< Platformă Web pentru Analiza și Vizualizarea Datelor Medicale pentru îmbunătățirea Deciziilor Clinice >

# Documentul de proiectare

Cuprins

[1. Introducere 1](#_Toc160527836)

[1.1 Scopul documentului 1](#_Toc160527837)

[2. Prezentare generală și abordări de proiectare 2](#_Toc160527838)

[2.1 Prezentare generală 2](#_Toc160527839)

[2.2 Presupuneri/ Constrângeri/ Riscuri 2](#_Toc160527840)

[2.2.1 Presupuneri 2](#_Toc160527841)

[2.2.2 Constrângeri 2](#_Toc160527842)

[2.2.3 Riscuri 3](#_Toc160527843)

[3. Considerațiii de proiectare 4](#_Toc160527844)

[3.1 Obiective și linii directoare (ghiduri) 4](#_Toc160527845)

[3.2 Metode de dezvoltare 4](#_Toc160527846)

[3.3 Strategii de arhitectură 4](#_Toc160527847)

[4. Arhitectura Sistemului și Proiectarea Arhitecturii 6](#_Toc160527848)

[4.1 Vedere logică 6](#_Toc160527849)

[4.2 Arhitectură hardware 6](#_Toc160527850)

[4.3 Arhitectură software 6](#_Toc160527851)

[4.4 Arhitectura informațiilor 7](#_Toc160527852)

[4.5 Arhitectura de comunicații interne 7](#_Toc160527853)

[4.6 Diagrama de arhitectură a sistemului 8](#_Toc160527854)

[5. Proiectarea sistemului 9](#_Toc160527855)

[5.1 Proiectarea bazei de date 9](#_Toc160527856)

[5.1.1 Obiecte de date și structuri de date rezultante 9](#_Toc160527857)

[5.1.2 Fișiere și baze de date 9](#_Toc160527858)

[5.2 Conversii de date 9](#_Toc160527859)

[5.3 Interfețe utilizator 10](#_Toc160527860)

[5.3.1 Intrări 10](#_Toc160527861)

[5.3.2 Ieșiri 10](#_Toc160527862)

[5.4 Proiectarea interfețelor cu utilizatorul 10](#_Toc160527863)

[6. Scenarii de utilizare 11](#_Toc160527864)

[7. Proiectare de detaliu 12](#_Toc160527865)

[7.1 Proiectare hardware de detaliu 12](#_Toc160527866)

[7.2 Proiectare software de deatliu 12](#_Toc160527867)

[7.3 Proiectare detaliată de securitate 13](#_Toc160527868)

[7.4 Proiectare de detaliu pentru performanța sistemului 14](#_Toc160527869)

[7.5 Proiectare detaliată a comunicațiilor interne (între componente) 14](#_Toc160527870)

[8. Controale pentru verificarea integrității sistemului 15](#_Toc160527871)

[Anexa A: Gestiunea modificărilor documentului 16](#_Toc160527872)

[Anexa B: Acronime 17](#_Toc160527873)

[Anexa C Documente la care se face referire 18](#_Toc160527874)

## Introducere

Instrucțiuni: Furnizați informații de identificare pentru sistemul existent și/sau propus sau situația pentru care se aplică Documentul de Proiectare a Sistemului (de exemplu, numele complet și acronimele proiectului de dezvoltare, sistemul sau situația existentă și propusă, după caz), și evoluția așteptată a documentului. De asemenea, descrieți orice considerații referitor la securitate sau confidențialitate asociate cu utilizarea acestui document..

Documentul de față descrie proiectarea sistemului aferent lucrării de licență intitulate „Platformă Web pentru Analiza și Vizualizarea Datelor Medicale pentru Îmbunătățirea Deciziilor Clinice”. Scopul acestei platforme este de a oferi un suport decizional clinic bazat pe inteligență artificială (IA), prin colectarea, analiza și prezentarea vizuală a datelor medicale furnizate de pacienți și gestionate de specialiști.

Platforma este un sistem informatic propus, dezvoltat de la zero în cadrul lucrării de licență, și nu reprezintă o extensie directă a unui sistem existent. Totuși, designul acesteia se bazează pe nevoile identificate în practica medicală actuală, în special în contextul creșterii volumului de date și al necesității de asistență tehnologică pentru luarea deciziilor clinice rapide și informate.

Evoluția acestui document este strâns legată de dezvoltarea incrementală a platformei. Pe măsură ce funcționalitățile sunt implementate și testate, documentația va fi actualizată pentru a reflecta arhitectura detaliată, tehnologiile utilizate, modul de interacțiune între componente și regulile de securitate și confidențialitate adoptate.

Considerații privind securitatea și confidențialitatea:

Având în vedere natura sensibilă a datelor prelucrate, respectiv informații medicale personale sistemul impune un set riguros de măsuri de securitate informatică. Acestea includ criptarea datelor stocate și transmise, autentificarea utilizatorilor pe roluri distincte (pacient, medic, administrator), precum și respectarea principiilor legislației în vigoare, în special Regulamentul General privind Protecția Datelor (GDPR). Accesul la acest document este, de asemenea, restricționat în afara contextului educațional sau profesional explicit aprobat, întrucât poate conține specificații detaliate ale arhitecturii sistemului și ale măsurilor de securitate implementate.

### Scopul documentului

*Instrucțiuni: Furnizați scopul documentului. Acest document ar trebui să fie adaptat pentru a se potrivi nevoilor unui proiect specific.*

Prin acest livrabil se documentează și urmăresc informațiile necesare pentru a defini eficient arhitectura și designul sistemului, în scopul de a oferi echipei de dezvoltare îndrumare asupra arhitecturii sistemului ce urmează să fie dezvoltat.

Documentele de proiectare sunt produse incremental și iterativ pe parcursul ciclului de viață al dezvoltării sistemului, în funcție de circumstanțele particulare ale proiectului de tehnologie informațională (IT) și de metodologia de dezvoltare a sistemului utilizată.

Publicul țintă este managerul de proiect, echipa de proiect și echipa de dezvoltare. Anumite părți ale acestui document, cum ar fi interfața cu utilizatorul (UI), pot fi împărtășite cu clientul/utilizatorul și cu alte părți interesate ale căror contribuții/aprobări sunt necesare în UI.

Scopul acestui document este de a furniza o descriere structurată și detaliată a arhitecturii și designului sistemului informatic propus în cadrul proiectului „Platformă Web pentru Analiza și Vizualizarea Datelor Medicale pentru Îmbunătățirea Deciziilor Clinice”. Acest livrabil are rolul de a documenta și urmări toate informațiile relevante necesare pentru a defini în mod eficient structura și componentele sistemului, oferind echipei de dezvoltare o bază solidă pentru implementarea coerentă și conformă a cerințelor funcționale și nefuncționale.

Publicul țintă al acestui document este reprezentat de managerul de proiect, echipa de proiect și echipa de dezvoltare. De asemenea, anumite secțiuni ale documentului, precum cele referitoare la interfața cu utilizatorul (UI), pot fi diseminate către client, utilizatori finali și alte părți interesate, ale căror feedback și aprobare sunt esențiale pentru validarea componentelor vizibile ale sistemului.

## Prezentare generală și abordări de proiectare

Această secțiune descrie principiile și strategiile care vor fi utilizate ca ghiduri în momentul proiectării și implementării sistemului.

Proiectarea Platformei Web pentru Analiza și Vizualizarea Datelor Medicale pentru Îmbunătățirea Deciziilor Clinice se bazează pe un set clar definit de principii arhitecturale și strategii de dezvoltare, menite să asigure scalabilitatea, securitatea, modularitatea și ușurința în mentenanță a sistemului. Aceste principii oferă direcții generale care guvernează toate deciziile tehnice adoptate în cadrul proiectului și constituie fundamentul pentru dezvoltarea unei soluții informatice robuste și adaptabile la cerințele mediului medical contemporan.

Principalele principii:

* Modularitate: istemul este împărțit în componente independente (frontend, backend, baze de date, servicii AI), ceea ce facilitează dezvoltarea separată, testarea individuală și întreținerea eficientă a fiecărui modul.
* Separarea responsabilităților: Fiecare componentă a sistemului este responsabilă pentru un set bine definit de funcționalități, contribuind la claritatea și organizarea codului.
* Scalabilitatea: Fiecare componentă a sistemului este responsabilă pentru un set bine definit de funcționalități, contribuind la claritatea și organizarea codului.
* Acesibilitate si User Experience: Platforma este proiectată pentru a oferi o interfață intuitivă, accesibilă atât pacienților, cât și profesioniștilor din domeniul medical, cu accent pe claritate, eficiență și compatibilitate cross-platform.

### Prezentare generală

Instrucțiuni: Prezentați succint contextul sistemului și abordarea sau organizarea de bază a proiectării. Oferiți o prezentare generală a arhitecturilor sistemului și software-ului și a obiectivelor de proiectare.

Platforma MedAI este concepută ca un sistem informatic web destinat sprijinirii procesului decizional clinic prin analiza și vizualizarea datelor medicale colectate de la pacienți. În contextul digitalizării accelerate a serviciilor medicale și al necesității de a gestiona volume tot mai mari de informații, sistemul propus integrează componente de inteligență artificială pentru interpretarea simptomelor, generarea de sugestii preliminare și susținerea deciziilor luate de medici.

Arhitectura sistemului este organizata in patru componente majore:

1. Frontend
2. Backend
3. Componenta AI
4. Baza de date

### Presupuneri/ Constrângeri/ Riscuri

#### Presupuneri

Instrucțiuni: Descrieți orice presupuneri sau dependențe legate de sistem, software și utilizarea sa. Acestea pot viza probleme precum: software sau hardware asociat, sisteme de operare, caracteristicile utilizatorilor finali și posibilele și/sau probabilele modificări ale funcționalității.

1. **Presupuneri privind infrastructura tehnică:**

* **Sistemul va fi implementat pe un server compatibil cu tehnologiile web moderne**, care suportă un stack tehnologic bazat pe Node.js (pentru backend), o bază de date relațională (precum PostgreSQL sau MySQL) și un server web (ex. Nginx sau Apache).
* **Utilizatorii vor accesa platforma prin intermediul unui browser modern**, compatibil cu HTML5, CSS3 și JavaScript (Google Chrome, Mozilla Firefox, Microsoft Edge, Safari).
* **Componenta de inteligență artificială** va rula fie local (în cazul unei instanțe de testare), fie prin intermediul unei platforme de cloud computing (ex. Google Cloud, AWS, Azure), cu resurse hardware minime garantate (ex. CPU dedicat, memorie RAM > 4GB pentru rularea modelelor).

1. **Presupuneri privind utilizatorii finali:**

* Se presupune că **utilizatorii dispun de competențe digitale de bază**, fiind capabili să acceseze, completeze formulare și să navigheze într-o interfață web standard.
* Utilizatorii medici și administratori sunt familiarizați cu aplicațiile informatice din domeniul sănătății și vor parcurge un scurt instructaj privind utilizarea funcțiilor platformei.

1. **Presupuneri legate de funcționalitate și dezvoltare:**

* Este posibil ca, în viitor, să fie necesare **adaptări ale funcționalității pentru integrarea cu sisteme externe** (ex. dosarul electronic al pacientului, aplicații de telemedicină sau farmacii digitale).
* Funcționalitatea de analiză a simptomelor pe baza inteligenței artificiale este oferită **cu titlu de suport**, nu ca diagnostic medical final, și presupune acceptarea acestei limitări de către utilizatori.
* Se presupune că **funcționalitățile vor evolua incremental**, iar arhitectura modulară va permite integrarea ulterioară a unor noi module fără restructurarea profundă a sistemului existent.

#### Constrângeri

Instrucțiuni: Descrieți orice limitări sau constrângeri globale care au un impact semnificativ asupra designului hardware-ului, software-ului și/sau comunicațiilor **s**istemului și descrieți impactul asociat. Astfel de constrângeri pot fi impuse de oricare dintre următoarele (lista nu este exhaustivă):

* Mediu hardware sau software
* Mediu utilizator final
* Disponibilitatea sau volatilitatea resurselor
* Conformitatea cu standardele
* Cerințe de interoperabilitate
* Cerințe de interfață/protocol
* Cerințe de licențiere
* Cerințe pentru depozitarea și distribuția datelor
* Cerințe de securitate (sau alte reglementări similare)
* Limitări ale capacității de memorie sau ale altor resurse
* Cerințe de performanță
* Comunicații de rețea
* Cerințe de verificare și validare (testare)
* Alte mijloace de a aborda obiectivele de calitate
* Alte cerințe descrise în Documentul de Cerințe

1. **Constrângeri tehnice:**

* **Limitări de infrastructură hardware și software** – Sistemul trebuie să funcționeze eficient în medii hardware standard (servere comerciale, echipamente on-premises sau cloud) și pe sisteme de operare compatibile (Linux, Windows Server). Acest lucru impune alegeri tehnologice care să minimizeze consumul de resurse și să maximizeze compatibilitatea.
* **Resurse computaționale limitate pentru componenta AI** – Modelele de inteligență artificială utilizate pentru interpretarea simptomelor pot necesita resurse de procesare semnificative. În lipsa unui GPU dedicat, funcționalitățile avansate pot fi limitate sau externalizate către servicii cloud, cu implicații asupra latenței și costurilor.

1. **Constrângeri legate de utilizatori:**

* **Diversitatea profilului utilizatorilor finali** – Platforma trebuie să fie accesibilă atât personalului medical, cât și pacienților, inclusiv celor cu competențe digitale minime. Aceasta presupune o interfață prietenoasă, clară, cu un design inclusiv și elemente vizuale intuitive.

1. **Constrângeri de conformitate și reglementare:**

* **Conformitatea cu reglementările GDPR** – Având în vedere prelucrarea datelor cu caracter medical, sistemul trebuie să implementeze măsuri stricte de securitate (criptare, autentificare, audit) și să permită utilizatorilor gestionarea consimțământului și a drepturilor asupra propriilor date.
* **Cerinte de interoperabilitate** – Platforma poate necesita integrare cu alte sisteme medicale (ex. dosarul electronic al pacientului, baze de date instituționale), ceea ce impune respectarea unor protocoale de comunicare standard

1. **Constrângeri operaționale:**

* **Cerințe de stocare și distribuție a datelor** – Datele trebuie să fie stocate în mod securizat și să fie accesibile în timp real pentru utilizatori autorizați, fără întârzieri semnificative. Aceasta presupune o structură optimizată a bazei de date și implementarea unor politici de backup și redundanță.
* **Licențierea tehnologiilor utilizate** – Alegerea componentelor software (framework-uri, biblioteci, modele AI) trebuie să țină cont de tipurile de licență (ex. MIT, GPL, licențe comerciale), pentru a evita restricții legale sau costuri neprevăzute.
* **Cerințe de performanță și disponibilitate** – Platforma trebuie să asigure un timp de răspuns acceptabil (sub 2 secunde pentru acțiuni comune) și o disponibilitate ridicată (ideal 99.5%), mai ales în contextul utilizării în medii clinice.

1. **Constrângeri de verificare și testare:**

* **Necesitatea validării riguroase a componentelor AI** – Funcționalitatea de analiză a simptomelor trebuie testată extensiv pentru a asigura acuratețea rezultatelor și pentru a preveni generarea de sugestii incorecte sau potențial dăunătoare.
* **Testare pe multiple platforme și browsere** – Platforma trebuie testată pe diverse sisteme de operare și browsere, pentru a garanta compatibilitatea și funcționalitatea uniformă.

#### Riscuri

Instrucțiuni: Descrieți orice riscuri asociate cu designul sistemului și strategiile propuse de reducere a acestora.

1. **1. Riscuri tehnice**

* **Riscul de incompatibilitate între componentele software**  
  *Descriere:* Utilizarea unor tehnologii open-source și integrarea unor module AI externe poate genera conflicte de versiune sau incompatibilități.  
  *Măsuri de reducere:* Testare continuă în medii de integrare, utilizarea containerizării (ex. Docker) și a unei documentații stricte a versiunilor.
* **Performanță redusă în cazul încărcării mari**  
  *Descriere:* Sistemul poate întâmpina dificultăți de scalare la un număr mare de utilizatori simultani.  
  *Măsuri de reducere:* Adoptarea unei arhitecturi scalabile, folosirea serviciilor cloud și implementarea de mecanisme de cache și load balancing.

1. **2. Riscuri de securitate**

* **Expunerea neautorizată a datelor medicale sensibile**  
  *Descriere:* Platforma gestionează informații cu caracter personal și medical, ceea ce o face o țintă pentru atacuri cibernetice.  
  *Măsuri de reducere:* Implementarea criptării end-to-end, autentificare multi-factor (MFA), politici de acces bazate pe roluri (RBAC) și audit de securitate periodic.
* **Vulnerabilități în modulele AI**  
  *Descriere:* Modelele AI ar putea fi manipulate sau interpretate greșit, ceea ce poate afecta fiabilitatea recomandărilor.  
  *Măsuri de reducere:* Antrenarea modelelor pe seturi de date validate, testare extensivă, limitarea deciziilor critice bazate exclusiv pe AI.

1. **3. Riscuri legale și de conformitate**

* **Nerespectarea reglementărilor privind protecția datelor (GDPR)**  
  *Descriere:* Orice abatere de la normele privind datele cu caracter personal poate atrage sancțiuni legale.  
  *Măsuri de reducere:* Consultarea permanentă cu experți în protecția datelor, implementarea funcționalităților de consimțământ explicit și drept de ștergere a datelor.

## Considerații de proiectare

Instrucțiuni: Descrieți problemele care trebuie abordate sau rezolvate înainte de a încerca să elaborați o soluție de design completă.

1. Confidentialitatea si securitatea datelor medicale: Gestionarea datelor cu caracter personal și medical impune respectarea strictă a principiilor de protecție a datelor, în special conform legislației europene.

2. Accesibilitatea si UX: Dat fiind că platforma va fi utilizată de pacienți, medici și administratori cu niveluri diferite de pregătire tehnologică, designul interfeței trebuie să respecte principii de accesibilitate și să ofere o experiență intuitivă, coerentă și eficientă pentru toți utilizatorii.

3. Scalabilitatea arhitecturii: Posibilitatea extinderii sistemului odata cu cresterea numarului de utilizatori

4. Interoperabilitatea cu alte sisteme medicale: sa tina cont de nevoia de integrare cu sisteme externe ca de exemplu: farmacii online.

5. Precizia si transparenta AI: Având în vedere rolul critic al AI în analizarea simptomelor și generarea sugestiilor, este necesară definirea clară a limitelor acestor funcționalități. Modelele AI trebuie să fie antrenate pe seturi de date relevante, validate clinic și să ofere o trasabilitate a procesului decizional, astfel încât să poată fi auditate și înțelese de către medici.

### Obiective și linii directoare (ghiduri)

Instrucțiuni: Descrieți orice obiective, linii directoare, principii sau priorități care domină sau încorporează designul sistemului și al software-ului. Exemple de astfel de obiective ar putea fi: accentul pe viteză versus utilizarea memoriei; sau utilizare, aspect sau "să se simtă" similar cu un produs existent. Linii directoarele includ ghiduri și convenții de codare. Pentru fiecare ghid, descrieți motivul, cu excepția cazului în care este implicit evident. Descrieți orice politici și/sau strategii de design care nu au implicații arhitecturale majore (adică nu afectează semnificativ organizarea generală a sistemului și structurile sale de nivel înalt), dar care totuși afectează detaliile interfeței și/sau implementarea diferitelor aspecte ale sistemului (de exemplu, alegerea unui produs specific de utilizat).

1. **Obiective principale de proiectare**

* **Securitatea și confidențialitatea datelor**  
  Obiectivul prioritar este protejarea datelor medicale cu caracter personal. Designul impune criptarea datelor în tranzit și în repaus, politici riguroase de autentificare și autorizare, precum și auditarea tuturor activităților sensibile.
* **Accesibilitatea și ușurința în utilizare**  
  Platforma trebuie să fie ușor de utilizat pentru o gamă diversă de utilizatori, inclusiv pacienți fără pregătire tehnologică. Interfața va respecta principiile de design centrat pe utilizator și standardele WCAG 2.1 pentru accesibilitate.
* **Modularitate și scalabilitate**  
  Sistemul va fi proiectat într-o arhitectură modulară (bazată pe microservicii), permițând adăugarea ulterioară de funcționalități fără afectarea componentelor existente.
* **Fiabilitate și reziliență**  
  Platforma trebuie să funcționeze în mod stabil și continuu, chiar și în condiții de trafic ridicat sau într-un mediu imprevizibil. Vor fi implementate mecanisme de toleranță la erori și backup automat.
* **Performanță optimizată**  
  Se va urmări un echilibru între viteză de execuție și consum de resurse. Prioritară este timpul de răspuns scurt în interacțiunea cu utilizatorul, mai ales în modulele AI și vizualizare.

1. **Linii directoare și convenții tehnice**

* **Standardizare și reutilizare a codului**  
  Se va respecta principiul DRY (Don’t Repeat Yourself), iar componentele UI și funcționalitățile comune vor fi implementate ca module reutilizabile. Aceasta contribuie la mentenanță și testare eficientă.
* **Utilizarea convențiilor de codare moderne**  
  Limbajele și framework-urile utilizate (ex. TypeScript + React, Node.js, Python pentru AI) vor respecta convențiile oficiale ale comunității și vor fi verificate automat (linting, formatare cu Prettier, ESLint, PEP8 pentru Python).
* **Separarea clară între logică și interfață**  
  Arhitectura va respecta modelul MVC (Model-View-Controller) sau o variantă derivată, astfel încât logica aplicației, datele și interfața să fie bine izolate, ușurând testarea și extinderea.
* **Design consistent al interfeței**  
  Se vor aplica principii moderne de UI/UX, utilizând un sistem unitar de design (ex. Material UI sau Tailwind CSS) pentru asigurarea consistenței vizuale și funcționale în întreaga aplicație.
* **Politica de versiune și documentare continuă**  
  Orice modificare majoră în cod va fi însoțită de documentație tehnică și de utilizare. Se vor utiliza instrumente precum Git pentru controlul versiunilor și Swagger/OpenAPI pentru documentarea API-urilor.
* **Testare automată și validare continuă**  
  Se va implementa un sistem de testare automată (unitară și de integrare) cu integrare continuă (CI/CD), pentru a asigura calitatea codului și detectarea rapidă a erorilor.

1. **Politici specifice de proiectare**

* **Utilizarea infrastructurii cloud (ex. AWS, Azure)**  
  Pentru a beneficia de scalabilitate, securitate și ușurință în administrare, sistemul va fi implementat pe o infrastructură cloud cu servicii gestionate (ex. baze de date, stocare, procesare AI).

### Metode de dezvoltare

Instrucțiuni: Descrieți succint metoda sau abordarea folosită pentru designul sistemului și al software-ului (de exemplu, structurat, orientat pe obiecte, prototipare, J2EE, UML, XML, etc.). Dacă una sau mai multe metode formale/publicate au fost adoptate sau adaptate, includeți o referință către o descriere mai detaliată a acestor metode. Dacă au fost luate în considerare mai multe metode serios, menționați fiecare astfel de metodă, împreună cu o explicație succintă a motivului pentru care întreaga metodă sau o parte din ea a fost utilizată sau nu. Descrieți orice contingente care ar putea apărea în designul sistemului și al software-ului care ar putea schimba direcția de dezvoltare. Posibilitățile includ lipsa acordurilor de interfață cu agențiile externe sau arhitecturile instabile în momentul în care documentul este pregătit. Abordați orice soluții de rezervă posibile sau planuri alternative.

* Modelare orientata obiect, pentru a reutiliza codul, ajutand modularitatea si extensibilitatea
* Documentarea arhitecturii prin diagrame UML.
* Folosire de tehnologii moderne : FrontEnd (TypeScript, TailwindCSS) BackEnd (NodeJS) iar pentru baze de date MongoDB, pentru inteligenta artificiala, modele preantrenate cu integrare prin API uri

### Strategii de arhitectură

Instrucțiuni: Descrieți orice decizii și/sau strategii de design care afectează organizarea generală a sistemului și structurile sale de nivel superior. Aceste strategii ar trebui să ofere o înțelegere a principalelor abstractizări și mecanisme utilizate în arhitectura sistemului. Descrieți raționamentul utilizat pentru fiecare decizie și/sau strategie (posibil referindu-vă la obiectivele și principiile de design anterior menționate) și modul în care au fost echilibrate sau compensat obiectivele sau prioritățile de designAtunci când descrieți o decizie de design, discutați orice alte alternative semnificative care au fost luate în considerare și motivele pentru respingerea lor (precum și motivele pentru acceptarea alternativei în cele din urmă alese).

Exemple de decizii de design ar putea include (dar nu sunt limitate la) următoarele:

* Utilizarea unui anumit tip de produs (limbaj de programare, bază de date, bibliotecă, produs comercial disponibil pe piață etc.)
* Reutilizarea componentelor software existente pentru implementarea diferitelor părți/caracteristici ale sistemului
* Planuri viitoare pentru extinderea sau îmbunătățirea software-ului
* Paradigmele interfeței utilizatorului (sau modelele de intrare și ieșire ale sistemului)
* Paradigmele interfeței hardware și/sau software
* Detectarea și recuperarea erorilor
* Politicile de gestionare a memoriei
* Bazele de date externe și/sau gestionarea și persistența datelor
* Date distribuite sau control asupra unei rețele
* Abordări generalizate pentru control
* Concurgență și sincronizare
* Mecanisme de comunicare
* Managementul altor resurse

1. Arhitectura general

* Sistemul adoptă o arhitectură de tip client-server multi-strat (multi-tier), separând clar responsabilitățile între interfața cu utilizatorul (frontend), logica de afaceri (backend) și nivelul de stocare a datelor (baza de date). Această separare oferă numeroase avantaje:
* permite scalarea independentă a fiecărui strat;
* facilitează întreținerea și testarea componentelor;
* asigură o mai bună izolare a erorilor și o gestionare eficientă a resurselor.

1. Tehnologii și produse utilizate

* Frontend: React.js, împreună cu TypeScript și TailwindCSS, a fost ales pentru dezvoltarea interfeței utilizator. Această tehnologie oferă performanță ridicată, componentizare eficientă și o experiență fluidă pentru utilizatorii finali (medici, pacienți și administratori). A fost preferat în detrimentul alternativelor precum Angular sau Vue.js datorită comunității extinse și documentației bogate.
* Backend: Node.js cu Express.js a fost adoptat pentru dezvoltarea logicii aplicației, oferind suport pentru programarea asincronă și integrare facilă cu servicii externe și baze de date. Alternative precum Django sau Spring Boot au fost analizate, însă Node.js a fost preferat pentru consistența în ecosistemul JavaScript și timpul redus de dezvoltare.
* Baza de date: PostgreSQL a fost ales pentru stocarea datelor relaționale, inclusiv informații despre utilizatori, programări, simptome, rapoarte și interacțiuni. Sistemul permite extinderea ulterioară cu baze de date NoSQL (ex. MongoDB) pentru procesarea datelor semistructurate (ex. fișiere JSON provenite din analiza AI). Decizia reflectă echilibrul între performanță, stabilitate și conformitate cu standardele din domeniul medical.

1. Mecanisme de comunicare și control

* Comunicarea între frontend și backend se face prin API RESTful, folosind formatul JSON pentru transferul datelor. Această abordare facilitează interoperabilitatea și permite eventuală integrare cu aplicații terțe (ex. sisteme HIS sau ERP din spitale).
* Sistemul este gândit pentru extensibilitate modulară, astfel încât noi funcționalități – precum module de telemedicină sau integrări cu dispozitive IoT – să poată fi adăugate fără a afecta funcționalitățile existente.
* Controlul asupra datelor și logicii aplicației este gestionat prin principii de separare a responsabilităților (Separation of Concerns) și Inversiunea Controlului (IoC), folosind design patterns precum MVC (Model-View-Controller) în backend.

1. 1.2.4 Persistența și securitatea datelor

* Gestionarea datelor sensibile este realizată conform reglementărilor GDPR. Platforma include:
* criptarea parolelor prin algoritmi SHA-256 + salt;
* criptarea datelor critice în bază de date;
* autentificare JWT (JSON Web Token) cu expirare și reîmprospătare controlată;
* backup periodic și salvare redundantă în cloud securizat.

1. Concurență și scalabilitate

* Pentru a gestiona mai mulți utilizatori simultan, backend-ul este scalabil orizontal, cu suport pentru implementare în containere (ex. Docker) și orchestrare cu Kubernetes în medii de producție.
* Mecanismele de blocare optimistă sunt utilizate pentru evitarea coliziunilor de date la modificarea simultană a resurselor. În plus, pentru sarcini intensive (ex. analiza AI a simptomelor), sunt utilizate queue-uri asincrone (ex. RabbitMQ sau Redis Queue) pentru decuplarea fluxurilor.

## Arhitectura Sistemului și Proiectarea Arhitecturii

Instrucțiuni: Descrieți arhitectura sistemului, modul în care aplicația interacționează cu alte aplicații. Nu este neapărat nevoie să detaliați cum funcționează aplicația în sine, ci mai degrabă cum datele corespunzătoare sunt corect transferate între componente. Furnizați o prezentare generală a modului în care funcționalitatea și responsabilitățile sistemului au fost partitionate și apoi atribuite sub-sisteme sau componente. Nu intrați prea mult în detalii despre componentele individuale în această secțiune. O secțiune ulterioară a documentului va furniza descrieri detaliate ale componentelor. Scopul principal aici este de a obține o înțelegere generală a modului și motivelor pentru care sistemul a fost descompus și cum părțile individuale lucrează împreună pentru a furniza funcționalitatea dorită.

Descrieți cum sistemul a fost dezmembrat în componente/subsistemele sale (identificând fiecare componentă/subsistem de nivel superior și rolurile/responsabilitățile atribuite acestuia). Descrieți modul în care componentele de nivel superior colaborează între ele pentru a obține rezultatele dorite. Furnizați o anumită justificare pentru alegerea acestei decompuneri particulare a sistemului (poate discutați alte decompuneri propuse și motivele pentru care au fost respinse).

Faceți uz de tiparele de proiectare ori de câte ori este posibil, fie pentru a descrie părți ale arhitecturii (în format de tipar), fie pentru a face referire la elementele arhitecturii care le utilizează. Furnizați o justificare pentru alegerea unui algoritm sau idiom de programare (sau tipar de proiectare) pentru a implementa părți ale funcționalității sistemului.

Această secțiune conturează designul arhitecturii sistemului și hardware-ului.

Descopunere in subsisteme:

1. Frontend:

 Componentele de frontend sunt responsabile pentru interacțiunea utilizatorilor cu platforma. Acestea permit utilizatorilor să vizualizeze datele medicale, să încarce simptome, să solicite analize ale AI-ului, să gestioneze programări, etc.

 Utilizatorii pot accesa platforma printr-o aplicație web accesibilă prin browser, bazată pe tehnologiile React.js și TypeScript.

 Colaborează cu backend-ul prin API RESTful pentru a trimite și a primi datele necesare.

2. Backend

* Backend-ul este responsabil pentru implementarea logicii de afaceri a aplicației. Acesta gestionează autentificarea, autorizarea, procesarea și analiza datelor medicale, și comunicarea cu baza de date.
* Tehnologiile utilizate includ Node.js cu Express.js, care sunt potrivite pentru gestionarea cererilor asincrone și procesarea rapidă a datelor.
* Colaborează cu frontend-ul pentru a expune API-uri RESTful și cu baza de date pentru persistarea datelor.

3. Baza de date

* Baza de date relațională este utilizată pentru stocarea datelor medicale, a utilizatorilor și a altor informații relevante. Sistemul va susține stocarea fișierelor JSON și a altor date semistructurate în caz de extindere viitoare.
* Baza de date este integrată cu backend-ul pentru a permite accesul rapid și sigur la informațiile necesare.

4.Componentele de AI și Procesare a Datelor:

* Acestea sunt utilizate pentru analiza simptomelor pacienților și generarea de recomandări pe baza algoritmilor AI, implementați pe partea de backend. Modelele de machine learning sunt utilizate pentru a sugera tratamente medicale.
* Datele necesare pentru antrenarea și inferența modelelor sunt preluate din baza de date și sunt procesate în backend pentru a returna recomandările pentru utilizatori.
* De asemenea, aceste componente vor suporta integrarea cu surse externe de date medicale, dacă se impune, precum și integrarea cu dispozitive IoT pentru monitorizarea în timp real a pacienților.
  + - 1. 5. **Interacțiunea dintre componente**

Fiecare dintre aceste componente interacționează într-un flux de date bine definit:

* **Frontend-ul** trimite cereri de informații și actualizări către **backend** prin API RESTful.
* **Backend-ul** procesează cererile, interacționează cu **baza de date** pentru a obține sau salva informațiile necesare și, dacă este necesar, utilizează componentele de AI pentru analiza simptomelor sau pentru generarea de rapoarte.
* **Componentele de AI** folosesc datele disponibile pentru a furniza sugestii sau recomandări, iar rezultatele sunt trimise înapoi la **frontend** pentru a fi prezentate utilizatorilor.

### Vedere logică

Instrucțiuni: Introduceți orice vederi logice relevante sau furnizați o referință către locul în care sunt stocate.

A diagram of a company

AI-generated content may be incorrect.

### Arhitectură hardware

Instrucțiuni: Descrieți hardware-ul și organizația generală a sistemului, indicând dacă sistemul de procesare este distribuit sau centralizat. Identificați tipul, numărul și locația tuturor componentelor hardware, inclusiv serverele de prezentare, aplicație și date și orice dispozitive periferice (de exemplu, load balancers, acceleratori SSL, comutatoare, firewall-uri), cu o descriere succintă a fiecărui element și diagrame care arată conectivitatea între componente, împreună cu firewall-urile, porturile și benzile de rețea utilizate (de exemplu, banda de management). Includeți estimările resurselor pentru capacitatea procesorului, memoria, stocarea online și stocarea auxiliară.

1. **Componente hardware**
2. **Server de Prezentare (Frontend Server)**:
   * **Tip**: Server dedicat pentru gestionarea interfeței cu utilizatorul.
   * **Funcție**: Acest server este responsabil pentru procesarea cererilor de la utilizatori (vizualizarea datelor, interacțiunea cu interfața AI), afișarea interfeței și trimiterea cererilor către backend.
   * **Număr**: 2 servere de prezentare (pentru scalabilitate și redundanță).
   * **Locație**: Centre de date regionale pentru a minimiza latența.
   * **Specificații hardware**:
     + Procesor: 8 core-uri, 3.0 GHz.
     + Memorie RAM: 16 GB.
     + Stocare: SSD de 500 GB.
   * **Sistem de operare**: Linux (Ubuntu sau CentOS).
3. **Dispozitive Periferice**:

* **Tip**: Dispozitive de stocare externe (backup).
* **Funcție**: Asigură backup-ul periodic al datelor din baza de date principală pentru a preveni pierderile de date.
* **Număr**: 2 dispozitive de stocare externe.
* **Specificații hardware**:
  + Stocare: 10 TB HDD.
  + Backup automatizat la intervale de 24 ore.

1. **Estimările resurselor hardware**
2. **Capacitate procesor**:
   * Estimare: 64 core-uri pentru procesarea cererilor de utilizator, procesarea datelor și rularea algoritmilor AI.
3. **Memorie RAM**:
   * Estimare: 128 GB RAM pentru serverele de aplicație, backend și baze de date pentru a asigura procesarea rapidă a cererilor și analiza AI.
4. **Stocare**:
   * **Stocare SSD**: 6 TB pentru a sprijini stocarea rapidă și accesul la datele utilizatorilor.
   * **Stocare auxiliară (backup)**: 10 TB pentru backup-ul periodic al datelor.

### Arhitectură software

Instrucțiuni: Descrieți toate componentele software. Enumerați elemente logice precum componente (de exemplu, JSP în stratul de prezentare, JNDI în stratul de aplicație, EJB și JDBC în stratul de date), platforme de baze de date, limbaje de programare, compilatoare, utilitare, sisteme de operare, software de comunicații, instrumente de inginerie software asistate de calculator, produse software externe – software comercial disponibil pe piață, framework-uri open source etc., cu o scurtă descriere a funcției fiecărui element și a oricăror informații de identificare, cum ar fi producătorul, numărul versiunii, numărul și tipurile de licențe necesare etc., dacă este cazul. Identificați toate elementele de configurare a software-ului computerului și interfețele de programare a aplicațiilor (APIs), inclusiv numele, tipul, scopul și funcția pentru fiecare; interfețele, mesajele și protocoalele pentru acele elemente; și raționamentul pentru designul arhitectural al software-ului.

Includeți module de software care sunt funcții, subrutine sau clase. Utilizați diagrame ierarhice funcționale, diagrame de organizare structurată (de exemplu, diagrame de structură), sau diagrame orientate pe obiecte care arată diferitele niveluri de segmentare până la nivelul cel mai jos. Toate caracteristicile din diagrame ar trebui să aibă numere de referință și nume.

Dacă este necesar, descrieți cum o componentă a fost împărțită în subcomponente, precum și relațiile și interacțiunile dintre subcomponente. Continuați în cât mai multe niveluri/subsecțiuni de discuție cât este necesar pentru a oferi o înțelegere de nivel înalt a întregului sistem sau subsistem, lăsând detaliile pentru includerea într-o secțiune ulterioară a documentului. Includeți diagrame de flux de date și furnizează fluxul fizic al proceselor și datelor legate de fluxul logic al proceselor și datelor dezagregat la nivelul procesului primitiv (descriind cum fiecare intrare este procesată/transformată în ieșirea rezultată). Dacă există părți ale sistemului care au existat deja înainte de începerea acestui efort de dezvoltare, atunci descrieți doar relațiile și interacțiunile dintre părțile vechi și cele noi. Părțile preexistente care sunt modificate sau îmbunătățite trebuie descrise doar în măsura în care este necesar pentru a oferi o înțelegere suficientă a naturii modificărilor care se fac.

Arhitectura software a platformei „Web pentru Analiza și Vizualizarea Datelor Medicale” este organizată pe trei straturi principale: stratul de prezentare (frontend), stratul de logică de afaceri (backend) și stratul de date (bază de date).

**Stratul de prezentare** este responsabil de interfața utilizatorului, adică modul în care pacienții, doctorii, administratorii și inteligența artificială interacționează cu platforma. Pentru acest strat s-au folosit tehnologii moderne precum React.js pentru construirea interfeței, Tailwind CSS pentru stilizare, Axios pentru apeluri HTTP și Vite pentru rularea și compilarea eficientă a aplicației.

**Stratul logicii de afaceri**, adică backendul, este implementat cu Node.js și Express.js. Acesta gestionează autentificarea, logica platformei și comunicarea cu baza de date. Pentru autentificare este utilizat JWT (JSON Web Token). Analiza simptomelor este realizată prin microservicii scrise în Python folosind FastAPI. Pentru generarea de răspunsuri conversaționale, platforma se bazează pe un API extern – DeepSeek AI. De asemenea, pentru documentarea API-ului este utilizat Swagger (OpenAPI).

**Stratul de date** folosește PostgreSQL, o bază de date relațională robustă și open-source. Pentru interacțiunea dintre backend și baza de date este utilizat Prisma ORM, care mapează datele între codul aplicației și structura bazei de date. Opțional, Redis poate fi utilizat pentru cache – de exemplu, pentru salvarea sesiunilor sau a contextului conversațiilor AI.

### Arhitectura informațiilor

Instrucțiuni: Descrieți informațiile care vor fi stocate în sistem (de exemplu, informații despre beneficiari, date despre cereri etc.). Identificați dacă vreuna dintre informații este informație cu caracter sensibil.

Identificați toate datele (precum și formatul datelor - hârtie, introducere manuală, date electronice) furnizate sistemului, precum și cine/ce furnizează datele.

1. **Tipuri de informații stocate:**
2. **Date despre utilizatori (pacienți, doctori, administratori):**
   * Nume complet
   * Adresă de e-mail
   * Număr de telefon (opțional)
   * Parolă (criptată)
   * Rolul utilizatorului în sistem
   * Tokenuri de autentificare (ex. JWT)  
     **→** Acestea sunt informații personale și, în anumite contexte, sensibile.
3. **Date medicale ale pacienților:**
   * Simptome introduse manual
   * Istoric de simptome și interacțiuni cu AI-ul
   * Recomandări și posibile diagnostice generate de AI
   * Programări la doctori
   * Note medicale completate de doctori în urma consultațiilor  
     **→** Acestea sunt informații medicale și sunt considerate date cu caracter sensibil conform legislației GDPR.
4. **Date despre programări:**
   * Data și ora
   * ID-ul pacientului
   * ID-ul doctorului
   * Statusul programării (în așteptare, confirmată, finalizată etc.)
5. **Date administrative:**
   * Jurnale de activitate pentru audit
   * Setări ale aplicației
   * Statistici de utilizare a platformei
   * Feedback de la utilizatori (opțional)
6. **Formatul datelor:**

* **Date introduse manual:**
  + De către pacienți (simptome, conturi, programări)
  + De către doctori (note medicale)
  + De către administratori (configurări sau conturi speciale)
* **Date generate automat:**
  + De microserviciile AI (analiză simptome, sugestii)
  + De backend (loguri, tokenuri, statistici)
* **Date electronice importate:**
  + (opțional) Fișiere PDF medicale, rezultate de laborator sau rapoarte importate de pacient sau doctor

1. **Sursa datelor:**

* **Pacientul** furnizează datele personale, simptomele și eventual documentele medicale.
* **Doctorul** adaugă notițe medicale și actualizează programări.
* **Administratorul** configurează platforma și poate introduce date administrative.
* **AI-ul** contribuie cu date generate prin analiza simptomelor.

### Arhitectura de comunicații interne

Instrucțiuni: Furnizați o descriere detaliată a rețelei de comunicații a sistemului, indicând arhitectura de comunicații implementată și modul în care componentele sistemului sunt conectate. Includeți descrieri ale echipamentelor necesare (de exemplu, hub-uri, routere, transmițătoare, module de comunicații, firewall-uri, porturi etc.). Furnizați o diagramă care să ilustreze fluxul de comunicații între componentele sistemului și subsisteme.

Includeți estimări ale resurselor necesare pentru capacitatea rețelei de comunicații (LAN și WAN) necesară pentru instalarea și executarea fiecărei aplicații pe fiecare platformă.

Front-end-ul aplicației (clientul) este accesat printr-un browser web, iar cererile sunt trimise către serverul de aplicație prin protocolul HTTPS.

Back-end-ul gestionează logica aplicației și comunică cu baza de date, serviciul AI și serviciile de autentificare/autorizare.

Microserviciile AI sunt accesate prin REST API intern, într-un mediu containerizat (ex: Docker).

Baza de date este găzduită pe un server dedicat sau într-un container separat, accesat doar din interiorul rețelei backend.

### Diagrama de arhitectură a sistemului

Instrucțiuni: Utilizând proiectarea hardware, software, de comunicații și de informații descrise mai sus, redați structura generală și integrată a sistemului.

## Proiectarea sistemului

### Proiectarea bazei de date

Instrucțiuni: Descrieți proiectarea tuturor fișierelor de sistem de gestionare a bazelor de date (DBMS) și a fișierelor non-DBMS asociate cu sistemul. Furnizați un dicționar de date cuprinzător care să conțină numele elementului de date, tipul, lungimea, sursa, regulile de validare, întreținerea (creare, citire, actualizare, ștergere (CRUD)), stocarea datelor, ieșirile, aliasurile și descrierea.

1. **Arhitectura generală**

Baza de date este împărțită în mai multe colecții, corespunzătoare actorilor și fluxurilor de date din sistem:

* **Users** – pentru gestionarea conturilor utilizatorilor (pacienți, doctori, administratori).
* **Appointments** – programările pacienților cu doctorii.
* **Symptoms** – datele despre simptomele introduse de pacienți și trimise spre analiză AI.
* **AI\_Responses** – răspunsurile generate de componenta AI.
* **MedicalRecords** – istoricul medical al pacienților.
* **SystemLogs** – pentru logarea acțiunilor importante din sistem.

"\_id": ObjectId,

"name": "Ion Popescu", string

"email": "ion@example.com", string

"role": "pacient", // sau "doctor", "admin", enum

"passwordHash": "...",

"createdAt": ISODate(),

"lastLogin": ISODate()

Datele stocate includ **informații sensibile cu caracter personal și medical**, ceea ce impune următoarele măsuri:

* Criptarea parolelor cu algoritmi puternici (ex: bcrypt).
* Controlul accesului la nivel de rol (ACL – Access Control List).
* Limitarea interogărilor pentru utilizatorii standard doar la propriile date.
* Auditarea acțiunilor sensibile (loguri).

1. **Formatul datelor**

* **Date electronice** – introduse prin interfață web și API-uri REST.
* **Date structurate** – organizate în JSON și stocate în documente MongoDB.
* **Informațiile** provin de la utilizatori (prin UI), componenta AI și sistemele externe (dacă este cazul).

#### Obiecte de date și structuri de date rezultante

Instrucțiuni: Pentru fiecare obiect de date funcțional, specificați structura/ile de date care va/vor fi utilizată/e pentru a stoca și procesa datele. Descrieți orice structuri de date care sunt o parte importantă a sistemului, inclusiv structurile de date majore care sunt transmise între componente.

Platforma gestionează mai multe **obiecte de date funcționale** care sunt esențiale pentru funcționarea corectă a sistemului. Fiecare dintre aceste obiecte este stocat și procesat sub formă de **document JSON**, utilizând colecțiile MongoDB. Structurile de date sunt proiectate astfel încât să susțină eficient operațiunile CRUD și să permită integrarea între componentele aplicației (interfață, backend, AI).

User:

"\_id": ObjectId,

"name": "Maria Ionescu",

"email": "maria@example.com",

"role": "pacient",

"passwordHash": "...",

"createdAt": ISODate()

Programare:

"\_id": ObjectId,

"patientId": ObjectId,

"doctorId": ObjectId,

"date": ISODate(),

"status": "confirmat"

Istoric medical :

"\_id": ObjectId,

"patientId": ObjectId,

"doctorId": ObjectId,

"recordType": "diagnostic",

"details": "Tensiune arterială crescută",

"date": ISODate(),

"attachments": ["doc1.pdf", "scan.jpg"]

#### Fișiere și baze de date

Instrucțiuni: Creați un model de date fizic care descrie stocarea și manipularea datelor în cadrul arhitecturii sistemului. Descrieți structurile de fișiere și locațiile acestora.

1. **Structura fizică a datelor:**
2. **Colecția Users**
   * **Fișier de stocare**: users.bson
   * **Locație**: Directorul de stocare al MongoDB (/data/db sau locație specifică configurată)
   * **Descriere**: Stochează informațiile despre utilizatori (pacienți, doctori, administratori).
   * **Indexuri**:
     + email: Index unic pentru a accelera autentificarea.
     + role: Index pentru a filtra utilizatorii după rol.
3. **Colecția Appointments**

* **Fișier de stocare**: appointments.bson
* **Locație**: Directorul de stocare al MongoDB
* **Descriere**: Stochează datele despre programările pacienților cu doctorii, inclusiv statusul și data.
* **Indexuri**:
  + patientId: Index pentru căutarea programărilor unui anumit pacient.
  + doctorId: Index pentru căutarea programărilor unui anumit doctor.
  + date: Index pe dată pentru a permite căutarea programărilor într-o anumită perioadă.

1. **Colecția Symptoms**

* **Fișier de stocare**: symptoms.bson
* **Locație**: Directorul de stocare al MongoDB
* **Descriere**: Stochează simptomele raportate de pacienți și informațiile legate de analiza AI.
* **Indexuri**:
  + patientId: Index pentru căutarea simptomelor unui anumit pacient.
  + submittedAt: Index pentru a căuta simptomele pe baza datei de raportare.

1. **Colecția MedicalRecords**

* **Fișier de stocare**: medical\_records.bson
* **Locație**: Directorul de stocare al MongoDB
* **Descriere**: Stochează fișele medicale, inclusiv istoricul tratamentelor și diagnosticele anterioare.
* **Indexuri**:
  + patientId: Index pentru a căuta istoricul medical al unui pacient.

##### Baze de date

Instrucțiuni: Furnizați proiectarea detaliată a fișierelor DBMS.

**Colecția Users**

* **Fișierul asociat**: users.bson
* **Descriere**: Stochează datele utilizatorilor, inclusiv pacienți, doctori și administratori.
* **Tipuri de date**: String (pentru nume, email), Date (pentru data creării contului), ObjectId (pentru identificatori).
* **Indexuri**:
  + email: Index unic pentru autentificare.
  + role: Index pentru a filtra utilizatorii pe baza rolului.

**Colecția Appointments**

* **Fișierul asociat**: appointments.bson
* **Descriere**: Stochează informațiile despre programările pacienților cu doctorii.
* **Tipuri de date**: ObjectId (pentru referințele la utilizatori), Date (pentru data programării), String (pentru statusul programării).
* **Indexuri**:
  + patientId: Index pentru căutarea programărilor unui pacient.
  + doctorId: Index pentru căutarea programărilor unui doctor.
  + date: Index pe data pentru căutarea rapidă a programărilor.

**Colecția MedicalRecords**

* **Fișierul asociat**: medical\_records.bson
* **Descriere**: Stochează istoricul medical al pacienților, incluzând diagnosticele și tratamentele anterioare.
* **Tipuri de date**: ObjectId (pentru referințele la pacienți și doctori), String (pentru tipul fișei, de exemplu, diagnostic), Date (pentru data fiecărei fișe), Array (pentru fișierele atașate, de exemplu, PDF-uri sau imagini).
* **Indexuri**:
  + patientId: Index pentru a căuta istoricul unui pacient.

MongoDB utilizează fișiere BSON pentru a stoca datele. Fiecare colecție este stocată într-un fișier fizic. De exemplu, colecția users va fi stocată într-un fișier denumit users.bson și va fi localizată în directorul de date al MongoDB (/data/db/ sau alt director configurat).

* **Fișierele de index**: MongoDB creează fișiere separate pentru indexuri, de obicei în aceleași locații ca și fișierele BSON.
* **Redundanță și replicare**: În funcție de configurarea MongoDB, poate fi folosită replicarea pentru a asigura backup și redundanță. Fiecare replică va avea fișierele sale de bază de date și indexuri.

##### Fișiere non-DBMS

Instrucțiuni: Furnizați descrierea detaliată a tuturor fișierelor non-DBMS și includeți o descriere narativă a utilizării fiecărui fișier care identifică dacă fișierul este utilizat pentru intrare, ieșire sau ambele, și dacă fișierul este un fișier temporar. De asemenea, oferiți o indicație a modulului care citește și scrie fișierul și includeți structurile de fișiere (faceți referire la dicționarul de date). În funcție de caz, informațiile despre structura fișierului ar trebui să includă următoarele:

* Structurile de înregistrare, cheile de înregistrare sau indexele și elementele de date referite în înregistrări
* Lungimea înregistrării (lungime fixă sau maximă variabilă) și factorii de blocare
* Estimarea dimensiunii fișierului sau volumului de date din fișier
* Definiția frecvenței de actualizare a fișierului (dacă fișierul face parte dintr-un sistem bazat pe tranzacții online, furnizați numărul estimat de tranzacții per unitate de timp și media statistică, modul și distribuția acestor tranzacții.)
* Specificații de backup și recuperare
* **Documente incarcate de utilizator** : ceste fișiere sunt stocate pe server și conțin documente care sunt încărcate de utilizatori, cum ar fi imagini de identificare, fișiere PDF cu istoricul medical, rapoarte de laborator, etc. Aceste fișiere sunt stocate într-un director organizat pe bază de ID-uri unice ale utilizatorilor
* **Fisiere log** : Aceste fișiere sunt utilizate pentru a salva informații de jurnal (loguri) despre activitățile aplicației, erorile de sistem, informațiile de diagnostic și informațiile de acces. Acestea sunt utile pentru depanarea aplicației și pentru monitorizarea activității utilizatorilor.
* Fisiere de configurare: Aceste fișiere stochează setările de configurare pentru aplicație, inclusiv informațiile de autentificare, parametrii de conectare la MongoDB, setările de API, chei de criptare, etc.

### Conversii de date

Instrucțiuni: Inserați orice documente care descriu conversiile de date necesare sau furnizați o referință către locul unde sunt stocate.

1. Conversie intre formatele fisierelor: Atunci când utilizatorii încarcă fișiere de tip imagine sau documente PDF, sistemul poate necesita conversia acestora într-un format uniform, care poate fi ușor procesat sau vizualizat de către aplicație.

2.Conversia din JSON in BSON

3.Conversia intre datele de la utilizator: atele introduse de utilizator, precum simptomele sau istoricul medical, pot fi capturate într-un format structurat și trebuie să fie convertite într-un format standardizat pentru a fi procesate de AI-ul platformei.

4.Conversia de date pentru importuri si exporturi: importul dintr-un fisier csv sau exportul de Exceluri

### Interfețe utilizator

Instrucțiuni: Furnizați o descriere a fiecărei clase de utilizatori sau roluri asociate cu sistemul. O clasă de utilizatori se diferențiază în funcție de modurile în care utilizatorii interacționează cu sistemul sau situația propusă. Factorii care disting o clasă de utilizatori includ responsabilitățile comune, nivelurile de competență, activitățile de lucru și modurile de interacțiune cu sistemul. În acest context, un utilizator este oricine interacționează cu sistemul propus, inclusiv utilizatori operaționali, personal de introducere a datelor, operatori de sistem, personal de suport operațional, cei care întrețin sistemul și formatori. Pentru fiecare clasă de utilizatori, furnizați estimări ale numărului total de utilizatori anticipați, un număr maxim de utilizatori simultani și numărul de utilizatori externi.

Pacientul este principalul utilizator al platformei. Acesta poate accesa interfața pentru a se programa la consultații, a trimite simptome către AI și a interacționa cu AI pentru a înțelege mai bine starea de sănătate. De asemenea, pacientul poate vizualiza istoricul medical și primește recomandări sau sfaturi de la medicul său.

Medicii sunt utilizatori cheie ai platformei, având rolul de a analiza simptomele pacienților, de a gestiona programările și de a oferi tratamente recomandate pe baza datelor furnizate de AI. Aceștia pot vizualiza istoricul medical al pacienților și pot actualiza informațiile medicale ale acestora.

Administratorul este responsabil pentru gestionarea și întreținerea platformei. Acesta are drepturi extinse de acces, putând gestiona utilizatorii, monitoriza performanța aplicației și efectua actualizări la sistemul informatic. De asemenea, administratorul gestionează toate aspectele legate de securitatea și integritatea datelor.

Deși AI-ul nu este un utilizator în sensul tradițional, acesta joacă un rol crucial în analiza simptomelor și în recomandarea tratamentelor pentru pacienți. AI-ul utilizează algoritmi de învățare automată și procesare a limbajului natural pentru a analiza datele pacientului și a oferi sfaturi și recomandări de tratament.

#### Intrări

Instrucțiuni: Furnizați o descriere a mijloacelor de intrare folosite de utilizator/operator pentru a furniza informații sistemului.

Arătați o mapare către fluxurile de date la nivel înalt (de exemplu, ecranele de introducere a datelor, cititoarele optice de caractere, scanerele de coduri de bare, etc.). Dacă este cazul, tipurile de înregistrări de intrare, structurile de fișiere și structurile de baze de date furnizate în secțiunea pentru Proiectarea datelor pot fi referite. Includeți definițiile elementelor de date sau faceți referire la dicționarul de date.

Furnizați aspectul tuturor ecranelor de intrare de date sau interfețe grafice (GUI-uri) (de exemplu, ferestre). Definiți toate elementele de date asociate fiecărui ecran sau GUI sau faceți referire la dicționarul de date. Furnizați criterii de editare pentru elementele de date, inclusiv valori specifice, interval de valori, obligatoriu/opțional, valori alfanumerice și lungime. Discutați controalele de introducere a datelor pentru a preveni trecerea peste editare. Discutați mesajele diverse asociate cu intrările utilizatorului/operatorului.

**Ecrane de introducere a datelor pentru pacienți:**

**1. Date personale**

**2. Simptome**

**Ecrane de introducere date pentru medici:**

**1.Date pacienti**

**2.Simptome si diagnostic**

**Ecran introducere date pentru administrator:**

**1.Conturile de utilizator**

**2.Setari de system**

#### Ieșiri

Instrucțiuni: Ieșirile sistemului includ rapoarte, ecrane de afișare a datelor și interfețe grafice, rezultatele interogărilor, etc. Fișierele de ieșire descrise în secțiunea pentru Proiectarea Datelor pot fi referite. Ar trebui să fie furnizate următoarele, dacă este potrivit:

• Identificarea codurilor și numelor pentru rapoarte și ecranele de afișare a datelor

• Descrierea conținutului raportului și al ecranului (furnizați o reprezentare grafică a fiecărei structuri și definiți toate elementele de date asociate cu structura sau faceți referire la dicționarul de date)

• Descrierea scopului ieșirii, inclusiv identificarea utilizatorilor principali

• Descrierea oricăror restricții de acces sau considerații de securitate

1. **Rapoarte pentru medici**
2. **Identificare cod/nume:**

* **Raport de diagnosticare pacient** – Cod: RP001

1. **Descrierea conținutului raportului:**

* **Date pacient**: Nume complet, vârstă, sex
* **Simptome**: Detalii despre simptomele semnalate de pacient, data debutului acestora, severitatea
* **Istoric medical**: Detalii despre tratamentele și diagnosticele anterioare
* **Recomandări tratamente**: Tratamentul propus de AI, urmat de o analiză făcută de medic
* **Diagnostic final**: Diagnosticul stabilit de medic

1. **Reprezentare grafică:**

Raportul poate fi generat în format PDF sau HTML, structurat pe secțiuni clare: informații pacient, simptome, diagnostic, tratament recomandat și diagnostic final. Fiecare secțiune va conține titluri și subsecțiuni pentru claritate.

1. **Scopul ieșirii:**

* **Utilizatori principali**: Medici, pentru a evalua istoricul și recomandările AI, precum și pentru a face ajustări în tratamentele propuse.

1. **Restricții de acces și considerații de securitate:**

* **Restricții**: Acest raport va fi accesibil doar medicilor autorizați și pacienților lor (prin intermediul unui portal securizat). Rapoartele sunt criptate pentru a proteja datele sensibile ale pacientului.
* **Considerații de securitate**: Accesul la rapoarte este limitat prin autentificare multi-factor (MFA) pentru a asigura confidențialitatea datelor medicale.

1. **Ecran de vizualizare a datelor pentru pacienți**
2. **Identificare cod/nume:**

* **Vizualizare profil pacient** – Cod: VP001

1. **Descrierea conținutului ecranului:**

* **Informații personale**: Nume, vârstă, sex, adresa de e-mail, număr de telefon
* **Simptome**: Lista simptomelor curente, gravitatea acestora
* **Istoric medical**: Sumarizat pe baza consultațiilor anterioare
* **Recomandări AI**: Propuneri pentru tratamente bazate pe analizele AI

1. **Reprezentare grafică:**

Ecranul va fi structurat în mai multe secțiuni:

* **Informații personale** în partea superioară
* **Simptome curente** afișate sub formă de listă
* **Istoric medical** într-un tab separabil
* **Recomandări AI** într-o fereastră dedicată, cu opțiunea de a adăuga comentarii

1. **Scopul ieșirii:**

* **Utilizatori principali**: Pacienți, pentru a vizualiza istoricul lor medical și recomandările făcute de AI. Acest ecran ajută pacienții să înțeleagă mai bine situația lor medicală și să fie informați cu privire la pașii următori.

1. **Restricții de acces și considerații de securitate:**

* **Restricții**: Accesul la aceste informații este restricționat doar pacientului respectiv, iar login-ul se face cu autentificare securizată. Este folosit un sistem de criptare pentru datele sensibile.
* **Considerații de securitate**: Aceste informații sunt disponibile numai printr-o sesiune securizată și nu pot fi partajate fără acordul explicit al pacientului.

1. **Interfața grafică pentru administratori**
2. **Identificare cod/nume:**

* **Dashboard administrator** – Cod: DA001

1. **Descrierea conținutului ecranului:**

* **Utilizatori activi**: Listă cu toți utilizatorii activi (pacienți, medici), statusul acestora
* **Setări platformă**: Posibilitatea de a actualiza setările de securitate, de a adăuga sau elimina utilizatori, de a modifica parametrii sistemului
* **Statistici**: Grafice și rapoarte despre activitatea platformei (număr de pacienți, număr de consultații, etc.)

1. **Reprezentare grafică:**

Ecranul va include un **dashboard** interactiv cu secțiuni clare:

* **Utilizatori activi**: tabel cu informații relevante pentru fiecare utilizator
* **Setări platformă**: butoane de acțiune pentru gestionarea utilizatorilor și configurarea platformei
* **Statistici platformă**: grafice și diagrame care prezintă activitatea curentă a platformei

1. **Scopul ieșirii:**

* **Utilizatori principali**: Administratori, care vor gestiona platforma, vor urmări activitatea utilizatorilor și vor actualiza setările platformei.

1. **Restricții de acces și considerații de securitate:**

* **Restricții**: Doar administratorii autorizați vor avea acces la acest dashboard. Se va implementa autentificare multi-factor pentru a preveni accesul neautorizat.
* **Considerații de securitate**: Datele vizualizate pe acest ecran sunt critice pentru funcționarea platformei, iar accesul va fi limitat conform rolurilor și permisiunilor din sistem.

### Proiectarea interfețelor cu utilizatorul

Instrucțiuni: Introduceți interfețele proiectate împreună cu descrierea lor.

## Scenarii de utilizare

Instrucțiuni: Descrieți funcționalitatea generală a sistemului din perspectiva utilizatorilor și furnizați un flux de execuție sau operațional al sistemului prin scenarii operaționale care oferă descrieri pas cu pas despre modul în care sistemul propus ar trebui să funcționeze și să interacționeze cu utilizatorii săi și interfețele externe într-un set dat de circumstanțe. Scenariile leagă împreună toate părțile sistemului, utilizatorii și alte entități prin descrierea modului în care interacționează, și pot fi folosite și pentru a descrie ce nu ar trebui să facă sistemul.

Scenariile de utilizare ar trebui să fie descrise pentru toate modurile operaționale, tranzacțiile și toate clasele de utilizatori identificate pentru sistemul propusFiecare scenariu ar trebui să includă evenimente, acțiuni, stimuli, informații și interacțiuni în măsura în care este necesar pentru a oferi o înțelegere cuprinzătoare a aspectelor operaționale ale sistemului propus.

Un pacient dorește să își creeze un cont pentru a accesa platforma și pentru a beneficia de serviciile oferite, inclusiv trimiterea simptomelor și consultațiile cu AI.

1. **Pași:**
2. **Acces la platformă**:
   * Pacientul accesează aplicația web și vizualizează pagina principală.
3. **Creare cont**:
   * Pacientul apasă pe butonul „Creare cont” și este direcționat către formularul de înregistrare.
4. **Completați informațiile personale**:
   * Pacientul completează câmpurile obligatorii: nume, adresă de e-mail, număr de telefon, dată naștere, sex și parole.
5. **Verificare și validare date**:
   * Sistemul validează datele introduse și verifică dacă adresa de e-mail nu există deja în sistem.
6. **Creare cont**:
   * Dacă informațiile sunt valide, contul este creat și pacientul primește un e-mail de confirmare pentru activarea contului.
7. **Autentificare**:
   * Pacientul se autentifică cu e-mailul și parola pentru a accesa platforma.
8. **Evenimente, acțiuni și stimuli:**

* **Eveniment**: Pacientul completează formularul de înregistrare.
* **Acțiune**: Sistemul validează informațiile.
* **Stimuli**: Validarea datelor de intrare, trimiterea unui e-mail de activare.

1. **Interacțiuni:**

* **Utilizator (pacient)**: Introduce datele personale și accesează platforma.
* **Sistem**: Verifică datele și validează contul.

1. **Scenariul de utilizare: Trimiterea simptomelor de către pacient**
2. **Descriere:**

După crearea contului, pacientul dorește să trimită simptomele sale către AI pentru o analiză preliminară.

1. **Pași:**
2. **Autentificare**:
   * Pacientul se autentifică în platformă cu datele sale de utilizator.
3. **Accesare secțiune „Simptome”**:
   * Pacientul accesează secțiunea „Adăugare simptome”.
4. **Completare formular simptomatologie**:
   * Pacientul completează un formular cu simptomele resimțite: tipul de durere, locația, intensitatea, durerea asociată cu alte simptome etc.
5. **Trimite simptome**:
   * După completarea formularului, pacientul trimite informațiile către AI pentru analiză.
6. **Confirmare trimite simptome**:
   * Sistemul confirmă trimiterea și informează pacientul că răspunsul va veni în curând.
7. **Evenimente, acțiuni și stimuli:**

* **Eveniment**: Pacientul trimite simptomele.
* **Acțiune**: AI analizează simptomele și sugerează un tratament.
* **Stimuli**: Notificare de confirmare a trimiterii simptomelor.

1. **Interacțiuni:**

* **Utilizator (pacient)**: Completează și trimite formularul.
* **Sistem (AI)**: Procesează informațiile și le trimite pentru analiza detaliată.

1. **Scenariul de utilizare: Consultarea unui medic**
2. **Descriere:**

Un pacient dorește să programeze o consultație cu un medic pentru a discuta despre simptomele sale și recomandările AI.

1. **Pași:**
2. **Accesare secțiune „Programează consultație”**:
   * Pacientul accesează secțiunea de programare a consultațiilor.
3. **Selectare medic**:
   * Pacientul selectează un medic disponibil din lista celor care acceptă consultații online.
4. **Alegerea unui interval orar**:
   * Pacientul alege un interval orar disponibil pentru consultație.
5. **Confirmarea programării**:
   * După alegerea intervalului orar, pacientul confirmă programarea.
6. **Notificare pacient și medic**:
   * Sistemul trimite notificări către pacient și medic pentru confirmarea programării.
7. **Evenimente, acțiuni și stimuli:**

* **Eveniment**: Pacientul programează consultația.
* **Acțiune**: Sistemul confirmă programarea și notifică medicul.
* **Stimuli**: Alegerea intervalului orar și confirmarea.

1. **Interacțiuni:**

* **Utilizator (pacient)**: Selectează medicul și programul.
* **Sistem (platformă)**: Confirmă programarea și trimite notificările.

1. **Scenariul de utilizare: Vizualizarea recomandărilor de tratament de către medic**
2. **Descriere:**

Medicul dorește să vizualizeze recomandările de tratament generate de AI pentru un pacient și să le analizeze.

1. **Pași:**
2. **Autentificare medic**:
   * Medicul se autentifică în sistem cu datele sale de utilizator.
3. **Accesare istoricul pacientului**:
   * Medicul caută pacientul și accesează istoricul acestuia.
4. **Vizualizare recomandări tratament AI**:
   * Medicul vizualizează recomandările generate de AI pe baza simptomelor trimise de pacient.
5. **Aprobarea sau modificarea recomandărilor**:
   * Medicul poate aproba recomandările AI sau le poate modifica pe baza evaluării proprii.
6. **Confirmare tratament**:
   * După ce medicul aprobă sau modifică tratamentele, acestea sunt actualizate în sistem și pacientul va fi notificat.
7. **Evenimente, acțiuni și stimuli:**

* **Eveniment**: Medicul vizualizează recomandările AI.
* **Acțiune**: Medicul poate aproba sau modifica tratamentele.
* **Stimuli**: Actualizarea datelor în sistem.

1. **Interacțiuni:**

* **Utilizator (medic)**: Vizualizează și modifică recomandările de tratament.
* **Sistem (platformă)**: Afișează recomandările și actualizează statutul acestora.

## Proiectare de detaliu

Instrucțiuni: Furnizați informațiile necesare echipei de dezvoltare a sistemului pentru a construi și integra efectiv componentele hardware, pentru a programa și integra componentele software, și pentru a interconecta segmentele hardware și software într-un produs funcțional.

**Servere pentru găzduirea platformei**:

* Utilizăm servere dedicate pentru găzduirea platformei web și bazei de date (MongoDB). Aceste servere trebuie să fie echipate cu procesor performant (de preferat Intel Xeon sau AMD EPYC), cel puțin 32GB RAM și 1TB SSD pentru viteza de citire/scriere rapidă a datelor.
* Serverele vor fi configurate cu un sistem de operare Linux (de exemplu, Ubuntu 20.04 LTS) pentru stabilitate și suportul robust al mediilor de dezvoltare web.

1. **Frontend**

* **Tehnologii**:
  + **HTML5, CSS3 și JavaScript** pentru structura și design-ul paginilor web.
  + **React** sau **Vue.js** pentru dezvoltarea interfeței utilizatorului (UI), asigurând o interactivitate fluidă și performantă.
  + **Bootstrap** pentru a crea un design responsiv care se adaptează diferitelor dimensiuni de ecran.

1. **1.2.2 Backend**

* **Tehnologii**:
  + **Node.js** și **Express** pentru construirea unui server backend eficient.
  + **MongoDB** pentru gestionarea bazei de date, utilizând modelul document (JSON-like) pentru stocarea datelor pacientului, istoricul medical și recomandările AI.
  + **JWT** (JSON Web Tokens) pentru autentificare și autorizare a utilizatorilor (pacienți, medici, administratori).
  + **Socket.io** pentru notificări în timp real între pacienți și medici (de exemplu, pentru programările la consultații sau actualizările tratamentului).

1. **1.2.3 Integrarea AI**

* **Tehnologii**:
  + **Python** pentru dezvoltarea modelului de AI care va analiza simptomele pacienților. Utilizarea unor biblioteci de machine learning precum **TensorFlow** sau **PyTorch** pentru crearea unui model de clasificare a simptomelor și recomandări de tratament.
  + API-uri RESTful pentru comunicarea dintre backend-ul platformei și modelul AI.

1. **Flux de date**

* **Datele pacientului** (de exemplu, simptomele, istoricul medical) vor fi colectate prin frontend (formular de completat) și vor fi trimise la backend pentru stocare în MongoDB.
* **Datele de la AI** vor fi analizate pe serverele backend și rezultatele vor fi returnate în aplicația web pentru a fi vizualizate de către pacient sau medic.
* **Comunicarea între componentele hardware și software**:
  + Datele vor fi stocate și gestionate pe serverele din infrastructura cloud, iar toate interacțiunile între componentele software (frontend, backend, AI) vor fi gestionate prin intermediul API-urilor și al fluxurilor de date.

### Proiectare hardware de detaliu

Instrucțiuni: Furnizați suficiente informații detaliate despre fiecare dintre componentele hardware individuale pentru a construi și/sau achiziționa corect toate componentele hardware pentru sistem (sau pentru a integra articole COTS). Dacă există multe componente sau dacă documentația componentelor este extensivă, plasați-o într-o anexă. Adăugați diagrame și informații suplimentare, dacă este necesar, pentru a descrie fiecare componentă și funcțiile sale. Ar trebui să fie urmate practici standard din industrie pentru specificațiile componentelor.

Pentru componente externe, identificați furnizorul specific și denumirile și numerele de model corespunzătoare. Includeți următoarele informații în proiectele detaliate ale componentelor, după caz:

* Cerințe de intrare de alimentare pentru fiecare componentă
* Impedanțe și stări logice ale semnalului
* Specificații ale conectorilor (serial/paralel, 11 pini, mascul/femelă, etc.)
* Specificații ale spațiului de memorie și/sau stocare
* Cerințe ale procesorului (viteză și funcționalitate)
* Reprezentare grafică care prezintă numărul de articole hardware (de ex., servere, dispozitive I/O, monitoare, imprimante etc.), și poziționarea relativă a componentelor între ele

Acestea pot fi introduse ca anexe ale acestui document.

1. **Servere pentru Găzduirea Platformei Web și Bazei de Date**

* **Specificații**:
  + **Model**: Dell PowerEdge R740
  + **Procesor**: Intel Xeon Gold 6240R (20 nuclee, 40 fire de execuție), cu viteză de 3,4 GHz
  + **Memorie**: 64GB DDR4 (extensibilă până la 512GB)
  + **Stocare**: 2TB SSD NVMe, configurate în RAID 1 pentru redundanță
  + **Conectivitate rețea**: 2 porturi Ethernet 10GbE
  + **Sisteme de răcire**: Ventilatoare de mare performanță
  + **Sistem de operare**: Ubuntu Server 20.04 LTS
  + **Securitate**: TPM 2.0 pentru criptarea hardware a datelor
* **Cerințe de alimentare**:
  + **Tensiune**: 100-240V AC
  + **Putere**: 550W cu redundanță
* **Furnizor**: Dell Technologies
* **Conectori**: RJ-45 pentru rețea, 12V DC pentru alimentare internă

1. **Echipamente de Stocare (Sistem NAS)**

* **Model**: Synology DS1819+ NAS
* **Capacitate**: 8 bay-uri, fiecare cu 8TB HDD, total de 64TB pentru stocare de date
* **Conectivitate**: 2 porturi 10GbE pentru transfer rapid de date
* **Tip de stocare**: RAID 5 pentru redundanță și performanță
* **Backup**: Suportă backup-uri automate către cloud (AWS, Azure)
* **Cerințe de alimentare**:
  + **Tensiune**: 100-240V AC
  + **Putere**: 120W
* **Furnizor**: Synology
* **Conectori**: RJ-45 pentru rețea, 12V DC pentru alimentare

1. **Switch de Rețea**

* **Model**: Cisco Catalyst 9300
* **Porturi**: 48 porturi Gigabit Ethernet + 4 porturi 10GbE SFP+
* **Specificații**:
  + **Viteză**: Gigabit Ethernet (1GbE) pentru porturile standard, 10GbE pentru porturile SFP+
  + **Protocol**: Support pentru VLAN, QoS, Layer 3 routing
  + **Securitate**: 802.1X, ACL-uri pentru protecție
  + **PoE**: 480W pentru alimentare dispozitive PoE (ex. telefoane IP, camere de supraveghere)
* **Cerințe de alimentare**:
  + **Tensiune**: 100-240V AC
  + **Putere**: 400W
* **Furnizor**: Cisco Systems
* **Conectori**: RJ-45 pentru Ethernet, SFP+ pentru conexiuni de mare viteză

1. **Router de Internet**

* **Model**: Ubiquiti EdgeRouter 4
* **Specificații**:
  + **Porturi**: 4 porturi Gigabit Ethernet
  + **Viteză**: 1GbE pe toate porturile
  + **Protocol**: IPsec VPN, suport pentru multiple subneturi, NAT
  + **Securitate**: Firewall integrat cu protecție DDoS
* **Cerințe de alimentare**:
  + **Tensiune**: 100-240V AC
  + **Putere**: 25W
* **Furnizor**: Ubiquiti Networks
* **Conectori**: RJ-45 pentru Ethernet

### Proiectare software de detaliu

Instrucțiuni: Furnizați o descriere detaliată pentru fiecare serviciu software din sistem. O mare parte din informațiile care apar în această secțiune ar trebui să fie cuprinse în antete/prologuri și secțiunile de comentarii ale codului sursă pentru fiecare componentă, subsistem, modul și subrutină. În cazul în care este așa, această secțiune poate consta în mare parte din referințe sau extrase din diagramele anotate și codul sursă. Orice diagrame sau extrase din codul sursă referite ar trebui furnizate la orice revizii de proiectare.

* Identificator serviciu - Identificatorul unic și/sau numele serviciului software
* Clasificare - Tipul de serviciu (de exemplu, aplicație, serviciu de date, etc.)
* Definiție - Scopul specific și semnificația semantică a serviciului
* Cerințe - Cerințele specifice funcționale sau nonfuncționale pe care serviciul le satisface
* Structuri de date interne - Structurile de date interne pentru serviciu
* Constraingeri - Orice relevante, presupuneri, limite sau constrângeri pentru serviciu (aceasta ar trebui să includă constrângerile de timp, de stocare sau de stare a serviciului și ar putea include reguli pentru interacțiunea cu serviciul (cuprinzând precondiții, postcondiții, invariante, alte constrângeri privind valorile de intrare sau ieșire și valorile locale sau globale, formatele datelor și accesul la date, sincronizarea, excepțiile, etc.))
* Compoziție - O descriere a utilizării și semnificației subserviciilor care fac parte din serviciu
* Utilizatori/Interacțiuni - O descriere a colaborării serviciului cu alte servicii (ce alte servicii utilizează această entitate? ce alte servicii utilizează această entitate (inclusiv orice efecte secundare pe care le-ar putea avea acest serviciu asupra altor părți ale sistemului)? acest lucru include metoda de interacțiune, precum și interacțiunea în sine. Designurile orientate pe obiecte ar trebui să includă o descriere a oricăror sub-clase, super-clase și meta-clase cunoscute sau anticipate)
* Procesare - O descriere precisă a modului în care serviciul își îndeplinește responsabilitățile (aceasta ar trebui să cuprindă o descriere a oricăror algoritmi utilizați; modificări ale stării; complexitate relevantă în timp sau spațiu; concurență; metode de creare, inițializare și curățare; și gestionarea condițiilor excepționale)
* Interfețe/Exporturi - Setul de servicii (resurse, tipuri de date, constante, subrutine și excepții) pe care serviciul le oferă (definiția sau declarația precisă a fiecărui astfel de element ar trebui să fie prezentă, împreună cu comentarii sau adnotări care descriu semnificațiile valorilor, parametrilor etc.; pentru fiecare element de serviciu descris, includeți sau furnizați o referință în discuția sa la o descriere a atributelor sale importante ale serviciului software (Identificator Componentă, Clasificare, Limbaj, Estimare SLOC, Definiție, Responsabilități, Cerințe, Structuri de Date Interne, Constraingeri, Compoziție, Utilizări/Interacțiuni, Resurse, Procesare și Interfețe/Exporturi))
* Raportare Design și Integrare - Dacă este inclus, furnizați detalii despre traficul și volumele de date.

1. Serviciul de Autentificare și Gestionare a Utilizatorilor

* Identificator serviciu: AuthService
* Clasificare: Serviciu de autentificare
* Definiție: Acest serviciu gestionează autentificarea utilizatorilor în platformă, crearea și actualizarea conturilor de utilizator, și verificarea accesului. Se asigură că doar utilizatorii autorizați pot accesa funcționalitățile sensibile ale platformei.
* Cerințe:
  + Autentificare bazată pe token (JWT).
  + Validare email la înregistrare.
  + Politici de parolă (minim 8 caractere, 1 caracter special, 1 literă mare).
  + Permite recuperarea parolelor.

1. Serviciul de Analiză a Simptomelor (AI)

* Identificator serviciu: SymptomAnalyzerService
* Clasificare: Serviciu de date
* Definiție: Serviciul utilizează un model de inteligență artificială pentru a analiza simptomele trimise de utilizatori și pentru a sugera posibile afecțiuni. Modelele de AI sunt antrenate pe baza unui set extins de date medicale.
* Cerințe:
  + Analiza simptomelor să fie realizată în maxim 5 secunde.
  + Recomandarea unui diagnostic preliminar pe baza simptomelor.

1. Serviciul de Gestionare a Programărilor

* Identificator serviciu: AppointmentService
* Clasificare: Serviciu de management
* Definiție: Permite pacienților să programeze consultații cu doctorii, să vizualizeze programul acestora și să primească notificări.
* Cerințe:
  + Posibilitatea de a adăuga, șterge și modifica programări.
  + Sincronizare cu calendarul doctorului.

### Proiectare detaliată de securitate

Instrucțiuni: Furnizați informații detaliate pentru fiecare dintre componentele individuale de securitate hardware. Specificați elementele de mai jos, după cum este necesar.

• Autentificare

• Autorizare

• Jurnalizare(arhivare) și auditare

• Criptare

• Utilizarea porturilor de rețea

• Detectare și prevenire a intruziunilor.

Autentificarea utilizatorilor va fi gestionată folosind autentificarea bazată pe token (JWT - JSON Web Token). Utilizatorii trebuie să se autentifice prin furnizarea unui email și a unei parole.

**Parole**: Parolele utilizatorilor vor fi stocate folosind algoritmi de hashare criptografică (de exemplu, **bcrypt** sau **Argon2**) pentru a preveni expunerea acestora chiar și în cazul unui atac asupra bazei de date.

**Token JWT**: După autentificare, un token JWT va fi emis pentru a permite accesul utilizatorilor la resursele protejate. Tokenul va conține informații de sesiune și va fi semnat cu o cheie secretă.

**Autentificare cu două factori (2FA)**: Pentru utilizatorii cu privilegii înalte (de exemplu, administratori sau medici), se va implementa un sistem de 2FA, utilizând aplicații precum Google Authenticator sau prin SMS.

Autorizarea:

**Roluri**: Fiecare utilizator va avea un rol asociat, care va determina ce permisiuni are. De exemplu, roluri precum Pacient, Doctor, Administrator.

* **Permisiuni**: Fiecare rol va avea un set de permisiuni care definesc ce operațiuni pot fi efectuate. De exemplu, doar utilizatorii cu rolul Doctor vor putea vizualiza diagnosticul pacienților.
* **Controale de acces pe bază de roluri (RBAC)**: Accesul la fiecare funcționalitate a aplicației va fi controlat printr-un sistem de autorizare bazat pe roluri.
* **Criterii de autorizare**: Cerințele de autorizare vor include verificarea permisiunilor utilizatorilor pentru fiecare operațiune solicitată.

Criptarea:

C**riptare în tranzit**: Toate comunicațiile între utilizatori și servere vor fi criptate folosind **TLS (Transport Layer Security)** pentru a proteja datele împotriva interceptării.

**Criptare în stocare**: Datele sensibile (de exemplu, informațiile de sănătate ale pacienților) vor fi criptate folosind algoritmi de criptare simetrică.

**Criptare pentru parole**: Parolele utilizatorilor vor fi criptate folosind algoritmi de hashare unidirecționali.

Vor fi deschise doar porturile necesare pentru funcționarea sistemului, de exemplu, portul 443 pentru HTTPS.

Sistemele vor fi protejate de un firewall care va bloca traficul din surse nesigure.

### Proiectare de detaliu pentru performanța sistemului

Instrucțiuni: Furnizați o reprezentare grafică cu informații detaliate pentru fiecare dintre componentele hardware individuale de performanță și fiabilitate, pentru a include elementele de mai jos:

• Cerințe/estimări de capacitate și volum

• Așteptări de performanță

• Cerințe de disponibilitate

• Proiectare de performanță pentru a îndeplini cerințele de capacitate

• Proiectare de fiabilitate pentru a îndeplini cerințele de disponibilitate

• Proiectare de backup, recuperare și arhivare

Identificați punctele unice de eșec și, dacă este relevant, descrieți proiectarea de disponibilitate ridicată (de exemplu, clustering).

Pana la 1000 de utilizatori simultan incluzand pacienti,medici si utilizatori.

100 integrari api simultane pe secunda in perioadele de varf.

Timp de raaspuns mediu de 2 secunde pentru interogarile API

Interfata cu utilizatorul va avea un rasapuns de 1-2 secunde pentru incarcarea paginilor si vizualizarea datelor.

Serverele aplicației vor fi configurate în grupuri de tip **clustering** pentru a asigura că, în cazul unei defecțiuni hardware, alt server poate prelua instantaneu sarcina. Acest lucru va fi valabil și pentru serverele de baze de date.

Toate datele vor fi replicate pe mai multe servere pentru a asigura redundanța. În caz de eșec al unui server sau al unui centru de date, sistemul va putea recupera datele din replica activă.

### Proiectare detaliată a comunicațiilor interne (între componente)

Instrucțiuni: Dacă sistemul include mai mult de o componentă, este posibil să existe o cerință pentru comunicații interne pentru a schimba informații, a furniza comenzi sau a susține funcțiile de intrare/ieșire. Furnizați suficiente informații detaliate despre proiectarea comunicațiilor pentru a construi și/sau a achiziționa corect componentele de comunicații pentru sistem. Includeți următoarele informații în proiectările detaliate ale componentelor, după caz:

* Numărul de servere și clienți care urmează să fie incluși pe fiecare rețea de zonă
* Specificații pentru cerințele de sincronizare și control al busului
* Format(e) pentru datele schimbate între componente
* Reprezentare grafică a conectivității între componente, arătând direcția fluxului de date (dacă este aplicabil), și distanțele aproximative între componente (informațiile ar trebui să furnizeze suficient detaliu pentru a susține achiziționarea hardware-ului pentru finalizarea instalării într-o anumită locație)
* Topologia LAN
* Servere de aplicație: 3-5 servere dedicate pentru aplicația web, configurate în clustering pentru a asigura redundanța și scalabilitatea.
* Servere de baze de date: 2-3 servere configurate într-o arhitectură de replicare master-slave pentru asigurarea redundanței și performanței.
* Servere de stocare: 1-2 servere de stocare pentru fișierele mari (imagini medicale, documente etc.), configurate pentru scalabilitate.
* Clienți interni: Medicii, administratorii și angajații care vor accesa sistemul prin stații de lucru sau dispozitive mobile.
* Clienți externi: Până la 1000 de pacienți care se conectează periodic pentru a-și vizualiza rezultatele medicale, a interacționa cu AI-ul sau a comunica cu medicii.

Rețeaua va fi configurată în topologie **"star"** cu un **switch central** care conectează toate serverele de aplicație, serverele de baze de date și serverele de fișiere.

## Controale pentru verificarea integrității sistemului

Instrucțiuni: Furnizați specificații de proiectare pentru următoarele nivele minime de control și orice controale suplimentare adecvate sau necesare:

* Securitate internă pentru a restricționa accesul la datele critice doar pentru acele tipuri de acces necesare de către utilizatori/operatori
* Proceduri de audit pentru a îndeplini cerințele de control, raportare și perioade de reținere pentru rapoartele operaționale și de management
* Piste de auditare a aplicațiilor pentru a audita dinamic accesul la recuperare la datele critice designate
* Tabele standard care urmează să fie utilizate sau solicitate pentru validarea câmpurilor de date
* Procese de verificare pentru adăugarea, ștergerea sau actualizarea datelor critice
* Capacitatea de a identifica toate informațiile de auditare prin identificarea utilizatorului, identificarea terminalului de rețea, dată, oră și datele accesate sau modificate.

Sistemul va implementa un sistem de control al accesului bazat pe roluri pentru a asigura că doar utilizatorii autorizați pot accesa datele critice. Utilizatorii vor fi clasificați în funcție de rolul lor: pacient, medic, administrator, și vor avea acces doar la informațiile relevante pentru fiecare tip de rol.

Conturi de utilizator: Fiecare utilizator va avea un cont unic, asociat cu un set de permisiuni specific pentru funcțiile sale. Accesul va fi restricționat la datele relevante în funcție de aceste permisiuni.

Criptarea datelor sensibile: Toate datele sensibile (de exemplu, istoricul medical al pacientului) vor fi criptate atât la stocare (pe serverele de baze de date) cât și în tranzit (prin TLS).

Sistemul va monitoriza continuu activitățile pentru a identifica orice comportament neobișnuit care poate sugera o breșă de securitate sau un acces neautorizat.

Alarme automate vor fi generate în caz de acces neautorizat sau de modificări suspecte ale datelor critice.

Toate acțiunile de acces la date critice (de exemplu, accesul la istoricul medical al pacientului, tratamentele recomandate) vor fi înregistrate într-un fișier de jurnal de audit.

Anexa A: Gestiunea modificărilor documentului

Instrucțiuni: Furnizați informații despre modul în care dezvoltarea și distribuția documentului va fi controlată și urmărită. Utilizați tabelul de mai jos pentru a furniza numărul de versiune, data versiunii, autorul/deținătorul versiunii și o scurtă descriere a motivului pentru crearea versiunii revizuite.

Tabel 1 – Înregistrarea modificărilor asupreaa documentului curent

| versiune | Data | Autorul/Deținătorul | Descriere |
| --- | --- | --- | --- |
| <X.X> | <ZZ/LL/AAAA> | <nume autor> | <Descrierea modificării> |
| <X.X> | <ZZ/LL/AAAA> | <nume autor> | <Descrierea modificării> |
| <X.X> | <ZZ/LL/AAAA> | <nume autor> | <Descrierea modificării> |

Anexa B: Acronime

*Instrucțiuni: Furnizați o listă de acronime și traduceri literale asociate utilizate în cadrul documentului. Enumerați acronimele în ordine alfabetică folosind un format tabular, așa cum este ilustrat mai jos.*

Tabel 2 - Acronime

| Acronim | Forma completă |
| --- | --- |
| <Acronim> | <Forma completă> |
| <Acronim> | <Forma completă> |
| <Acronim> | <Forma completă> |

Anexa C Documente la care se face referire

*Instrucțiuni: Sintetizați relația acestui document cu alte documente relevante. Furnizați informații de identificare pentru toate documentele folosite pentru a ajunge la și/sau referite în acest document (de exemplu, documente conexe și/sau asociate, documente prealabile, documentație tehnică relevantă, etc.).*

Tabel 3 – Documente la care se facce referire

| Nume document | Locație sau URL | Dată emitere document |
| --- | --- | --- |
| < Nume document > | <Locație sau URL> | <ZZ/LL/AAAA> |
| < Nume document > | <Locație sau URL> | <ZZ/LL/AAAA> |
| < Nume document > | <Locație sau URL> | <ZZ/LL/AAAA> |