Ministerul Educaţiei, Culturii și Cercetării
Universitatea Tehnică a Moldovei

Faculatea Calculatoare, informatică și microelectronică
Departamentul Ingineria Software și Automatică

RAPORT

Lucrare de Laborator nr.6
Disciplina: Proiectarea sistemelor informaționale
Tema : Dezvoltarea unei aplicații pentru monitorizarea stării animalelor de companie

A efectuat:

Valciuc Andrei

A verificat:

Bodoga Cristina

Chişinău 2023

INTRODUCERE

În cadrul procesului de dezvoltare a unui sistem complex, modelarea și descrierea structurii statice reprezintă o etapă fundamentală ce contribuie semnificativ la înțelegerea arhitecturii și relațiilor dintre componentele sistemului. Acest proces impune utilizarea unor instrumente specifice, iar pentru atingerea acestui obiectiv, se adoptă o abordare detaliată și sistematică.

Prima componentă a acestei etape presupune utilizarea diagramei de clasă pentru a oferi o reprezentare coerentă a entităților și a relațiilor dintre acestea în cadrul sistemului. Astfel, fiecare clasă este definită cu atributele și metodele sale, evidențiindu-se ierarhiile și asocierile între obiectele ce compun sistemul. Această abordare facilitează o înțelegere clară a structurii logice a sistemului, contribuind la o implementare eficientă și modulară. Un alt obiectiv crucial constă în identificarea și reprezentarea relațiilor de dependență dintre diversele componente ale sistemului. Acest proces se realizează cu ajutorul diagramei de componente, unde fiecare element este detaliat în funcție de rolul său în sistem. Această perspectivă oferă o imagine globală asupra interdependențelor între module și permite o gestionare mai eficientă a complexității sistemului. Ultima parte a acestui proces se concentrează pe modelarea echipamentelor din mediul de implementare prin intermediul diagramei de deployment. Această diagramă ilustrează modul în care componentele software și hardware interacționează în cadrul mediului de execuție. Identificarea resurselor hardware, conexiunile de rețea și distribuția logică a aplicațiilor reprezintă aspecte cheie pentru asigurarea unei implementări robuste și eficiente. Prin intermediul diagramei de clasă, se evidențiază entitățile-cheie ale sistemului și relațiile între acestea. Aceasta oferă un cadru comprehensiv pentru proiectarea obiectelor și a structurii logice a sistemului, înlesnind dezvoltatorii să înțeleagă interacțiunile și să identifice punctele-cheie ale arhitecturii. Atributele și metodele claselor sunt detaliate, iar asocierile dintre obiecte sunt reprezentate vizual, contribuind la definirea clară a comportamentului sistemului. Diagramele de componente aduc în prim-plan arhitectura fizică a sistemului, evidențiind interdependențele între module și clarificând distribuția acestora în cadrul aplicației. Această perspectivă permite echipei de dezvoltare să gestioneze mai eficient complexitatea proiectului, să optimizeze interacțiunile dintre componente și să faciliteze reutilizarea modulelor în proiecte viitoare.

În final, diagramele de deployment oferă o privire asupra modului în care aplicația va fi implementată în mediul său de execuție. Detaliind echipamentele hardware și conexiunile de rețea, aceste diagrame ajută la anticiparea și evitarea potențialelor probleme de implementare, asigurând o execuție eficientă și fiabilă a sistemului.

1. DIAGRAME DE CLASĂ

Diagramele de clasă sunt instrumente esențiale în lumea dezvoltării software, utilizate pentru a modela și descrie structura statică a sistemelor orientate pe obiect. Aceste diagrame oferă o reprezentare vizuală a entităților-cheie ale sistemului, inclusiv claselor, atributelor, metodelor și relațiilor dintre acestea.

Prin intermediul diagramelelor de clasă, proiectanții și dezvoltatorii pot defini cu claritate tipurile de obiecte din sistem și interacțiunile dintre acestea. Clasele reprezintă tipurile de obiecte și includ atribute (caracteristici) și metode (comportamente). Relațiile între clase, precum asocieri, agregări sau moștenire, evidențiază modul în care obiectele interacționează și cooperează în cadrul sistemului. Aceste diagrame sunt utilizate în diverse contexte ale dezvoltării software. În stadiul de proiectare, diagramele de clasă ajută la definirea unei structuri solide și coerente pentru sistem. Ele sunt, de asemenea, esențiale în comunicarea între membrii echipei de dezvoltare și în interacțiunea cu clienții sau stakeholderii. Prin intermediul diagramelelor de clasă, echipele pot transmite concepte și idei complexe într-un mod accesibil și ușor de înțeles.

Pe lângă rolul lor în proiectare și comunicare, diagramele de clasă au și o funcție practică în procesul de dezvoltare a software-ului. Ele pot fi utilizate pentru generarea automată a codului sursă în anumite medii de dezvoltare și facilitează analiza și refactorizarea codului existent. De asemenea, reprezintă o sursă de documentare utilă, oferind o viziune de ansamblu asupra structurii sistemului pentru dezvoltatori și echipele de mentenanță. În figura 1.1 putem observa diagrama diagrama de clasă pentru programarea la veterinar în cadrul sistemului de monitorizare:

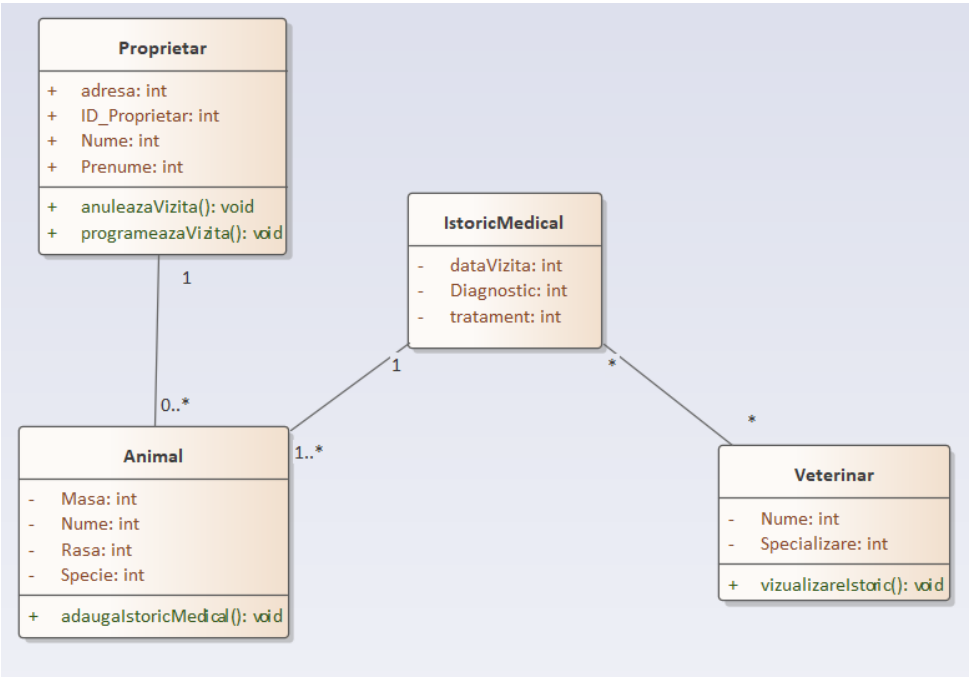

Figura 1.1 – Diagrama de clasă pentru înregistrarea la veterinar

Diagrama de clasă pentru programare la veterinar evidențiază relațiile complexe dintre diferitele clase implicate în această funcționalitate esențială în domeniul veterinar.

Clasa "Programare" reprezintă o programare la veterinar și cuprinde proprietăți precum data și ora programării, proprietarul și animalul de companie implicate, tipul programării, precum și informații legate de diagnostic și tratament. Această clasă este strâns legată de clasa "Proprietar", care descrie detaliile personale ale persoanei care deține animalul de companie, și de clasa "Animal", care detaliază aspecte precum numele, rasa și specia animalului.

Clasa "Veterinar" reprezintă medicul veterinar responsabil pentru efectuarea programării, evidențiind informații despre nume, prenume și specializare. Relațiile dintre clase sunt esențiale pentru a înțelege fluxul de informații în cadrul sistemului: o programare are un singur proprietar, dar un proprietar poate avea mai multe programări; o programare este asociată cu un singur animal, dar un animal poate avea mai multe programări. De asemenea, o programare poate implica un tip specific de programare, un **singur diagnostic și un singur tratament.**

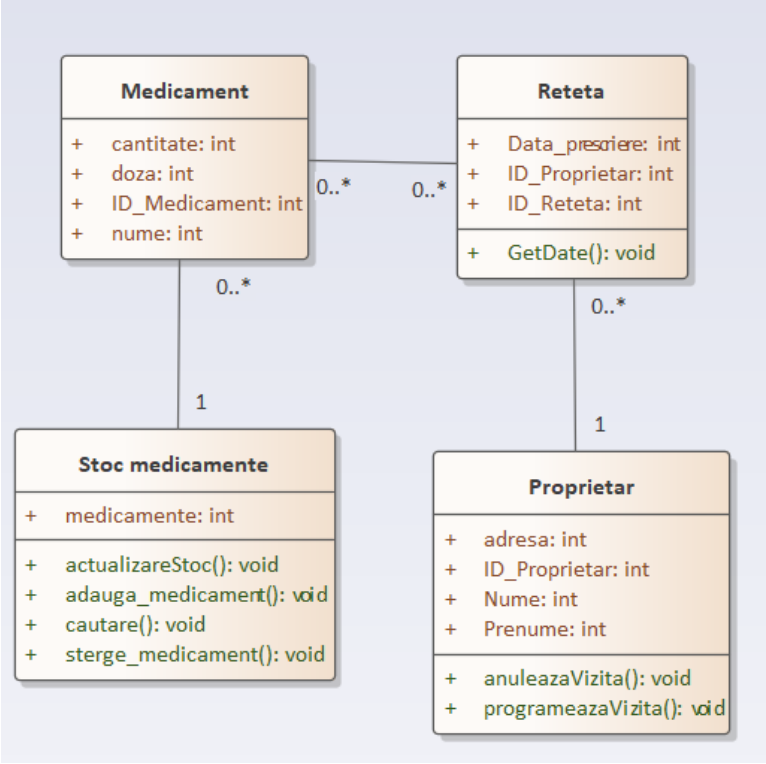

Figura 1.2 – Diagrama de clasă pentru gestinonarea medicamentelor

Diagrama de clasă pentru gestionarea medicamentelor oferă o imagine detaliată a relațiilor intricate dintre clasele implicate în această funcționalitate esențială. Clasa "Medicament" reprezintă un element central, reflectând medicamentele utilizate pentru tratarea animalelor de companie. Proprietățile sale, precum numele medicamentului, doza recomandată, modul de administrare și posibilele efecte

secundare, sunt esențiale în definirea caracteristicilor acestor medicamente. În continuare, clasa "Proprietar" descrie detaliile personale ale persoanei care deține animalul de companie, inclusiv numele, prenumele, adresă și numărul de telefon. La rândul său, clasa "Animal" aduce în discuție caracteristicile specifice ale animalului de companie, cum ar fi numele, rasa și specia. Clasa "Reteta" devine pivotul pentru gestionarea prescripțiilor medicale. Aceasta include proprietăți precum un identificator unic, data emiterii rețetei, medicamentul prescris, doza, modul de administrare și posibilele efecte secundare.

Relațiile între clase adaugă o dimensiune complexă întregului sistem. De exemplu, o rețetă este asociată cu un singur medicament, dar același medicament poate fi prescris mai multor animale de companie. Similar, o rețetă este legată de un singur proprietar, dar același proprietar poate avea mai multe animale de companie pentru care primesc prescripții.

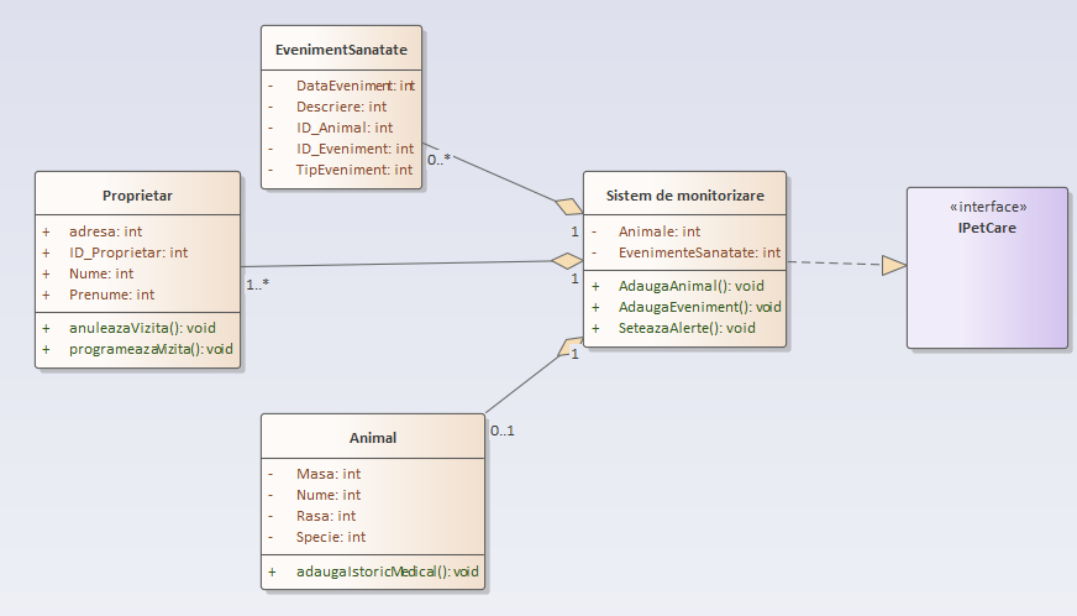

Figura 1.3 – Diagrama de clasă pentru gestionare evenimente de sănătate

Diagrama de clasă pentru gestionarea evenimentelor de sănătate oferă o perspectivă detaliată asupra relațiilor complexe dintre clasele implicate în această funcționalitate crucială.

Clasa "EvenimentSanatate" constituie elementul central al acestei diagrame, reprezentând un eveniment de sănătate ce are loc în cadrul vieții unui animal de companie. Proprietățile acestei clase includ informații precum data evenimentului, o descriere detaliată a acestuia, precum și identificatorii unici pentru animalul de companie și tipul de eveniment asociat.

În continuare, clasa "Proprietar" descrie detaliile personale ale persoanei responsabile pentru animalul de companie, precum numele, prenumele, adresă și numărul de telefon. Clasa "Animal" aduce în discuție caracteristicile specifice ale animalului de companie, cum ar fi numele, rasa și specia

2. DIAGRAME DE COMPONENTE

Diagramele de componente reprezintă instrumente esențiale în domeniul dezvoltării software, oferind o perspectivă vizuală asupra structurii și interacțiunilor între componente în cadrul unui sistem software. Aceste diagrame sunt utilizate pentru a ilustra modul în care diferitele module sau părți ale unei aplicații interacționează și cooperează pentru a atinge obiectivele sistemului.

Fiecare componentă este reprezentată grafic, evidențiind interfețele sale, dependențele față de alte componente și rolurile pe care le joacă în funcționarea generală a sistemului. Diagramele de componente facilitează înțelegerea arhitecturii software, facilitând comunicarea între membrii echipei de dezvoltare și stabilirea unor standarde clare pentru integrarea și dezvoltarea ulterioară.

Un alt aspect important al diagramei de componente este evidențierea relațiilor și dependențelor dintre componentele individuale. Aceste relații includ, de obicei, asocierea, agregarea și compoziția, oferind o viziune holistică asupra structurii sistemului.

Diagramele de componente nu sunt doar instrumente de proiectare, ci și un mijloc eficient de comunicare cu stakeholderii și clienții. Ele permit o prezentare clară a arhitecturii software, ajutând la evitarea ambiguităților și la asigurarea unei înțelegeri comune între toți cei implicați în procesul de dezvoltare. În figura 2.1 poate fi observată diagrama de componente pentru realizarea unei tranzacții:

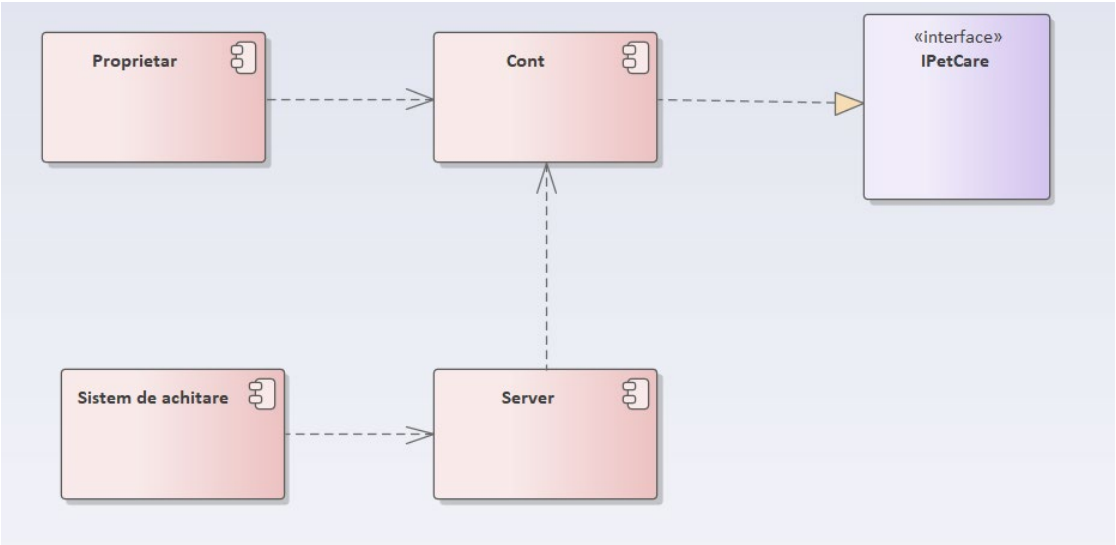

Figura 2.1 – Diagrama de componente pentru realizarea unei tranzacții

Diagrama de componente relevată evidențiază un sistem dedicat realizării tranzacțiilor. Acest sistem complex se compune din mai multe componente interconectate, fiecare având un rol esențial în asigurarea funcționalității corespunzătoare.

În centrul acestei reprezentări, "Proprietarul" este identificat ca utilizatorul care inițiază și efectuează tranzacția. Interacțiunea cu sistemul se realizează prin intermediul "Contului" său, care reprezintă platforma prin care "Proprietarul" comunică cu întregul sistem.

"Sistemul de achitare" ocupă un rol crucial în procesul de realizare a tranzacției, asigurând acceptarea și procesarea plăților. "Proprietarul" utilizează "Contul" său pentru a comunica cu "Sistemul de achitare" și pentru a introduce informațiile necesare pentru plată.

De asemenea, "Sistemul de achitare" are o relație directă cu "Serverul" care găzduiește datele de tranzacție. Această conexiune este crucială pentru procesarea eficientă a plăților și pentru a asigura înregistrarea corectă a tranzacțiilor în sistem.

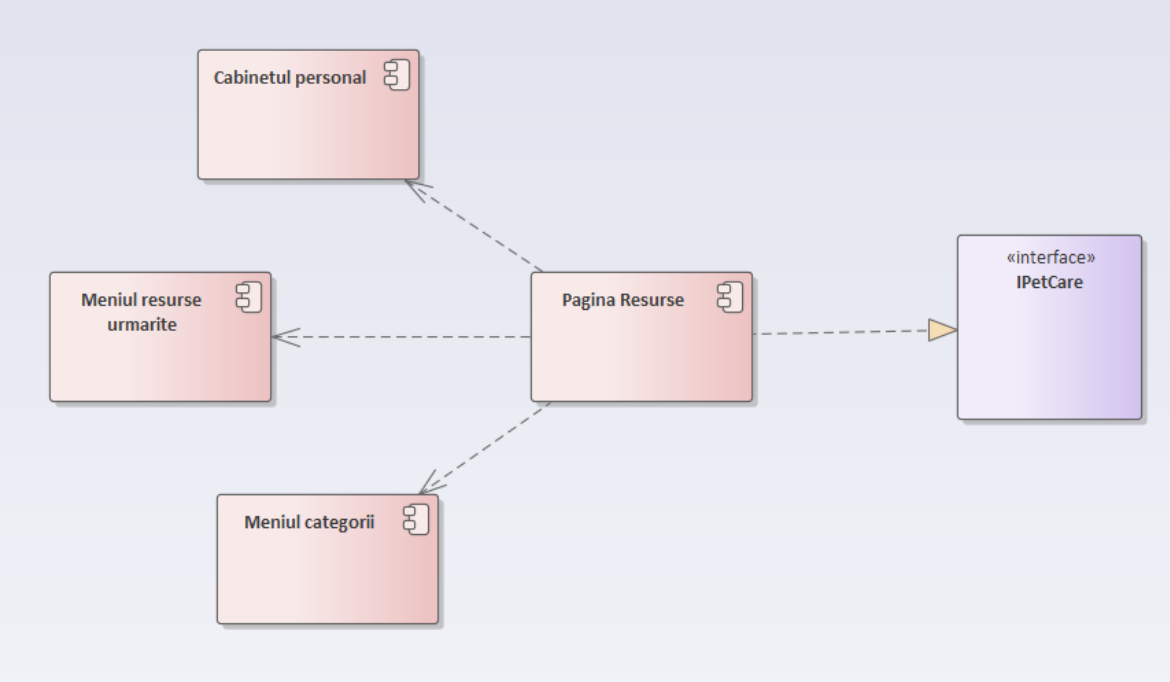

Figura 2.2 – Diagrama de componente pentru gestionarea resurselor

Diagrama de componente relevată evidențiază un sistem dedicat gestionării resurselor. Această structură complexă este compusă din mai multe componente esențiale care lucrează împreună pentru a asigura funcționalitatea corespunzătoare a sistemului.

În centrul acestei reprezentări, "Cabinetul personal" se manifestă ca o interfață grafică de utilizator (GUI) care oferă utilizatorilor posibilitatea de a vizualiza și gestiona resursele disponibile. Acesta servește drept punte între utilizator și întregul sistem, furnizând o modalitate intuitivă de interacțiune.

"IPetCare" se prezintă sub forma unei interfețe software, facilitând accesul la datele referitoare la resurse. Interacțiunea dintre "Cabinetul personal" și "IPetCare" se realizează prin intermediul acestei interfețe, în care "IPetCare" furnizează datele despre resurse către "Cabinetul personal".

Elemente precum "Meniul resurse", "Pagina resurse urmărite" și "Meniul categorii" sunt componente cheie ale GUI-ului, oferind utilizatorilor opțiuni clare pentru selecția și gestionarea resurselor. "Meniul resurse" permite utilizatorilor să aleagă resursele dorite, "Pagina resurse urmărite" afișează resursele marcate ca fiind urmărite, iar "Meniul categorii" permite selecția categoriilor de resurse.

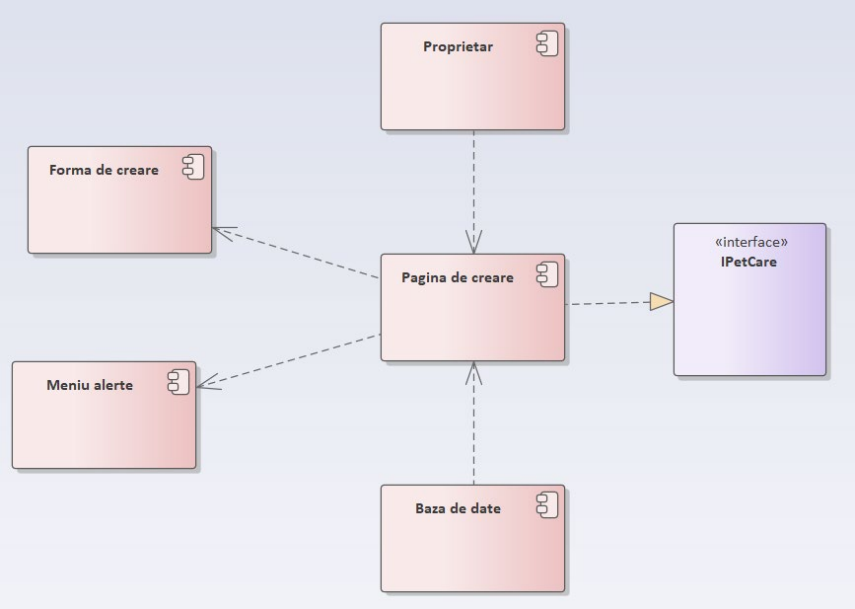

Figura 2.3 – Diagrama de componente pentru gestionarea evenimentelor

"Proprietarul" este un element central al sistemului, reprezentând utilizatorul care creează și gestionează evenimentele. Interacțiunea cu sistemul are loc prin intermediul "Formei de creare", o interfață grafică de utilizator concepută pentru a facilita procesul de creare a evenimentelor. Acesta este un instrument esențial, permitând "Proprietarului" să introducă detaliile necesare pentru evenimentul dorit. "Pagina de creare" este o secțiune specifică a GUI-ului care afișează informațiile relevante despre evenimentul în curs de creare. Această interfață îmbunătățește experiența "Proprietarului", permițându-i să vizualizeze și să ajusteze detaliile evenimentului în timp real. "Meniul alerte" constituie un alt element cheie al GUI-ului, oferind "Proprietarului" posibilitatea de a crea sau edita alerte asociate evenimentelor. Acesta adaugă un strat suplimentar de funcționalitate, permitând utilizatorului să stabilească notificări sau să facă modificări în legătură cu evenimentele existente. Pe lângă interacțiunea directă cu "Forma de creare", "Proprietarul" comunică și cu "Baza de date". Acest element esențial stochează datele despre evenimente, asigurându-se că informațiile sunt salvate și accesibile în mod corespunzător.

3. DIAGRAMA DE DEPLOYMENT

Diagramele de deployment oferă o perspectivă detaliată asupra modului în care diferitele componente software și hardware interacționează și sunt plasate într-un mediu operațional. Aceste diagrame servesc la ilustrarea distribuției fizice a sistemului, inclusiv a serverelor, a stațiilor de lucru, a rețelelor, precum și a altor dispozitive și elemente hardware implicate în funcționarea unei aplicații sau a unui sistem software.

Într-o astfel de diagramă, entitățile hardware precum serverele, calculatoarele sau dispozitivele de stocare sunt reprezentate grafic pentru a indica modul în care acestea comunică între ele. De asemenea, se evidențiază conexiunile de rețea, indicând modul în care datele sunt transferate între diferitele componente ale sistemului. Fiecare componentă este plasată într-un context fizic specific, evidențiind modul în care sunt distribuite pe diverse noduri sau locații geografice, dacă este cazul.

Diagramele de deployment sunt utile pentru a înțelege aspecte practice ale implementării unui sistem, inclusiv cum este configurată infrastructura hardware și cum este gestionată comunicarea între componente. Acestea pot acoperi diverse aspecte, de la configurarea rețelelor la modul în care serverele sunt distribuite pentru a asigura o performanță optimă și o redundanță adecvată.

În general, aceste diagrame reprezintă o unealtă valoroasă în proiectarea și documentarea sistemelor complexe, permițând echipei de dezvoltare și altor părți interesate să obțină o imagine clară asupra modului în care infrastructura fizică susține și optimizează funcționarea aplicației sau sistemului software. În figura de mai jos este reprezentată diagrama de deployment pentru sistemul de monitorizare a stării animalelor de companie:

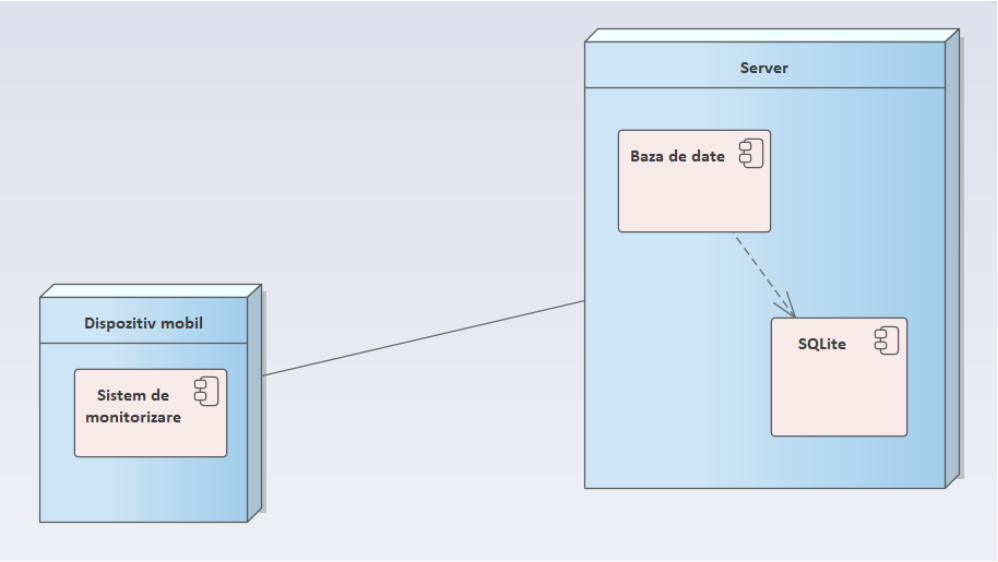

Figura 3.1 – Diagrama de deployment pentru sistemul de monitorizare a starii animalelor de companie

Diagrama de deployment relevă un sistem de monitorizare dedicat animalelor de companie, prezentând o arhitectură compusă din mai multe componente esențiale. Un element central al sistemului este dispozitivul mobil, reprezentat de un telefon sau o tabletă, purtat de animalul de companie. Aceste dispozitive sunt dotate cu senzori care colectează o gamă variată de date, inclusiv locația, temperatura corpului și nivelul de activitate al animalului.

Conexiunea dintre dispozitivul mobil și server este facilitată prin intermediul unei conexiuni Wi-Fi sau celulare. Serverul, la rândul său, îndeplinește două funcții cruciale: găzduirea bazei de date și a aplicației web asociate sistemului. Baza de date este destinată stocării datelor colectate de dispozitivele mobile, oferind un depozit centralizat și accesibil eficient pentru informațiile referitoare la starea animalului de companie.

Aplicația web servește ca interfață pentru proprietari, permițându-le să vizualizeze și să gestioneze datele acumulate de dispozitivele mobile. Această interfață intuitivă oferă proprietarilor posibilitatea de a monitoriza în timp real locația animalului, temperatura corpului și nivelul de activitate, contribuind la menținerea bunăstării acestuia.

Funcționalitatea sistemului se extinde prin posibilitatea ca proprietarii să stabilească alerte personalizate. Aceste alerte pot notifica proprietarii în situații specifice, cum ar fi părăsirea unei anumite zone de către animalul de companie sau atingerea unei temperaturi corporale prea ridicate. Astfel, sistemul oferă o soluție comprehensivă de monitorizare la distanță, aducând beneficii semnificative în ceea ce privește grija și securitatea animalelor de companie.

CONCLUZII

În urma parcurgerii obiectivelor stabilites-a realizat o abordare comprehensivă a aspectelor esențiale ale arhitecturii sistemului. Utilizând diagramele Class, s-a reușit o descriere detaliată a structurii statice a sistemului, evidențiind clasele componente și relațiile semnificative între acestea. Această metodă a oferit o perspectivă clară asupra entităților-cheie din cadrul sistemului, precum și asupra modului în care acestea interacționează. În etapa următoare, s-a făcut uz de diagramele Component pentru a identifica și reprezenta relațiile de dependență între componentele sistemului. Acest aspect a contribuit la conturarea conexiunilor esențiale dintre diferitele module și elemente componente, consolidând astfel înțelegerea modului în care acestea colaborează pentru a susține funcționalitățile globale ale sistemului.

Modelarea echipamentelor mediului de implementare a fost realizată cu ajutorul diagramei Deployment. Aceasta a oferit o imagine detaliată a configurării fizice a sistemului, evidențiind distribuția și interconectivitatea echipamentelor. Prin această abordare, s-a realizat o vizualizare clară a modului în care componentele software și hardware interacționează și cooperează în cadrul mediului de implementare.

Această abordare detaliată în modelarea și descrierea structurii statice a sistemului oferă un fundament esențial pentru echipele de dezvoltare și alte părți interesate implicate în proiect. Prin intermediul diagramei Class, s-a reușit să evidențiem ierarhiile, atributele și metodele cheie ale claselor, facilitând astfel o înțelegere clară a entităților și relațiilor lor în cadrul sistemului. Adițional, diagramele Component au contribuit semnificativ la relevarea dependențelor și interconexiunilor dintre modulele și componente. Această perspectivă detaliată asupra structurii statice a sistemului este esențială pentru optimizarea eficienței și modularității, facilitând astfel procesul de dezvoltare și mentenanță ulterioară. Prin intermediul diagramei Deployment, s-a oferit o viziune asupra modului în care software-ul și hardware-ul se împletesc în cadrul mediului de implementare. Această înțelegere a configurării fizice a sistemului este crucială pentru asigurarea unei implementări corespunzătoare și pentru anticiparea posibilelor provocări legate de infrastructură.

În concluzie, prin atingerea acestor obiective, echipa de dezvoltare dispune acum de resursele necesare pentru a avansa în etapele următoare ale proiectului. Modelarea și descrierea structurii statice a sistemului reprezintă un pas crucial către construirea unui sistem robust, adaptabil și eficient în îndeplinirea obiectivelor propuse.