aplicația este în principal destinată publicului care deține cunoștințe legate de fotografie deoarece dispune de noțiuni tehnice care necesită documentare pentru a fi înțelese.

În același timp, aplicația poate fi utilizată și de persoane care nu au cunoștințe de fotografie, dar care doresc să admire fotografiile uimitoare realizate de fotografi din toată lumea. Pentru aceștia aplicația va servii ca o galerie simplă dar plină de imagini spectaculoase.

Pentru îndeplinirea acestui scop, galeria online își propune să îndeplinească următoarele obiective:

* Determinarea tehnologiilor care ne permit implementarea acestei aplicații
* Descrierea aplicației
* Implementarea aplicației care are la bază două obiective principale:
* Să pună la dispoziție un mediu ușor și sigur unde să poată încărca și partaja imagini
* Să puna la dispoziție opțiuni diverse de organizare și sortare a fotografiilor pe baza anumitor criterii definite de noțiuni generale precum dată, locație dar și prin criterii specifice precum metadata-ul fiecarei fotografii
* Să pună la dispoziție un sistem de feedback pentru utlizatorii care încarcă fotografii în aplicație și le distribuie publicului
* Să pună la dispoziție o modalitate prin care utilizatorii pot distribuii imaginile din galeria privată pentru a fi vizualizate și analizate de ceilalți utilizatori
* Realizarea documentației
  1. **Realizări similare acestei teme**

Există o multitudine de platforme online care se axează pe acest domeniu și care oferă un mediu virtual pentru distribuirea fotografiilor personale ale utilizatorilor. O diferență majoră între acestea și aplicația “Aplicație de organizare portofoliu” este informația tehnică legată de parametrii utilizați la realizarea fotografiei și posibilitatea sortării imaginilor bazată pe acești parametrii.

Toate aplicațiile care vor fi prezentate în următoarele rânduri permit sortarea și filtrarea imaginilor, dar nu cu ajutorul a parametrilor tehnici din metadata-ul fotografiilor.

Câteva dintre exemplele cele mai cunoscute sunt:

* Flickr[1]
* Instagram[2]
* DevianArt[3]

1. **Teorie și tehnologii utilizate**
   1. **Visual Studio Code**

Visual Studio Code este un editor text dezvoltat de Microsoft pentru diferite platforme precum Windows, macOS și Linux. Aplicația a ajuns foarte populară datorită diferitelor funcționalități puse la dispoziție utilizatorului precum modulul de debugging, prin completarea inteligentă a codului, funcție cunoscută și sub denumirea de autocompletare a codului, prin compatibilitatea sa cu Git și nu în ultimul rând prin multitudinea de extensii disponibile în propria galerie din aplicație.  
Din punct de vedere al limbajelor suportate, Visual Studio Code poate fi folosit cu o multitudine de limbaje de programare precum C/C#/C++/Java/JavaScript/Node.js/Python precum și multe altele.

VS Code a fost prima dată pus la dispoziția publicului in anul 2015 de către Microsoft. Datorită compatibilității dezvoltate cu github, codul sursă al aplicației a fost pus disponibil pe GitHub câteva luni mai târziu, iar una dintre cele mai importante funcționalități ale aplicației, anume extensiile, au fost puse la dispoziție un an mai târziu dupa lansare, în 2016.

La cațiva ani de la debutul impresionant al aplicației, în anul 2021, în colaborare cu GitHub, în VS Code este introdus GitHub Copilot. Acest tool reprezintă un pas foarte important deoarece devine o funcționalitate utilizată la nivel mondial cu beneficii semnificative pentru dezvoltatori datorită suportului impresionant oferit. GitHub Copilot este descris ca fiind un tool bazat pe AI care oferă sugestii de autocompletare a codului pentru programator. Acesta reprezintă un model antrenat pe toate limbajele de programare care apar în bazele de date ale companiei. Având în vedere dimensiunea informațiilor disponibile pe GitHub, modelul este antrenat la un nivel foarte înalt. Totuși, calitatea sugestiilor oferite poate să difere deoarece unele limbaje de programare sunt mai puțin utilizate decât altele și astfel rezulta mai puține informații disponibile pentru model.

În anul 2022, VS Code a fost nominalizat ca fiind cel mai popular IDE. Ținând cont că există aproximativ 70000 de alte tool-uri similare, această nominalizare este una foarte importantă și pune în evidență versatilitatea aplicației.

Avantaje față de alte IDE-uri:

* Varietatea extensiilor oferite
* Gratuit și open-source
* Nivelul la care poate fi personalizată aplicația
* Ușor de utilizat
* Debugging al codului direct în aplicație
* Comunitatea activă
  1. **Arhitectura MERN**

MERN reprezintă o combinație de tehnologii utilizate pentru dezvoltarea aplicațiilor WEB. Reprezintă un acronim pentru MongoDB, Express.js, React și Node.js. Arhitectura MERN ajută dezvoltatorii sa construiască o arhitectură pe trei nivele(front end, back end, database), utilizând JavaScript și JSON.

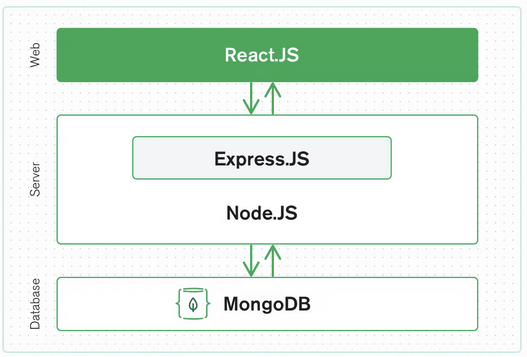


Figura 2.1 – Diagramă MERN Stack

În mare, MERN reprezintă o soluție full-stack pentru dezvoltatori datorită arhitecturii. Arhitectura MERN este utilizată deoarece crează o legatură simplă între cele trei niveluri de zevoltare, front end, back end și baza de date. Astfel, baza de date stochează date JSON și funcționează deosebit de bine cu Node.js făcând stocarea, manipularea și reprezentarea datelor JSON extrem de versatilă la fiecare nivel de dezvoltare al aplicației. React.js și Express.js care rulează pe Node.js contribuie la completarea aplicațiilor JavaScript finalizând noțiunea de full-stack. Express.js reprezintă un framework care rulează în paralel cu serverul, bazându-se pe cereri și răspunsuri HTTP, iar React.js este un framework front end în JavaScript utilizat la creearea interfețelor pentru utilizator.

* + 1. **MongoDB**

MongoDB reprezintă unul dintre cele 3 niveluri ale arhitecturii MERN. Este o platformă care pune la dispoziție servicii și unelte necesare dezvoltării rapide a aplicațiilor, la nivelul și performanța cerută de utilizatori. A fost fondată în anul 2007 și funcționează pe principiul stocării datelor diferit de bazele de date SQL. Astfel în loc să stocheze date în tebele construite din rânduri și coloane, fiecare înregistrare într-o bază de date MongoDB reprezintă un document BSON, pe scurt o reprezentare binară a datelor. Ulterior aplicațiile pot primii aceste documente într-un format JSON. Reprezintă o modalitate foarte flexibilă oferind variații în structura documentelor, astfel un document poate avea alte documente în interiorul acestuia.

Motivul pentru care utilizatorii aleg să utilizeze MongoDB în schimbul unei baze de date SQL este dat de formatul utilizat, și anume JSON. Acest format mapează obiecte native în majoritatea limbajelor moderne de programare deci devine automat o alegere naturală pentru majoritatea utilizatorilor.

MongoDB a fost dezvoltată pentru utilizatori care construiesc aplicații web și oferă modalități foarte simple de implementare, conectare și utilizabilitate.

* + 1. **Express.js**

Express.js reprezintă un framework pentru dezvoltarea aplicațiilor împreună cu Node.js pus la dispoziția publicului din anul 2010. Încă de la început a fost pusă la dispoziție ca un software open-source, desemnat dezvoltării aplicațiilor web și a API-urilor, fiind nominalizat ca standard framework pentru Node.js.

Cele trei puncte cheie pe care acest framework se bazează sunt

* Performață maximă
* Utilizabilitatea HTTP
* Routarea
  + 1. **React**

React reprezintă o modalitate simplă și interactivă pentru crearea interfețelor utilizatorilor. React are la bază construirea interfețelor reutilizabile și modulare, folisind o sintaxă specifica numită JSX. Aceasta combină HTML și JavaScript. Utilizând React utilizatorii pot definii starea componentelor, astfel actualizările de stare au loc automat, astfel modificările aduse aplicației sunt construite automate în interfață. Fiint folosit împreună cu alte api-uri precum React Router, această bibliotecă ajută la o dezvoltare fluidă și ușoară din punct de vedere tehnic al aplicațiilor web.

Exemplu de utilizare a unei componente React implementată în profilePage/index.jsx:

import { Box } from "@mui/material";

import { React } from "react";

const ProfilePage = () => {

return (

<Box>

</Box>

);

};

export default ProfilePage;

* + 1. **Node.js**

Node.js este definit ca un mediu de execuție back end JavaScript utilizat pentru executarea de cod JavaScript în afara unui browser web. Acest tool pune la dispoziția utilizatorilou un mediu în care aceștia pot rula comenzi JavaScript pentru generarea uni conținut dinamic în paginile web.

Rolul principal al acestei componente este de a constuii serverul API care asigură comunicare între acesta, partea de front-end a aplicației și baza de date MongoDB. Principala ocupație a acestei componente este gestionarea cererilor HTTP primite de la client sau a datelor trimise de la baza de date afiliată. Un alt domeniu important în care node.js figurează reprezintă autentificarea si autorizarea. Aici Node.js are rolul de a verifica aceste date și de a proteja datele.

Reprezentând o componentă foarte importantă în cadrul arhitecturii MERN, fiind utilizată pentru gestionarea în mod eficient a cererilor multiple specifice aplicațiilor web, Node.js este esențială în dezvoltarea acestor aplicații, fiind o platformă solidă.

În esență, Node.js asigură o comunicare cât se poate de fluentă între clientul aplicației, server și baza de date MongoDB.

* 1. **API-uri**
     1. **React Router și Express API**

React Router este o funcționalitate a Express ce ofera posibilitatea navigarii la nivel de client în cadrul aplicației. În aplicațiile web tradiționale, browser-ul realizează o cerere pentru un document dintr-un server web, descarcă datele și le evaluează, iar procesul este repetat de fiecare dată când un utilizator apasă pe un link.

Rutarea la nivel de client are la bază modificarea URL-ului curent fără realizarea altei cereri către serverul aplicației. Această funcționalitate este importantă deoarece aplicația poate afișa altă interfață utilizatorului într-un mod rapid și eficient, utilizând funcția fetch pentru a modifica pagina cu informațiile noi aduse. Astfel se dezvoltă o experiență mult mai rapidă pentru utilizator deoarece browser-ul nu este nevoit să re-evalueze documente JavaScript și CSS de fiecare dată când un nou link este apăsat, sau o nouă interfață trebuie afișată. Routarea la nivel de client este inițializată prin utilizarea tag-ului Router.

Exemplu de implementare a unei componente React Router în cadrul serverului:

function App() {

  return (

    <div *className*="app">

      <BrowserRouter>

          <Routes>

            <Route

*path*="/"

*element*={<LoginPage />}

/>

            <Route

*path*="/home"

*element*={<HomePage />}

            />

            <Route

*path*="/profile/:userId"

*element*={<ProfilePage />}

            />

          </Routes>

      </BrowserRouter>

    </div>

  );

}

export default App;

Exemplu de utilizare a componentei React utilizând express la nivel server:

import express from "express";

import { loginFunc } from "../controllers/auth.js";

const router = express.Router();

router.post("/login", loginFunc);

export default router;

Exemplu de utilizare a componentei React utilizând express la nivel client:

 const login = async (*values*, *onSubmitProps*) => {

    const loggedInResponse = await fetch("http://localhost:3001/auth/login", {

      method: "POST",

      headers: { "Content-Type": "application/json" },

      body: JSON.stringify(*values*),

    });

    const loggedIn = await loggedInResponse.json();

*onSubmitProps*.resetForm();

    if (loggedIn) {

      navigate("/home");

      );

    }

  };

* + 1. **Hooks**

React Hooks reprezintă o modalitate prin care dezvoltatorii pot folosii starea aplicației în diferite componente. În termeni simplii, React Hooks sunt funcțtii JavaScript capabile să izoleze părți reutilizabile dintr-o componentă funcțională, în acelasi timp avânt capabilitatea de a gestiona efecte secundare.

React pune la dispoziție un număr variat de hooks, precum:

* useState: Acest hook este utilizat pentru a gestiona statusul unei componente. Returnează o valoare de stare de tip boolean, adevărat sau fals, și o funcție utilizată pentru a modifica starea acestei variabile.
* useEffect: Acest hook gestionează efectele secundare ale aplicației precum apelurile API, timere și multe altele.

Exemplu utilizare useState:

import { useState } from "react";

const [checkSomething, setcheckSomething] = useState(0);

Exemplu utilizare useEffect:

import { useEffect} from "react";

useEffect(() => {

    if (checkSomething))) {

     setcheckSomething(false);

    } else {

    }

  }, [checkSomething]);

* + 1. **Mongoose**
    2. **Multer**
    3. **JWT**
    4. **EXIF**
    5. **Adauga si restu api urilor aici**

1. **Realizarea aplicației**
   1. **Mediul de dezvoltare**
   2. **Dezvoltarea propriu-zisă a aplicației**
      1. **Register/Login**
      2. **Pagina Social Media**
      3. **Pagina Galerie**

* + - 1. **Încărcare imagini**
      2. **Extragerea metadata din imagini folosind exif api**
      3. **Filtrare utilizând metadata**
      4. **Sortare**
      5. **Creearea modurilor de vizualizare grid a galeriei**
    1. **Funcții aplicabile pe imaginea postată**
    2. **Modul rapoarte**

1. **Concluzii și dezvoltări ulterioare**
   1. **Concluzii**
   2. **Dezvoltări ulterioare**
2. **Bibliografie**
   1. **Webliografie**

* [1]<https://www.flickr.com/>
* [2]<https://www.instagram.com/>
* [3]<https://www.deviantart.com/>