**FindMyPet**

**Candidat: Laura-Nicoleta, Ilie**

**Coordonator științific: Conf.dr.ing. Loredana-Mihaela, Stanciu**

Sesiunea: Iulie 2025

**REZUMAT**

Lucrarea de față prezintă dezvoltarea unei aplicații web numită „FindMyPet”. Această aplicație web este destinată persoanelor care au pierdut sau au găsit animale de companie (câini sau pisici). Proiectul are ca scop principal facilitarea interacțiunii între utilizatori prin publicarea de anunțuri geo localizate, cu posibilitatea de a vizualiza locația pe hartă și de a contacta rapid proprietarii.

Aplicația este implementată utilizând tehnologii actuale precum React.js și Vite pentru partea de frontend și Firebase ca soluție de backend-as-a-service, oferind servicii integrate de autentificare, baze de date (Firestore) și stocare de fișiere (Storage). Integrarea hărții Google Maps permite afișarea dinamică a animalelor pierdute/găsite sub formă de , fiecare însoțit de detalii relevante și imagine.

Pentru trimiterea de mesaje prin intermediul formularului de contact, s-a utilizat serviciul EmailJS, care permite trimiterea de emailuri direct din aplicație fără un server propriu. Aplicația oferă o interfață simplă, scalabilă și intuitivă, adaptată pentru utilizatori fără cunoștințe tehnice.

Rezultatul final este o platformă funcțională, modernă, prietenoasă, plăcută vizual și scalabilă, care acoperă toate cerințele unui sistem de gestionare a anunțurilor pentru animale pierdute/găsite. Lucrarea subliniază avantajele folosirii unei arhitecturi moderne și ușor de extins, precum și potențialul viitor de dezvoltare prin adăugarea de funcționalități precum notificări sau editarea anunțurilor.

**ABSTRACT**

The current paper presents the development of a modern web application called FindMyPet, created to assist individuals who have lost or found companion animals. The goal of the platform is to simplify the process of posting and finding lost/found pet announcements through a geolocated and easy-to-use interface. Users can view existing announcements directly on an interactive map and, if authenticated, can add their own, including images, details, and the last known location of the animal.

The application is built using React.js and Vite for the frontend, and Firebase as a Backend-as-a-Service (BaaS) solution. Firebase provides essential services such as user authentication, a NoSQL database (Firestore), and file storage. For mapping functionalities, Google Maps API is integrated, allowing the display of announcements as map markers. Additionally, EmailJS is used to enable sending messages through a contact form, without a dedicated backend server.

FindMyPet was developed in iterative steps, each bringing new features. The result is a scalable, responsive, and intuitive platform that can serve as a centralized solution for pet-related announcements. This application stands out through its simplicity, practical utility, and potential for future expansion with features such as notifications, user dashboards, or advanced filtering.

# INTRODUCERE

## Contextul

Animalele de companie sunt considerate membri ai familiei, iar pierderea acestora reprezintă o experiență greu de gestionat și cu impact emoțional major. În astfel de situații, timpul este un factor esențial pentru recuperarea animalului. În ultimii ani metodele de distribuire ale anunțurilor (postări pe rețele sociale, afișe, mesaje distribuite între prieteni) sunt limitate și ineficiente în lipsa unei soluții centralizate.

În paralel, accesibilitatea tehnologiilor moderne web oferă oportunitatea de a crea aplicații cu impact social, care să răspundă unor nevoi reale ale utilizatorilor. În acest context, a apărut ideea dezvoltării unei aplicații web denumită FindMyPet, destinată persoanelor care pierd sau găsesc animale, oferindu-le o platformă rapidă, gratuită, accesibilă și vizuală pentru a publica și consulta anunțuri localizate geografic.

## Conturarea domeniului exact al problemei

Lucrarea se încadrează în domeniul dezvoltării aplicațiilor web interactive, combinând tehnologii frontend moderne (React.js, Vite), servicii Cloud (Firebase) și API-uri externe (Google Maps, EmailJS). Aceasta are ca scop proiectarea și implementarea unei aplicații complete, cu funcționalități specifice: autentificare, formulare pentru anunțuri și de contact, stocare imagini, localizare automată și afișare dinamică pe hartă.

Tematica lucrării implică elemente din mai multe arii:

* interacțiune om–computer (UI/UX)
* baze de date NoSQL (Firebase Firestore)
* programare web (JavaScript/React)
* servicii Cloud de tip BaaS
* geocodificare și hărți interactive (Google Maps API)

## Tema propriu-zisă

Lucrarea de față constă in dezvoltarea unei aplicații web denumite „FindMyPet”, destinată utilizatorilor care au nevoie de un mijloc de a-și căuta/găsi animalele de companie pierdute ori găsite. Această aplicație oferă utilizatorilor o interfață accesibilă, scalabilă si prietenoasă pentru adăugarea anunțurilor cu suport pentru imagini, descrieri si localizare geografică pe hartă.

## Obiectivele principale ale lucrării

Obiectivele lucrării sunt formulate pentru a răspunde unor cerințe reale identificate în contextul actual al gestionării anunțurilor pentru animale pierdute sau găsite. Aceste obiective vizează atât partea funcțională a aplicației, cât și integrarea unor tehnologii moderne și ușor de extins, astfel încât aplicația să fie practică, accesibilă și scalabilă. În cele ce urmează sunt prezentate obiectivele principale ale lucrării, împreună cu o scurtă explicație a fiecăruia.

1. Oferirea unei interfețe prietenoase si scalabile. Aceasta presupune o aplicație web moderne, care să funcționeze eficient atât pe desktop, cât și pe dispozitivele mobile. Scalabilitatea este importantă deoarece aplicația poate fi extinsă în viitor cu funcționalități suplimentare, fără a necesita rescrierea completă a codului. Prin utilizarea framework-ului React.js, structura aplicației este una modulară și ușor de întreținut.
2. Implementarea autentificării utilizatorilor. Acest obiectiv contribuie la filtrarea conținutului adăugat în aplicație și la editarea/ștergerea acestuia. Prin intermediul serviciului Firebase Authentification, autentificarea prin email si parola se gestionează automat, fără a necesita implementarea unei infrastructuri backend dedicate, ceea ce duce la economisirea timpului si resurselor.
3. Permisiunea de atașare a imaginilor și salvarea acestora. Imaginile încărcate de către utilizatori în formularul de adăugare sunt stocate în Firebase Storage, iar link-ul către imagine este asociat anunțului din baza de date Firestore. Astfel imaginile sunt asociate anunțurilor pentru a oferi un context vizual clar care este esențial pentru recunoașterea rapidă a animalelor pierdute sau găsite de către un alt utilizator din comunitate și astfel contribuie la eficiența aplicației web.
4. Integrarea unei hărți interactive. Aceasta este una dintre funcționalitățile cele mai eficiente din punct de vedere vizual al aplicației web. Harta interactivă este afișată pe prima pagină a aplicației si este integrată cu Google Maps API. Utilizatorii pot introduce locația în momentul publicării anunțului și aplicația plasează automat un pin pe hartă în locul specificat. Astfel, vizitatorii pot căuta rapid animalele de companie găsite sau căutate din zona dorită.
5. Vizualizarea rapidă a anunțurilor existente fără autentificare. Anunțurile publicate pot fi consultate de către orice utilizator al site-ului, fără ca acesta sa fie nevoit să își creeze un cont. Această decizie a fost luată pentru a evita situațile în care vizitatorii sunt într-un moment de grabă. Astfel se încurajează accesul mai larg si mai rapid la informații si se eficientizează găsirea unui animal de companie pierdut.
6. Implementarea unui formular de contact prin care utilizatorii pot trimite mesaje direct din aplicația web către dezvoltator. Formularul de contact funcționează cu ajutorul serviciului EmailJS și Cloud Firestore. Acesta folosește serviciul EmailJs pentru trimiterea de email-uri direct catre adresa de email a dezvoltatorului, astfel notificându-l. Serviciul Cloud Firestore este utilizat pentru a salva mesajele si pentru a le gestiona mai eficient.
7. Structurarea aplicației pe componente reutilizabile. Aplicația este dezvoltată în React.js, iar acest lucru de permite sa împărtim codul in componente independente (NavBar, ANimalCard, Hero etc.). Acest lucru îmbunătățește claritatea codului si permite reutilizarea elementelor comune, astfel ușurând procesul de intreținere.

# Studiu bibliografic

## Introducerea în cadrul bibliografic

Dezvoltarea unei aplicatii web moderne, precum FindMyPet, necesită înțelegerea unui set de tehnologii actuale. Studiul bibliografic oferă o sinteză a principalelor surse și documetații utilizate pentru alegerea, înțelegerea și implementarea acestor tehnologii.

## Aplicații web de tip SPA (Single Page Application)

Acest tip de aplicații sunt o solutie modernă în dezvoltarea aplicaților web. În cazul acestor aplicații, întreaga interfață se încarcă o singură dată și conținutul este actualizat dinamic fără reîncărcarea completă a paginii. Conform articolului publicat de SoftwareMind[[1]](#footnote-1), această abordare oferă o experiență mult mai rapidă si fluidă.

În etapa de documentare au fost consultate articole precum GeeksforGeeks [[2]](#footnote-2)care evidențiază avantajele folosirii unei astfel de arhitecturi pe bază de componente.

## Framework-ul React.js

React.js este una dintre cele mai populare biblioteci javaScript pentru construirea de interfețe web moderne si bazată pe componente. Sursa principală în înțelegerea mecanismului de state, props, hook-uri și routing a fost react.dev[[3]](#footnote-3). De asemenea, resurse importante au fost tutorialele video și articolele pentru a accelera procesul de învățare și implementare

## Firebase și BaaS

Platforma Firebase oferă un model de Backend-as-a-Service, astfel eliminând nevoia unui server propriu. Aceasta platforma a fost implementa pentru a favoriza autentificarea, baza de date Firestore și stocarea fișierelor, care sunt componente de bază pentru dezvoltarea aplicației. Pentru documentație au fost consultate atât surse oficiale (firebase.google.com/docs), cât și tutoriale video.

## Google Maps API

Pentru integrarea unei hărți interactive în aplicația web a fost folosit Google Maps API. Această integrarea a fost prin biblioteca @react-google-maps/api. Deși nu a fost consultată documentația oficială, implementarea a fost intuitivă din platforma console.cloud.google[[4]](#footnote-4).

## EmailJS

EmailJS este o soluție practică și modernă pentru trimiterea emailurilor direct din frontend, fără un server propriu. Acest serviciu oferă integrare cu platforme Gmail, Outlook sau alți provideri SMTP. EmailJS este utilizat în realizarea formularului de contact, permițând trimiterea de mesaje de feedback sau sugestii. Acest serviciu este recomandat de către comunitatea JavaScript datorită documentației clare.

## Soluții similare și justificarea temei

În procesul de documentare au fost analizate aplicații existente atât din România, cât și sin SUA sau Australia. Aplicațiile sunt însă fie limitate la anumite regiuni, fie contra cost. În țara noastră nu există o aplicație complet gratuită, centralizată, cu hartă integrată pentru eficientizarea căutării și backend Cloud, ceea ce justifică necesitatea și oportunitatea dezvoltării acestei aplicații web complet gratuite.

# Fundamentare teoretică

Aplicația FindMyPet se bazează pe o serie de concepte fundamentale din dezvoltarea web moderna.

În primul rând, aplicația este construită ca o Single Page Application (SPA). Asta înseamnă că pagina se încarcă o singură dată, iar apoi totul se actualizează dinamic, fără reîncărcări de fiecare dată când utilizatorul navighează între pagini. Din punctul de vedere al experienței, totul este mult mai rapid, mai fluid și mai apropiat de o aplicație mobilă, ceea ce contează atunci când cineva caută sa adauge rapid un anunț.

Folosirea bibliotecii React.js este în parte și deoarece este una dintre cele mai populare biblioteci JavaScript. Aceasta bibliotecă este foarte potrivită și deoarece permite organia=zarea aplicației în componente. De exemplu, în cadrul aplicației FindMyPet, am folosit componente separate pentru cardul cu informații despre animal (AnimalCard,jsx), pentru hartă (Map), pentru formularul de adăugare animale etc. Această compartimentare este eficientă in menținerea unei structuri organizate si ajută la modificarea în secțiuni fără efectarea altor zone.

# Specificaiile aplicaiei

Specificaiile urmresc descrierea interfeelor (canalelor de comunicaie) dintre aplicaia software (vzut ca o cutie neagr ) i mediu. Prin mediu

se înelege mul imea tuturor sistemelor cu care interacioneaz aplicaia. Categorii de astfel de sisteme pot fi:

x utilizatorul – sau utilizatorii (de aici rezultând necesitatea descrierii interfeei cu utilizatorul);

x sistemele cu care programul interacioneaz (în sensul din teoria sistemelor): procese industriale, procese din alte domenii (sisteme

complexe de programe, reele de calculatoare, sisteme bancare, sisteme biologice, sisteme de baze de date etc.), alte aplicaii sau sisteme

de calcul cu care programul interacioneaz , asimilabile acestor sisteme ( în particular chiar i pacienii din aplicaiile de informatic

medical );

x bazele de date, fi ierele cu care lucreaz aplicaia (ele pot fi privite ca f când parte din mediu întrucât sunt stocate pe suporturi de

memorie extern i deci coninutul lor trebuie de asemenea specificat);

x perifericele calculatorului pe care ruleaz programul (consol , imprimant etc. care trebuie privite, de asemenea, ca f când parte din

mediu întrucât sunt externe aplicaiei i deci coninutul lor trebuie de asemenea specificat).

Practic orice „intr ” sau „iese” din aplicaie trebuie descris în cadrul specificaiilor, cu explicitarea coninutului i formei de prezentare, astfel

încât proiectantul, clientul i utilizatorul final al aplicaiei s poat s cunoasc funcionarea aplicaiei, atât la nivel de funciuni cât i la nivel

formal (aspectul interfeelor). Descrierea trebuie realizat astfel încât:

x pe tot parcursul proiect rii s se poat urmri în permanen , ca obiectiv esenial, concordana dintre specificaii i ceea ce se obine;

x s se poat valida produsul obinut prin verificarea în cele mai mici detalii a concordanei dintre acesta i specificaii;

x clientul i utilizatorul final s poat s îneleag funcionarea i s cerceteze aspectul interfeelor i interconect rii cu alte sisteme

înainte de a începe proiectarea propriu-zis , pentru a valida modelul dezirabil care s se constituie ulterior în obiectiv pentru

proiectare

x proiectanii s poat s continue modelarea i proiectarea sistemului dorit, prin împ rirea în subsisteme, astfel încât intr rile i ie irile

s fie definite exact, plecând de la cele ale aplicaiei vzute ca o cutie neagr ;

x proiectanii s poat s abordeze în paralel i proiectarea altor sisteme cu care aplicaia interacioneaz , în condi iile în care “canalele

de comunicaie” sunt definite riguros.

4.1. Schema-bloc a sistemului. Scurt descriere a aplicaiei

Paragraful este destinat prezent rii unei scheme care s ilustreze calculatorul pe care ruleaz aplicaia, periferia care este folosit , precum i

interaciunea acestui calculator sau a aplicaiei cu sisteme din exteriorul aplicaiei – utilizatori, alte aplicaii cu care interacioneaz , alte sisteme

de calcul, procese cu care interacioneaz etc. Pe baza acestei scheme se procedeaz la o scurt descriere a obiectivelor (scopului) pentru care

este conceput aplicaia precum i a modului de funcionare pentru îndeplinirea acelor obiective, performanele vizate.

4.2. Funciile sistemului

Funciile sistemului se stabilesc pe baza obiectivelor i se sistematizeaz de regul sub forma de list în care se descriu succint aciunile pe care

le realizeaz programul. Funciile trebuie s fie net delimitate unele de altele. Ele trebuie încadrate în categorii bine definite.

Exemplu:

comunicarea cu nivelul ierarhic inferior, pentru un sistem de supraveghere de proces, arhivarea/readucerea datelor etc.

De regul , se pot identifica în cadrul unei aplicaii uzuale 4-8 funcii. Funciile trebuie prezentate distinct. Fiecare funcie trebuie descris prin 1

3 fraze. În cadrul descrierii unei funcii, trebuie identificat prin accentuarea în text („bold”) un grup de cuvinte (o sintagm) care s ofere o

caracterizare sintetic a funciei respective.

În cadrul descrierii funciilor trebuie s fie transpuse sintetic toate cerinele exprimate în cadrul Temei de proiectare. Astfel, dac sistemul

trebuie s reacioneze într-un anumit mod la evenimente externe, atunci aceast reacie trebuie s fie explicat clar i univoc la funcia c reia ea

îi corespunde. Toate cerinele trebuie s se regseasc sub o form sau alta în cadrul prezent rii funciilor.

4.3. Interfaa cu utilizatorul

Paragraful este destinat definirii tuturor aspectelor care privesc interfaa cu utilizatorul (aspectul ecranului în toate situaiile posibile, meniuri,

submeniuri, cu aciuni preconizate la fiecare comand , ecrane de dialog, definirea aciunilor pentru toate elementele de comand , listele de

mesaje ale sistemului, aspectul generic al graficelor, rapoartelor, schemelor, listelor, modalit i de interaciune cu utilizatorul, specifice etc.).

Se vor realiza desene care s ilustreze o descriere complet a interfeelor în sensul raiunilor prezentate în preambul.

4.4. Structuri de baze de date i fi iere

Trebuie definite la nivel logic structurile de baze de date i fi iere (adic la nivelul la care sunt specificate spre exemplu câmpuri ale unui tabel,

dar cu referire doar la coninut i nu la mrimea câmpului). Unele eventuale leg turi între tabele urmeaz a fi definite doar la proiectarea de

detaliu (altele pot fi deja specificate aici).

4.5. Comunicarea cu alte sisteme

Dac sistemul interacioneaz cu alte sisteme, pentru fiecare canal de comunicaie identificat trebuie explicate:

x modul de comunicare (suportul fizic)

x protocolul de comunicare folosit (dac este un standard, se menioneaz acest lucru i dac nu este foarte cunoscut, se prezint succint

principalele aspecte)

x structurarea i semnificaia informaiei vehiculate

Exemplu:

Pentru un sistem ierarhic de conducere de proces, trebuie prezentat pe larg structura informaiei vehiculate în ambele sensuri între nivele

ierarhice, semnificaiile pentru fiecare component , la nivelul de detaliere minim necesar, semnificaiile comenzilor etc., astfel încât s se poat

realiza proiectarea simultan a ambelor componente software pe baza referenialului comun al acestor specificaii.

4.6. Tiprirea la imprimant

Se prezint generic rapoartele ( eventual exemple) i lista eventualelor mesaje care pot fi tip rite de sistem la imprimant .

Atunci când e cazul se vor aduga paragrafe distincte pentru alte periferice.

4.7. Analiza de risc

Paragraful este destinat evalu rii aspectelor de siguran în funcionare dorite (dac e cazul). O aplicaie desktop obinuit (pentru calcule

tiinifice sau simulare spre exemplu) nu este considerat a fi din categoria care trebuie s aib o siguran deosebit în funcionare, în schimb

una de conducere de proces sau un sistem bancar, da.

Se va face, dac e cazul, o analiz în funcionare degradat , care urmre te consecinele cderii unor componente ale sistemului: aceste cderi

vor fi ierarhizate în ordinea cresc toare a consecinelor asupra bunei funcion ri a sistemului; vor fi prezentate aceste consecine, modul de

reacie dezirabil al operatorului i eventual vor fi propuse msuri ce pot fi luate chiar prin proiectarea programului, astfel ca aplicaia s

minimizeze consecinele.

4.8. Planificarea lucr rilor

Se va propune un model de ciclu de via pentru dezvoltarea programului, cu justificarea alegerii i un grafic de ealonare a lucr rilor, conform

acestui model (o diagram GANTT). Pentru realizarea diagramei se recomand , dac se cunoa te, utilizarea Microsoft Project.

Fiecare etap va cuprinde denumirea, termenul, resursele necesare (proiectani, sisteme de calcul, software, altele).

# Proiectarea de detaliu

Obiectivul primordial al proiect rii de detaliu e s se ajung la o descriere care s permit pe de o parte delimitarea exact a muncii în cadrul

echipei i continuarea cu aciunea de codificare (scriere de cod-surs ) în cadrul c reia fiecare proiectant s tie exact ce s fac pentru a scrie

codul ca activitate de rutin .

5.1. Arhitectura programului

Aici se prezint o descriere succint a programului, pe baza unei scheme generale (“arhitectura programului”), cu descrierea componentelor i a

interaciunilor dintre acestea. Vor fi evideniate aspectele de tehnologie folosite, inclusiv mediul de dezvoltare, modelele arhitecturale (“client

server”, “tree-tier” etc.), principiile generale de funcionare, vor fi date detalii despre sistemul de operare, dac e cazul (eventual ca i paragraf

separat). Dac aplicaia este pe baz pe dialog, se va face o schem a formelor aplicaiei i a arborescenei de parcurgere a tuturor dialogurilor.

5.2. Descrierea componentelor

În funcie de tehnologia folosit , vor fi descrise individual toate componentele de program folosite, pentru fiecare fiind prezentate intr rile,

prelucr rile i ie irile i evideniate toate interaciunile cu alte module. Se poate propune eventual un ablon de descriere care s fie folosit pentru

descrierea unitar a tuturor modulelor.

5.3. Descrierea comunic rii între module

În funcie de tehnologia folosit , vor fi descrise individual toate canalele de comunicaie dintre componentele de program folosite, pentru fiecare

fiind prezentate modul de comunicare, lista de parametri, semnificaiile acestora, restricii etc. Se poate propune eventual un ablon de descriere

care s fie folosit pentru descrierea unitar a tuturor acestor canale.

5.4. Principalele structuri de date

Vor fi descrise (dac e cazul) principalele structuri de date folosite în program, semnificaiile, drepturile de acces ale diferitelor module de

program etc.

5.5. Structuri de baze de date i fi iere

Se prezint pe larg structurile, prin dezvoltarea informaiilor de la paragraful 4.4. Vor fi prezentate toate informaiile aferente unui câmp,

inclusiv cele deja existente (adic semnificaia), tip, lungime, restricii, eventual drepturi de acces. Vor fi definite leg turile, cheile etc.

De asemenea se va realiza o schem general care s cuprind toate tabelele, cu leg turi între ele.

5.6. Proceduri (funcii, subrutine)

Pentru fiecare procedur important vor fi prezentate: numele, descrierea prelucr rii specifice, lista parametrilor de apel i de retur, pentru

fiecare parametru tipul, mrimea, semnificaia, eventuale restricii, interaciunea cu structurile de date, cazuri de eroare etc. Se poate propune

eventual un ablon de descriere care s fie folosit pentru prezentarea unitar a tuturor funciilor.

În funcie de particularit ile aplicaiei se vor aduga sau scoate paragrafe, vor fi definite altele noi. Vor fi subliniate soluiile tehnice dificile, iar

apoi ele vor fi prezentate amnunit, cu explicarea detaliat a funcion rii.

# Utilizarea sistemului

Capitolul vizeaz o prezentare similar unui manual de utilizare (cu multe exemple, capturi de ecran etc.). Toi absolvenii au studiat manuale de

utilizare ale unor produse, fiind deci în msur s elaboreze în condiii foarte bune acest capitol. Totui se subliniaz importana unei exprimri

coerente, într-un limbaj tehnic i nu colocvial. Materialul trebuie s fie îngrijit, structurat logic i uor de îneles de un potenial utilizator.

# Realizarea, punerea în funciune i rezultate experimentale

7.1. Realizarea programului

Se va prezenta modul de realizare al programului (modelul de ciclu de via , platforma, alte aspecte tehnologice) i modul de respectare al

planului propus iniial (paragraful 4.8). Vor fi analizate motivele abaterilor de la planul iniial.

7.1. Tehnologia utilizat pentru testare

Vor fi prezentate pe larg tehnologiile i configuraiile de testare i validare. Se vor prezenta aciunile concrete întreprinse. Va fi explicat modul

de verificare a concordanei caracteristicilor programului cu specificaiile ini iale (Cap. 4).

7.2. .Probleme întâmpinate i modul lor de rezolvare

Vor fi date exemple concrete de probleme ap rute în timpul test rii i modul lor de rezolvare.

7.3. Rezultate experimentale

Vor fi prezentate rezultate considerate semnificative din punctul de vedere al utiliz rii produsului realizat (exemple de rulare, de utilizare, cu

capturi de ecran, grafice, tabele etc.). Trebuie s reias utilitatea produsului, eventual dac e posibil se vor face comparaii ale rezultatelor cu

date obinute pe alte c i.

Cap. 8. Concluzii

8.1. Ce s-a realizat

8.2. Comparaia cu alte realiz ri similare

8.3. Direcii de dezvoltare

Bibliografie

Anexe (inclusiv discheta sau CD cu cod surs i documentaie)

# N. CONCLUZII (14 pt, Bold, Uppercase, Center)

Lucrarea se va încheia cu un capitol de concluzii. Acesta va conţine principalele rezultate ale lucrării şi implicaţiile practice ale acestora. În cazul proiectelor de diplomă, se vor menționa principalele date sintetice obținute din procesul de proiectare

# BIBLIOGRAFIE

La sfârşitul lucrării va fi dată o listă de referinţe pentru textele ştiinţifice consultate pe parcursul realizării lucrării. Vor fi trecute toate sursele, inclusiv cele de pe internet. Acestea vor fi referite în text şi trecute în lista de referinţe în ordine alfabetică, după exemplele de mai jos.

Bibliografia trebuie să cuprindă toate titlurile din literatura de specialitate care au servit ca bază de documentare, respectiv autorii care au fost citați în text, la toate capitolele lucrării. Facultățile pot adopta sisteme de citare proprii în concordanță cu specificul domeniilor coordonate.

În lipsa unor cerințe de citare impuse, bibliografia va fi redactată conform normelor Academiei Române.

În textul lucrării referințele vor fi referite în paranteze, spre exemplu (Tremblay, 2002).Referințele vor fi menționate în limba în care au fost consultate (nu vor fi traduse).

Lista bibliografică va fi aranjată în ordine alfabetică, a primului autor. În cazul în care avem două sau mai multe lucrări scrise de același autor acestea vor fi trecute în ordine cronologică. Referințele vor include:

* numele autorului, prenumele, titlul - scris cu Italic - volumul, localitatea, editura, anul;
* pentru un studiu sau o publicație periodică se trec numele și prenumele autorului, titlul articolului - scris cu Italic - numele publicației, anul de apariție a seriei, seria, numărul, anul calendaristic.

Sursele bibliografice, publicate sau nepublicate trebuie să se regăsească în lista bibliografică finală, după cum toți autorii incluși în lista bibliografică trebuie să fie inserați în textul lucrării.

Preluarea identică a unei fraze sau paragraf va fi citată prin indicarea inclusiv a paginii din sursa utilizată, dar și prin ghilimele şi forma italică a literelor; pentru sursele preluate de pe internet, vor fi notate adresele de pagină web; în lista bibliografică finală lucrările se trec în ordinea alfabetică a numelor autorilor. La lucrările colective, regula referitoare la ordinea alfabetică este valabilă pentru primul autor.

Dacă se citează site-uri web, reviste sau articole, înainte de acestea se vor trece trei asteriscuri, informații referitoare la volum, număr, pagini consultate, adresa web exactă a articolului respectiv, data vizitării site-ului și a descărcării materialului, data accesării. Adresele de pagini web se regăsesc la finalul listei.

Sursele bibliografice la care nu se poate menționa autorul se vor specifica astfel: „\*\*\*”urmat de denumirea articolului și/sau a cărții, editura și locul apariției (pentru cărți), volumul, numărul acestuia, prima și ultima pagină a lucrării citate, anul apariției.

Sursele bibliografice consultate pe Internet se vor specifica astfel: pagina care a fost consultată, data ultimei accesări. Este recomandat să se alăture link-ul către pagina respectivă de internet. Notele și trimiterile bibliografice se trec la sfârșitul capitolului/lucrării, în ordinea în care apar în text; acolo unde este inserată nota sau trimiterea bibliografică, la ultimul cuvânt se marchează numărul acesteia, la exponent, cu cifre arabe; la subsol sau la sfârșitul capitolului se trece conținutul notei, respectiv trimiterii bibliografice, după cerințele menționate.

Exemple de referințe:

Coulouris, G., Dollimore, J., Kindberg, T., *Distributed systems, concepts and design*, 3rd edition, Pearson, Addison Wesley, 2001.

Tremblay, R., *Inelastic seismic response of steel bracing members,* Journal of Constructional Steel Research, 58, pp 665–701, 2001.

Neculau, A., Cozma, T. (coord.), Psiho -pedagogie - pentru examenul de definitivat şi gradul II, Iaşi, Editura Spiru Haret, 1994.

\*\*\* https://ro.wikipedia.org/wiki/Motor\_cu\_reac%C8%9Bie accesare februarie 2022)

<http://www.upt.ro/pdf/licenta&master/Declaratie_de_autenticitate_UPT.pdf>

1. SoftwareMind “What is a Single Page Application?”, 2023, https://softwaremind.com/blog/what-is-a-single-page-application-spa/ [↑](#footnote-ref-1)
2. GeeksforGeeks “How to Build Single Page Applications?”, https://www.geeksforgeeks.org/how-to-build-single-page-applications/ [↑](#footnote-ref-2)
3. React Official Documentation, https://react.dev/learn [↑](#footnote-ref-3)
4. APIs & Services, https://console.cloud.google.com/google/maps-apis [↑](#footnote-ref-4)