Aplicatie de gestionare a evenimentelor de tip workshop/prezentare in domeniul IT

PROIECT DE DIPLOMĂ

Autor: **Tămaș Bogdan David**

Conducător științific: **Titlu.ing. Dan Radu**

|  |  |
| --- | --- |
| DECAN  **Prof.dr.ing. Liviu MICLEA** | Vizat,  DIRECTOR DEPARTAMENT AUTOMATICĂ  **Prof.dr.ing. Honoriu VĂLEAN** |

Autor: **Tămaș Bogdan David**

Aplicatie de gestionare a evenimentelor de tip workshop/prezentare in domeniul IT

1. **Enunţul temei:** *Aplicatie de gestiune a unor evenimente in domeniul IT care sa permita unui utilizator anumite operatii de acces si de manipularea a unor evenimente.*
2. **Conţinutul proiectului:** *Pagina de prezentare, Declarație privind autenticitatea proiectului, Sinteza proiectului, Cuprins, Titlul capitolului 1, Titlul capitolului 2,… Titlul capitolului n, Bibliografie.*
3. **Locul documentaţiei:** *Universitatea Tehnică din Cluj-Napoca*
4. **Consultanţi:** *ing. Dan Radu (dacă este cazul)*
5. **Data emiterii temei:** 01.04.2020
6. **Data predării:**

Semnătura autorului

Semnătura conducătorului științific

**Declaraţie pe proprie răspundere privind**

**autenticitatea proiectului de diplomă**

Subsemnatul(a)  **Tămaș Bogdan David**, legitimat(ă) cu CI/BI seria MM

Nr. 801902, CNP 1970727244219, autorul lucrării:

Aplicatie de gestionare a evenimentelor de tip workshop/prezentare in domeniul IT elaborată în vederea susţinerii examenului de finalizare a studiilor de licență la **Facultatea de Automatică și Calculatoare**, specializarea **Automatică și Informatică Aplicată,** din cadrul Universităţii Tehnice din Cluj-Napoca, sesiunea Choose an item. a anului universitar 2019-2020, declar pe proprie răspundere, că această lucrare este rezultatul propriei activităţi intelectuale, pe baza cercetărilor mele şi pe baza informaţiilor obţinute din surse care au fost citate, în textul lucrării, şi în bibliografie.

Declar, că această lucrare nu conţine porţiuni plagiate, iar sursele bibliografice au fost folosite cu respectarea legislaţiei române şi a convenţiilor internaţionale privind drepturile de autor.

Declar, de asemenea, că această lucrare nu a mai fost prezentată în faţa unei alte comisii de examen de licenţă.

In cazul constatării ulterioare a unor declaraţii false, voi suporta sancţiunile administrative, respectiv, *anularea examenului de licenţă*.

Data Prenume NUME

(semnătura)

**SINTEZA**

proiectului de diplomă cu titlul:

Aplicatie de gestionare a evenimentelor de tip workshop/prezentare in domeniul IT

Autor: **Tămaș Bogdan David**

Conducător științific: **Titlu.ing. Dan Radu**

1. Cerinţele temei: Aplicația trebuie să permită înregistrarea și autentificarea unui utilizator într-o platforma care stochează evenimentele anterioare și ulterioare în doua interfețe diferite. Evenimentele la care utilizatorul s-a înscris vor apărea într-o a treia interfață. Se vor crea și alte interfețe care să permită utilizatorului următoarele acțiuni: propunerea unui topic, votarea oricărui topic existent și prioritizarea acestuia, acordarea de feedback evenimentelor anterioare într-un mod interactiv, accesarea unei pagini de instrucțiuni, accesarea unui profil personal, accesul la topul general precum și la o evidenta a rezultatelor acumulate.

2. Soluţii alese: Pentru a controla volumul mare de date acumulat după popularea aplicației am folosit o baza de date principala (MySql) și una secundara(Json-Server). Comunicarea cu baza de date se face direct (NodeJS), iar design-ul (ReactJS) este implementat folosind preponderent librăria material-ui.

3. Rezultate obţinute: Rezultatele obținute sunt mulțumitoare, iar aplicația are o interfață interactiva cu utilizatorul respectând restricțiile și performantele impuse.

4. Testări şi verificări: Testare s-a realizat prin mai multe verificări manuale a performantelor și a răspunsurilor primite de la baza de date precum și direct de la cod. Expresivitatea codului s-a produs prin mai multe procese de refactorizare și s-a verificat prin folosirea bibliotecii eslint.

5. Contribuţii personale: Pre-analiza și documentarea necesara, implementarea propriu-zisa a funcționalității și a design-ului aplicației, configurarea și integrarea bazelor de date, modificare aferente necesare pentru a asigura un cod cât mai explicit. 6.

Surse de documentare: Algoritmi de implementare eficienta, algoritmi de comunicarea rapida cu baza de date, site-uri web pentru librarii de design, tehnici de criptare, tutoriale de integrare a stilurilor, tehnici de refactorizare.

Semnătura autorului

Semnătura conducătorului științific

Cuprins

[1 Introducere 2](#_Toc477457095)

[1.1 Context general 2](#_Toc477457096)

[1.2 Obiective 2](#_Toc477457097)

[1.3 Specificații 2](#_Toc477457098)

[2 Studiu bibliografic 3](#_Toc477457099)

[3 Analiză, proiectare, implementare 4](#_Toc477457100)

[4 Concluzii 5](#_Toc477457101)

[4.1 Rezultate obținute 5](#_Toc477457102)

[4.2 Direcții de dezvoltare 5](#_Toc477457103)

[5 Reguli de formatare 6](#_Toc477457104)

[5.1 Formatarea paginii 6](#_Toc477457105)

[5.2 Titluri și stiluri 6](#_Toc477457106)

[5.3 Figuri, tabele și ecuații 7](#_Toc477457107)

[5.3.1 Figuri 7](#_Toc477457108)

[5.4 Tabele 7](#_Toc477457109)

[5.5 Ecuații 7](#_Toc477457110)

[5.6 Referințe bibliografice 8](#_Toc477457111)

[6 Bibliografie 9](#_Toc477457112)

# Introducere

## Context general

În ultimii ani în domeniul IT s-a înregistrat o evoluție continua cu o adaptabilitate pe schimbare axata pe informații și tehnologii cât mai recente ce a determinat o explozie din punct de vedere atât a calității, cât și a cantității de produse. Acest domeniu presupune o viață proactiva în care se dorește o dezvoltare permanenta a produsului și o evoluție a acestuia conform restricțiilor impuse de piață și de client.

Evoluția acestui domeniu a ajutat și la o creștere considerabila a numărului de angajați deoarece aceștia au o mulțime de oportunități de dezvoltare personala și profesionala.

Numărul evenimentelor și conferințelor din domeniul tehnic este într-o continua creștere ceea ce sugerează o atmosfera progresiva a interesului către acest domeniu. Deoarece acest progres a avut un ritm atât de alert în ultimii ani a dus la o peletizare de subdomenii care-i oferă participantului oportunitarea de a se îndreptă înspre orice tehnica dorește s-o abordeze. Odată cu creșterea numărului de evenimente s-a dorit și o organizare cât mai buna a acestora și o evidenta cât mai divizata.

Aplicația creata are un scop extrem de practic și efectiv în a ușură păstrarea evidentei și a ajuta la o planificare cât mai facila a evenimentelor. Obiectivul principal este acela de-a crea o interfață care să ilustreze un mediu familiar pentru utilizator ținând totodată cont de preferințele acestuia.

Acordarea de review precum și sortarea evenimentelor și reținerea acestora într-un loc prestabilit se întâmplă de obicei pe mai multe flancuri în aplicațiile generale ceea ce poate genera confuzie. Acest proiect dorește să ajute utilizatorul să treacă prin aceste etape într-un mod mult mai plăcut și rapid.

**// Prezentare pe capitole**

## 

## Obiective

Obiectivul principal al acestei aplicații este de a crește rata productivității și a vitezei de organizare în ceea ce privește răspândirea informațiilor în domeniul IT și de a determina oamenii din acest domeniu să participe la cât mai multe evenimente.

Participarea la evenimentele/workshop-urile angajatorilor din domeniu aduce cu ea mai multe oportunități:

- șansele de a găsi un job cât mai rapid cresc

- afli beneficiile oferite de fiecare firmă

- identificarea soluțiilor

- gratuitate

- diversitatea temelor prezentate

De asemenea, votarea fiecărui topic care ar putea să fie prezentat într-un workshop/prezentare viitoare de către orice utilizator ajută organizatorii să aleagă o temă cât mai utilă și care să stârnească interesul cât mai rapid.

De ce această temă?

Consider că acest domeniu oferă multe oportunități de a învăță și a rămâne în temă cu tot ce este de actualitate, iar dezvoltarea unei platforme care să ajute la o mai mare popularitate a evenimentelor este un lucru vital. Varietate temelor oferă de asemenea o mai mare maleabilitate în ceea ce privește orientarea fiecărui individ.

## Specificații

În aplicația rezultată se dorește obținerea unei interfețe friendly care să ajute utilizatorul să se acomodeze cât mai rapid cu o arhitectură software maleabilă care să manipuleze date într-un volum mediu cât mai agil.

Specificațiile/funcționalitatea/performanțele/securitatea trebuie să respecte o serie de constrângeri:

- câmpurile cu datele confidențiale ale fiecărui utilizator nu vor fi vizible

- datele confidențiale vor fi trimise codat către bază de date

- accesul la rutele private se va face pe o perioadă limitată de timp

- call-urile către bază de date (principală/ secundară) trebuie să aibă o duarată medie mai mică de 4 secunde, iar acest proces va fi mascat de fiecare dată de un buffer

- token-ul accesului principal va fi implementat printr-un algoritm de generare cu un anumit pattern

- existența unei pagini de informare a unui utilizator este impusă, iar aceasta va putea fi accesată de pe o rută publică

- la procesul de înregistrare al unui user va exista o serie de mesaje și de validări care să poată să-l ghidez

- top-urile și listele vor fi ilustrative și cât mai comprehensibile pentru user implicând un design cât mai transparent

- accesul la profilul principal se va putea efectua de pe orice rută privată

- elementele din liste vor fi afișate printr-un algoritm care să limiteze o recurență fără a le afișa exhaustiv de la început

- user-ul va avea control deplin asupra evenimentelor personale, dar nu va putea să influențeze deciziile celorlalți

În aplicația rezultată se dorește obținerea unei interfețe friendly care să ajute utilizatorul să se acomodeze cât mai rapid cu o arhitectură software maleabilă care să manipuleze date într-unvolum mediu cât mai agil.

# Studiu bibliografic

Functionalitatea aplicatiei este divizata dupa urmatoarea structura:

- **Front-end**:

- HTML

- CSS

- Javascript

Implementarea efectiva a design-ului si functionalitatii din fata s-a realizat ajutorul librariei **ReactJS** care constituie o comprimare a tehnologiilor de mai sus intr-o maniera mai organizata pentru construirea interfetelor cu utilizatorul.

- **Back-end**:

- NodeJS

Comunicarea cu baza de date principale s-a realizat prin **NodeJS** printr-o maniera simpla si efectiva folosin sintaxa de JS si comenzi de SQL.

- **Baze de date**:

- MySQL

- JSON Server

- **Platforma de control:**

**-** Github

## Front-end

**2.1.1. HTML**

HyperText Markup Language reprezinta limbajul de marcare a componentelor in paginile WEB. Prin HTML se asigura organizarea prezentarii informatiilor, ierarhia lor si pozitia in pagina.

Extensia “.html” sau “.htm” definesc fisierele de tip html, iar citirea si modificarea acestora nu necesita un editor sau mediu de programare specific.

Componentele dintr-o pagina html sunt definite de anumite atribute, iar tipul lor este ilustrat printr-un tag specific.

Integrarea algoritmilor implementati in JS si a design-ului creat in CSS se face in general in HTML, iar pentru a evita integrarea manuala se poate apela la diverse librarii ca si jQuery, Vue.js, ReactJS…

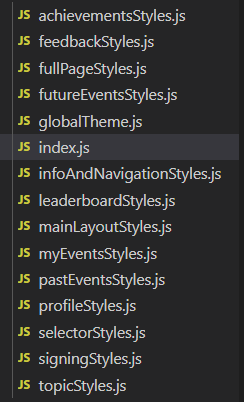
**2.1.2. CSS**

Cascading Style Sheets este un standard de formatare a design-ului unor componente care ajuta la crearea unor interfete personalizate.

Modul in care am implementat structura de design de tip CSS este unul direct, folosind libraria material-ui si apeland la metoda “withStyles” am putut sa creez un design integrat in fisierul de tip JS.



Style-urile tuturor componentelor s-au stocat intr-un folder principal si s-au exportat printr-un index.



Asadar, datorita maleabilitatii oferite de aceasta librarie stylurile au fost definite ca si obiecte constante.

**2.1.3. Javascript**

Javascript este un limbaj de programare orientat pe obiect care se foloseste mai ales la introducerea functionalitatilor intr-o pagina web/ aplicatie, dar si pentru accesul la obiecte încapsulate in diferite aplicatii. Deoarece acest limbaj este bazat pe mai multe concepte de prototipuri acesta are o comunicare mai practica in ceea ce priveste structura paginilor web ingloband paginile de tip HTML cu usurinta si avand o structura mult mai ductila.

Design-ul principal si cea mai mare parte a functionalitatii aplicatiei a fost implementata in framework-ul React deoarece consider ca are o sintaxa foarte usor de inteles si multe tehnici de organizare a codului intr-un mod efectiv.

Interfata este dominata de elemente care fac parte din libraria **material-ui**. Am apelat la aceasta librarie deoarece consider ca are o structura eleganta si dinamica a componentelor, iar stilurile principale nu necesita existenta unor fisier de tip CSS.



Dependintele/libariile folosite exclusiv in React pentru implementarea aplicatiei sunt urmatoarele:

- design & animatii:

- material-ui-pickers

- @materia-ui/core

- @material-ui/icons

- @material-ui/lab

- react-animated-modal

- functionalitate:

- moment

- react-datepicker

- react-dom,

- react-duration-picker

- react-form-validator-core

- react-hot-loader

- react-material-ui-form-validator

- react-router-dom

- react-spring

- react-transition-group

- react-scripts

- validator

- securitate:

- react-cookie

- react-password-strength

- js-cookie

- universal-cookie

-comunicare cu baza de date:

- axios

- json-server

## Back-end

**2.2.1 NodeJS**

NodeJS este un mediu rulare a codului Javascript in afara unui browser Web. Acesta se foloseste in preponderenta pentru implementariile de tip back-end si nu necesita o interfata grafica pentru a-si indeplini scopul.

In aplicatia creata am folosit NodeJS pentru a realiza comunicarea cu baza de date manipuland actiunile de tip CRUD si comenzile de SQL printr-un Router pe care l-am extras din libraria express.



## Baze de date

**2.3.1. MySQL**

MySQL este un mediu folosit pentru gestiunea bazelor de date de tip relational folosind tabele.

Datele sunt stocate si manipulate prin comenzi specifice ca si:

- SELECT \* FROM [nume\_tabel] – actiunea de selectare dintr-o tabela

- INSERT INTO [nume\_tabel] – inserarea in tabela

- UPDATE [nume\_tabel] SET [campuri] – modificare a campurilor dintr-o tabela

- DELETE FROM [numa\_tabel] WHERE [conditie] –stergerea unui element dintr-o tabela.

Se impune datelor retinute in tabele sa aiba anumite campuri definitorii printre care si un camp primar diferit de celelalte elemente.

Pentru lansarea bazei de date MySQL am folosit pachetul XAMPP deoarece acesta ofera o permisivitate mare asupra bazei de date si este actualizat in permanenta.

Acesta ne ofera si acces la log-urile comenzilor efectuate in baza de date pentru a-o putea monitoriza continuu.



Baza de date are legatura directa cu NodeJS, iar rularea acesteia se face local pe portul 8080 utilizand comanda:



Daca baza de date este configurata bine terminalul va afisa: C:\Users\bogdan.tamas\AppData\Local\Microsoft\Windows\INetCache\Content.Word\server_listening.png

**2.3.2. JSON-Server**

Json-Sever este un modul inclus in pachetul npm care permite crearea unui server local de tip REST similar unei baze de date.

Creare serverului local este una relativ simpla folosind comanda ‘json-server’ dupa cum urmeaza:



Comanda de mai sus va crea un server local in care operatiile de CRUD se vor executa asupra fisierului appDatabase.json folosin portul 3001.

Am ales utilizarea acestui tip de server deoarece controlul asupra datelor este accesibil, iar operarea datelor de tip array/object este mult mai practica.

## Platforma de control

**2.4.1. Github**

Github este o platforma de control care ajuta la pastrarea evidentei codului si la monitorizarea acestuia in permanenta. Este des folosit in dezvoltarea proiectelor deoarece previne pierderea datelor si sustine tehnici de combinare a codului intr-un mod cat mai sigur.

Proiectul principal este impartit in mai multe branch-uri care servesc un scop unic. De regula, exista mai multe strategii de branching, iar in functie de aceasta se impune un anumit mod de munca si un anumit tipar al review-ului.

Utilizatorul isi inregistreaza modificarile pe un anumit branch prin comanda “git commit –m [mesaj]”. Mesajul trebuie sa fie unul explicit pentru ceilalti utilizatori si sa respecte structura “type: scope”.

Principala calitatea a platformelor de control este prezenta preponderenta a review-ului pe cod ori de cate ori se realizeaza combinarea branch-urilor. Desigur ca prezenta review-ului este uneori optionala, dar pentru o mai buna practica se recomanda utilizarea acesteia in orice imprejurime.

Cele mai comune comenzi pentru controlul software in github sunt:

- git add – adauga modificarile in zona de asamblare

- git commit – comanda urmata de un mesaj care va constitui eticheta modificarilor adaugate in zona de asamblare

- git push - local -> repository

- git pull – repository -> local

- git branch [nume\_branch] – crearea unui nou branch

Pentru o clarificare mai buna a ceea ce inseamna un branch si cum functioneaza comenzile de mai sus se poate folosi comanda “git gui” care va deschide o fereastra de control a modificarilor existente.

# Analiză, proiectare, implementare

**3.1. Scop, stil si integrarea datelor**

Asa cum am mentionat mai sus, scopul principal al aplicatiei este de-a eficientiza lucrul cu evenimente si de-a le gestiona printr-o structura organizatorica implementata pe baza unui design predefinit.

Design-ul este din punct de vedere analitic unul bazat pe animatii interactive care sa impiedice cursul aplicatiei sa devina unul monoton. Style-ul general este unul elegant fara elemente pozitionate sau modificate in mod excesiv. Fiecare style al unei componente aferente este retinut in folderul principal al style-urilor, iar obiectul fiecarui style este de forma :



Dupa ce obiectul principal de style este importat in componenta se va integra direct in elementele structurii prin metoda impusa de libraria « withStyles » :



Acum obiectele de style vor fi incluse in obiectul principal « classes » care este integrat in props prin metoda specificata mai sus.

Pentru implementarea propriu-zisa a functionalitatii am folosit diversi algoritmi de implementare care sa controleze datele manipulate atat din punct de vedere logic cat si analitic. Algoritmii principali sunt relativ simpli din punct de vedere matematic si se bazeaza dominant pe tehnici de filtrare a datelor apeland la diverse functii predefinite de manipulare a sirurilor simple si sirurilor de obiecte. Metodele principale folosite sunt :

- filter : filtreaza datele dupa o anumita conditie

- map : crearea unui sir nou rezultat din cel pe care se apeleaza metoda aplicand functia impusa

- reduce : valoriile care nu respecta conditia mentionata in functia reduce sunt eliminate din sirul principal

- sort : pozitia fiecarui element din sir se va modifica in functie de conditia principala rezultand un nou sir sortat iterativ

Deoarece datele retinute in Json-Server sunt suportate si intr-un volum mai mare este necesara filtrarea acestora inainte de executia efectiva a operatiilor pe acestea.

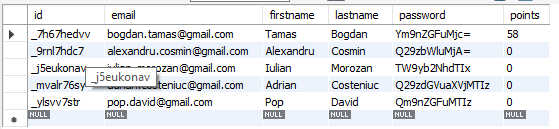
**3.2. Structura si integrare baza de date.**

**3.2.1. MySQL**

Baza principala de date presupune un singur tabel care are functia de a stoca datele de securitate si de acces.

Configurarea bazei de date s-a realizat dupa instalarea aplicatiei MySql Workbench 8.0 si XAMPP prin etapa de initializare a unei baze de date si a unui tabel incorporat si etapa de controlare a acestie prin aplicatia XAMPP.

Tabelul principal de administrare a securitatii aplicatie are urmatoarea forma:



Utilitatea fiecarui camp este urmatoarea:

* **id** : cheia principala a tabelului care retine valoare ce permite accesul utilizatorului in aplicatie. Valoare fiecarui camp respectiv este generata in NodeJS prin functia urmatoare :



* **email :** valoarea campului email din cardul procesului de inregistrate a fiecarui utilizator. Valoarea este unica fiecarui camp, iar existenta unui camp duplicat este impiedicata de functionalitatea implementata.
* **firstname & lastname :** valoarea campurilor firstname si lastname din stocate in procesul de inregistrare
* **password :** valoarea campului password & passwordConfirmation convertita in baza 64.
* **points :** valoarea ce determina pozitia user-ului in top este configurata si actualizata la fiecare logare in aplicatie si este sincronizata cu baza secundara.

Comenzile CRUD ale bazei de date sunt manipulate in NodeJS prin componenta Router dupa urmatoarea forma generala :

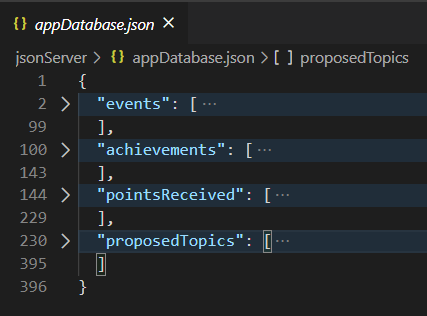




**3.2.2. Json-Server**

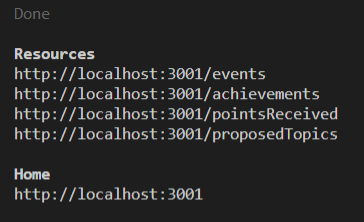
Baza secundara de date presupune un fisier de tip JSON ce are in componenta sa mai multe array-uri de obiecte complexe unde se retin date cu campuri unice.

Continutul bazei de date este urmatorul:



Fiecare dintre aceste campuri principale au mai multe campuri secundare care se acceseaza prin parcurgere in adancime sau prin destructurare succesiva.

Dupa comanda de lansare a bazei de date pe portul 3001 in terminal se va afisa urmatorul rezultat :



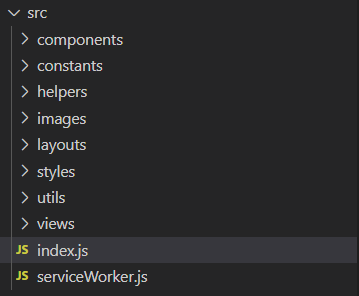
Cele 4 rute de mai sus vor fi definite de array-urile retinute in JSON, iar modficarea datelor se va face direct pe ruta dorita.

**3.3. Structura proiect**

Structura codului este una organizata care s-a definit prin mai multe procese de refactorizare realizate aferent implementarii.

Aplicatia este alcatuita din 3 foldere principale.

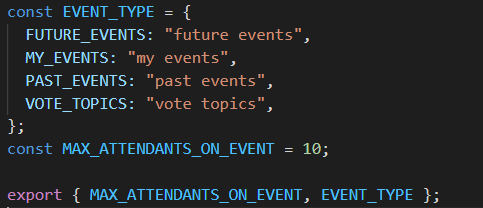
Front-end – **src**



Components – folderul ce are in compozitia sa elementele propriu-zise ale aplicatiei. Aici se integreaza stiulul in componente si se administreaza functionarea fiecareia. In compozitia acestuia avem mai multe subfoldere in care se respecta o structura de tipul list-item. Subfolderele care definesc componentele aplicatie sunt: achievements, feedback, fullPage, futureEvents, infoAndNavigation, leaderboard, myEvents, pastEvents, profile, signing, voteTopics. Accesul extern spre importarea acestora se face printr-un index.

Constants – folderul in care se retin datele folosite repetitiv in cadrul aplicatiei si care nu au un comportament dinamic sau sunt functii. Constantele sunt exportate din index-ul folderului si sunt impartite in: charLimitConstants, databaseConstants, eventsConstants, signingConstants, topicConstants.

Structura unui fisier de constante:



Helpers – retine functiile folosite repetitiv in cadrul aplicatiei.

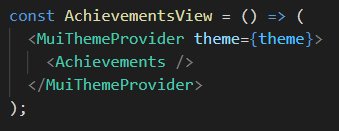
Images – imaginiile folosite in cadrul aplicatiei.

Layouts – aici se retin schemele principale : schema de inregistrare si schema generala.

Styles – aici sunt retinute style-urile.

Utils – in acest folder sunt retinute componentele secundare si cele exclusiv functionale care sunt folosite repetitiv de catre componentele principale. Componentele secundare sunt de tip Alerte, Selectori, Rute Private sau Tranzitii.

Views – aici sunt retinute containerele componentelor care au scopul de-a integra fiecare components in tema globala a aplicatiei.



index.js – aici se importa toate containerele aplicatiei si se atribuie fiecareia o anumita ruta specifica respectand structura urmatoare :

<Route path="/login" component={LoginView} />

<PrivateRoute path="/profile" component={ProfileView} />

Back-end – **api**

In acest folder avem urmatoarele componente :

* o functie de generare a id-ului fiecarui user cand se apeleaza metoda de insert
* fisierul de conexiune cu baza de date cu extensia « .sql »
* fisierul de rutare catre baza de date
* fisierul serverului in care se stabileste portul aferent serverului principal al bazei de date
* fisierul de creare a conexiunii sql

Database – **jsonSever**

Acest folder are in componenta sa un fisier de tip json care va stoca datele si un fisier de documentatie care sa explice modul de rulare si folosire al acesteia

**3.4. Implementare tehnica**

* + Detalii referitoare la analiză și proiectare:
    - descrierea metodelor pe care le-ați aplicat pentru rezolvarea problemei,
    - descrierea materialelor, procedurilor
    - calcule, tehnici, descrierea echipamentelor
    - metodologia de proiectare
    - informațiile necesare pentru ca cineva să poata reface lucrarea
  + Implementare :
    - Descrieti detaliile tehnice ale implementarii aplicatiei: mediul de implementare, modul de prezentare, modul de utilizare al aplicatiei, etc.
  + Testare si validare :
    - Descrieți metodologia de testare a aplicației și rezultatele
    - Includeți experimentele pe care le-ați realizat, analiza rezultatelor pe care le-ați obținut.

# Concluzii

## Rezultate obținute

Evidentiați toate rezultatele pe care le-ați obtinut și trageți concluzii din ele. Puteți prezenta o analiză critică a ceea ce ați realizat comparativ cu alte lucrări/studii anterioare.

Includeți o listă a contribuțiilor pe care le-ați avut în domeniul temei abordate.

## Direcții de dezvoltare

Descrieți direcțiile posibile de dezvoltare.

# Reguli de formatare

## Formatarea paginii

* + Dimensiunea paginii: A4
  + Margini: 2.5 cm (sus, jos, stânga, dreapta)
  + Antet și subsol: 1.27 cm de la marginea paginii
  + În antetul paginii (header): titlul capitolului, centrat, stil: Header\_style
  + În subsolul paginii: numărul paginii, centrat

## Titluri și stiluri

Titlurile capitolelor și subcapitolelor se marchează cu stilurile Heading 1 – 4, conform documentului model anexat în format Word. Descrierea stilurilor utilizate în document este prezentată în Tabelul 5.1.

Tabelul 5.1. Stiluri utilizate în acest document

| Nr. | Stil | Utilizat pentru | Format |
| --- | --- | --- | --- |
| 1 | Normal | Text normal | Font: (Default) Cambria, 12 pt, Justified, Line spacing: Multiple 1.1 li, Space After: 6 pt |
| 2 | Titlu | Titlul proiectului, prima pagină | Font: 24 pt, Small caps, Centered Line spacing: single, Space Before: 126pt, After: 0 pt, |
| 3 | Titlu2 | Titlul proiectului, pagina de prezentare | Font:14pt, Bold, Centered |
| 4 | Heading 1 | Titlurile capitolelor (nivel 1) | Font: 24 pt, Indent: Left: 0 cm Hanging: 0.76 cm, Space Before: 24pt, After: 12pt |
| 5 | Heading 2 | Titlurile subcapitolelor (nivel 2) | Font: 14 pt, Bold, Indent: Left: 0 cm  Hanging: 1.02 cm, Space Before: 18pt, After: 12pt |
| 6 | Heading 3 | Titlurile secțiunilor (nivel 3) | Font: Bold, Indent: Left: 0 cm Hanging: 1.27 cm, Space Before: 6 pt, After: 6pt |
| 7 | Heading 4 | Titlurile secțiunilor (nivel 4) | Font: Italic, Indent: Left: 0 cm Hanging: 1.52 cm, Space Before: 2 pt, After: 0 pt |
| 8 | Caption | Legenda figurilor și tabelelor | Font: Italic, Font color: Text 1, Line spacing: single, Space After: 10 pt, |
| 9 | Header\_style | Antetul paginii | Font: 10 pt, Italic, Centered, Border: Bottom: (Single solid line, Background 1, 0.5 pt Line width) |

## Figuri, tabele și ecuații

### Figuri

Figurile se inserează în text centrate, cu etichetă de numerotare și legendă (Caption) în partea de jos a figurii. Numărul figurii include și numărul capitolului, după exemplul prezentat în Figura 5.1.



Figura 5.1. Figură exemplu, stil: Caption

## Tabele

Tabelele se inserează în text centrate, cu etichetă și legendă (Caption) în partea de sus a tabelului, aliniată la stânga. Numărul tabelului include și numărul capitolului, după cum este prezentat, de exemplu, în Tabelul 5.1.

## Ecuații

Ecuațiile se inserează în text centrate, cu numerotare în partea dreaptă. Numărul ecuației include și numărul capitolului, conform exemplului din relația (5.1).

|  |  |
| --- | --- |
|  | (5.1) |

## Referințe bibliografice

Se recomandă ca citarea referințelor bibliografice să fie făcută în formatul IEEE.

În secțiunea Bibliografie sunt prezentate exemple pentru: o citare a unui capitol dintr-o carte [1], un articol publicat într-o revistă [2] și un articol publicat la o conferință [3].

Detalii cu privire la formatul citării diverselor tipuri de referințe pot fi găsite în [4] sau [5].

Referințele bibliografice se pot insera în text utilizând facilitățile Word de a adăuga surse și bibliografie unui document (References -> Citations & Bibliography). Dacă formatul IEEE pentru bibliografie nu este instalat implicit în Word, se poate descărca gratuit de la:

<https://bibword.codeplex.com/wikipage?title=Styles&referringTitle=Home>

Instrucțiunile de instalare pentru diferite versiuni de Word se pot obține de la aceeași adresă.

# Bibliografie

|  |  |
| --- | --- |
| [1] | P. Nume, "Titlul capitolului," in *Titlul cartii*, Oras, Editura, 2016, pp. 1-24. |
| [2] | P. Nume, "Titlul articolului," *Titlul revistei,* vol. 1, no. 2, pp. 22-30, 2016. |
| [3] | P. Nume, "Titlul articolului," in *Numele conferintei*, Oras, 2015. |
| [4] | "IEEE Citation Reference," 2009. [Online]. Available: https://www.ieee.org/documents/ieeecitationref.pdf. |
| [5] | "IEEE Editorial Style Manual," 2016. [Online]. Available: https://www.ieee.org/documents/style\_manual.pdf. |