Universitatea „Babeș-Bolyai” Cluj-Napoca

Facultatea De Științe Economice Și Gestiunea Afacerilor

Cluj Napoca

**Lucrare de licență**

Coordonator

Lect. Univ. dr. Cristian Bologa

Student

2016

Universitatea „Babeș-Bolyai” Cluj-Napoca

Facultatea de Științe Economice și Gestiunea Afacerilor

Cluj Napoca

**Lucrare de licență**

*Analiza și proiectarea unei aplicatii online pentru cabinet stomatologic*

Coordonator

Lect. Univ. dr. Cristian Bologa

Student

2016

CUPRINS

[PARTEA I – RAPORT DE ANALIZĂ 1](#_Toc453780183)

[INTRODUCERE 2](#_Toc453780184)

[1. IDENTIFICAREA ȘI DESCRIEREA PROBLEMEI 3](#_Toc453780185)

[1.1 MOTIVAȚIE 3](#_Toc453780186)

[1.2 CONTEXT 6](#_Toc453780187)

[2. CERINȚE DE SISTEM 10](#_Toc453780188)

[2.1 SURSE DE CERINȚE 10](#_Toc453780189)

[2.2 ELICITAȚIA CERINȚELOR 11](#_Toc453780190)

[2.3 FORMULAREA CERINȚELOR 26](#_Toc453780191)

[2.3.1 PROCESE ȘI ACTIVITAȚI 26](#_Toc453780192)

[3. MODEL DE DEZVOLTARE 30](#_Toc453780193)

[GLOSAR 32](#_Toc453780194)

[PARTEA II – PROIECTAREA SISTEMULUI INFORMATIC 33](#_Toc453780195)

[4. PROIECTAREA LOGICĂ 34](#_Toc453780196)

[4.1 ARHITECTURA SISTEMULUI 35](#_Toc453780197)

[4.2 BAZA INFORMAȚIONALĂ 37](#_Toc453780198)

[5.PROIECTAREA TEHNICĂ 38](#_Toc453780199)

[5.1 PROCESE ȘI ALGORITMI 38](#_Toc453780200)

[5.2 STRUCTURA FIZICA A DATELOR 40](#_Toc453780201)

[5.3 TEHNOLOGII SPECIFICE 41](#_Toc453780202)

[PARTEA III – TESTAREA 42](#_Toc453780203)

[6. TESTAREA 43](#_Toc453780204)

[6.1 TESTAREA MANUALĂ 43](#_Toc453780205)

[PARTEA IV – MENTENANȚĂ 49](#_Toc453780206)

[8.OPERAȚII DE ÎNTREȚINERE 50](#_Toc453780207)

[CONCLUZIE 51](#_Toc453780208)

[BIBLIOGRAFIE 52](#_Toc453780209)

**TABEL DE FIGURI**

[Figură 1. Diagrama “Fish-Bone” 8](file:///F:\PSI\Vargan_Ioana.docx#_Toc453774927)

[Figură 2. Diagrama de descompunere a obiectivelor 9](#_Toc453774928)

[Figură 3. Diagrama cazurilor de utilizare 15](#_Toc453774929)

[Figură 4. Diagrama cazurilor de utilizare pentru cazul “Administrează” 25](#_Toc453774930)

[Figură 5. Diagrama de secvențe 26](#_Toc453774931)

[Figură 6. Modelul de business 28](file:///F:\PSI\Vargan_Ioana.docx#_Toc453774932)

[Figură 7. Diagrama de activități “Vizitator” 31](file:///F:\PSI\Vargan_Ioana.docx#_Toc453774933)

[Figură 8. Diagrama de activități “Administrator” 32](file:///F:\PSI\Vargan_Ioana.docx#_Toc453774934)

[Figură 9. Modelul cascasdă 33](file:///F:\PSI\Vargan_Ioana.docx#_Toc453774935)

[Figură 10. Diagrama de flux de date 38](#_Toc453774936)

[Figură 11. Diagrama client-server 39](#_Toc453774937)

[Figură 12. Diagrama de componente 41](#_Toc453774938)

[Figură 13. Structura bazei de date 45](file:///F:\PSI\Vargan_Ioana.docx#_Toc453774939)

# PARTEA I – RAPORT DE ANALIZĂ

# INTRODUCERE

De ce este important ca un cabinet stomatologic sa aiba dezvoltată o aplicație online? În primul rând, studiile arată că persoanele care sunt interesate de servicii medicale apelează la Internet pentru a alege opțiunea potrivită și au încredere în aproximativ 70 % din informațiile pe care le găsesc în mediul online . Având în vedere acest aspect, este necesar ca un cabinet stomatologic să fie prezent în mediul virtual prin intermediul unui website. Scopul acestui tip de aplicație este, în principal, de a crește numărul de clienți și de a le oferi posibilitatea de a afla detaliile de care sunt interesați referitoare la serviciile oferite de cabinetul stomatologic.

Luând în considerare dorința unei noi afaceri în domeniul stomatologic de a intra mai usor pe piață , obiectivul acestui proiect este de a prezenta analiza și proiectarea unui website cu ajutorul căruia un cabinet stomatologic urmărește să influențeze deciziile utilizatorilor prin oferirea posibilității de a face programări online precum și prin prezentarea serviciilor si a ofertelor.

Modul în care va fi implementată aplicația online conține două tipuri de interfețe: pentru client și pentru persoana care se va ocupa de partea de administrare.

Interfața client se dorește a fi prietenoasă, ceea e implică oferirea clientului oportunitatea de a naviga în mod facil printre paginile site-ului web și de a accesa rapid informațiile de care este interesat. De asemenea, se va urmări în mod constant obiectivul principal, și anume, eficientizarea relației utilizatorului cu personalul, în acest scop fiind rezervata în cadrul paginii o zonă care va conține datele de contact.

În ceea ce privește interfața pentru administrator, putem afirma că va fi simplu de utilizat, dar, în același timp, complexă, întrucât va permite administratorului să efectueze operații de creare, citire, actualizare și stergere(CRUD) asupra bazei de date .

Referitor la modul de implementare al site-ului web, tehnologiile care se vor folosi sunt C# în ASP.NET pentru programarea orientată obiect, iar ca și metodă de conectarea la baza de date vom utiliza Enity Framework.

# IDENTIFICAREA ȘI DESCRIEREA PROBLEMEI

Societatea pentru care va fi dezvoltată aplicația, “Cabinet medicină dentară Alina-Elena Gotze” a fost înființată în anul 2015 și reprezintă o afacere la început de drum, cu un singur medic stomatolog, care dorește pe viitor să se extindă. Luând în considerare contextul prezentat, putem afirma că necesitatea unui sistem informatic în această organizație are la bază absența atât a unei metode de gestiune a pacienților, cât și a unui mod eficient de prezentare a serviciilor oferite. Trebuie precizat faptul că în prezent metoda principală de prezentare a afacerii în mediul virtual este prin intermediul Facebook-ului și a clienților care au beneficiat de serviciile medicale. Așadar, obiectivul nostru general este de a furniza o aplicație care să faciliteze procesul de realizare a unei programări și accesul utilizatorilor la informațiile oferite de societate și, de asemenea, să permită o îmbunătățire a părții de administrare.

## MOTIVAȚIE

După cum am menționat în secțiunea destinată identificării probelmelor, principalele dificultăți cu care se confruntă în prezent cabinetul stomatologic sunt legate de partea de administrare și de metodele de prezentare a organizației. În ceea ce privește gestiunea pacienților, în momentul de față, medicul stomatolog ține evidența clienților prin intermediul unei agende în care sunt trecute detaliile legate de client, de serviciile de care a beneficiat și de data vizitei la medic. Așadar, dacă medicul are nevoie de istoricul unui anumit pacient, trebuie să îi caute detaliile în agendă în funcție de data în care a avut loc vizita la cabinet, proces greoi care necesită un consum de timp relativ crescut per client.

Dacă luăm în considerare partea de marketing, deși constituie un aspect elementar, este slab abordat în cazul de față. În prezent, în mediul virtual, modul de prezentare este existența unui cont de Facebook unde sunt specificate datele de contact ale cabinetului; lipsesc atât detalii legate de serviciile oferite, cât și prețurile aferente acestora. Astfel, pentru medic este mai dificil sa informeze clienții de anumite modificări care au loc și, pe de altă parte, singurul mod prin care un client poate să afle detalii legate de serviciile oferite de către cabinet este prin intermediul unui apel telefonic.

Examinând argumentele prezentate anterior , putem afirma că soluția cea mai potrivită aferentă problemei generate de lipsa prezentării este creare unui website cu o interfață client prietenoasă.

Cel mai important aspect care stă la baza creări acestei aplicații online este dorința de a facilita efortul depus atât de client cât și de medic în procesul stabilirii unei programări. Este cunoscut faptul că utilizatorii duc o viată ocupată și că își petrec o perioadă de timp semnificativă pe internet, fapt pentru care posibilitatea de a realiza programări online, de a pune întrebări sau de a afla detalii despre cabinet constituie un avantaj atât pentru medic, care nu trebuie să își răpească din timp pentru a răspunde întrebărilor, cât și pentru client, care obține mai ușor informațiile dorite.

Totodată, necesitatea existenței unei platforme online pentru cabinetul stomatologic vine din dorința medicului de a avea o relație mai strânsă cu utilizatorul care se va solda cu creșterea numarului de clienți. Așadar, este necesar ca site-ul să conțină detaliile legate de modurile în care se poate lua legătura cu medicul, cat și o prezentare detaliată a serviciilor oferite și prețul aferent acestora.

Nevoia de prezentare și promovare în scopul extinderii afacerii este unu alt motiv principal pentru a avea o aplicatie online. Cu ajutorul unui website, afacerea este “deschisă” indiferent de oră și clienții pot afla într-un mod mai usor de existența cabinetului. De asemenea, existența unui site web pentru cabinet poate crește încrederea utilizatorilor în afacere și în serviciile oferite.

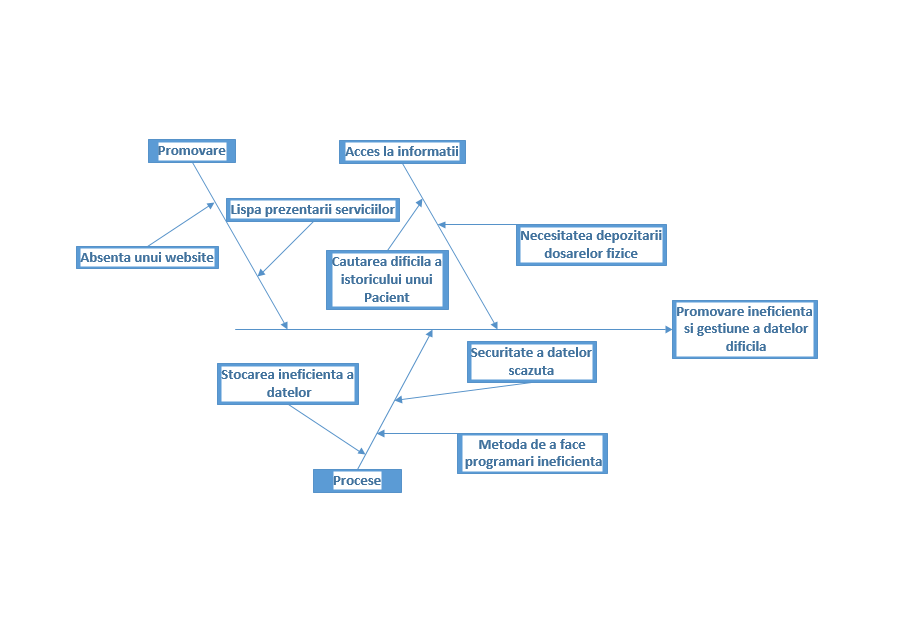
Fiind un cabinet nou pe piață, medicul isi dorește să afle părerea pacienților în ceea ce privește atât cabinetul, cât și serviciile prestate. Așadar, prin intermediul site-ului web care poate conține o secțiune dedicată observațiilor și feedback-ului, clienții își pot exprima părerea.

În ceea ce privește partea administrativă, obiectivul principal care sta la baza creării unei aplicații care să gestioneze pacienții este fluidizarea și îmbunătățirea fluxului de lucru din cabinetul medical .

Totodată, prin crearea părții administrative, se urmărește eliminarea transferului și a existenței dosarelor fizice în interiorul organizației. Având în vedere faptul că datele pacientului, tratamentele de care a avut parte și detalii legate de programările lui vor fi stocate într-o baza de date, nu va mai fi necesară existența unor documente tipărite care sa conțină aceste informații.

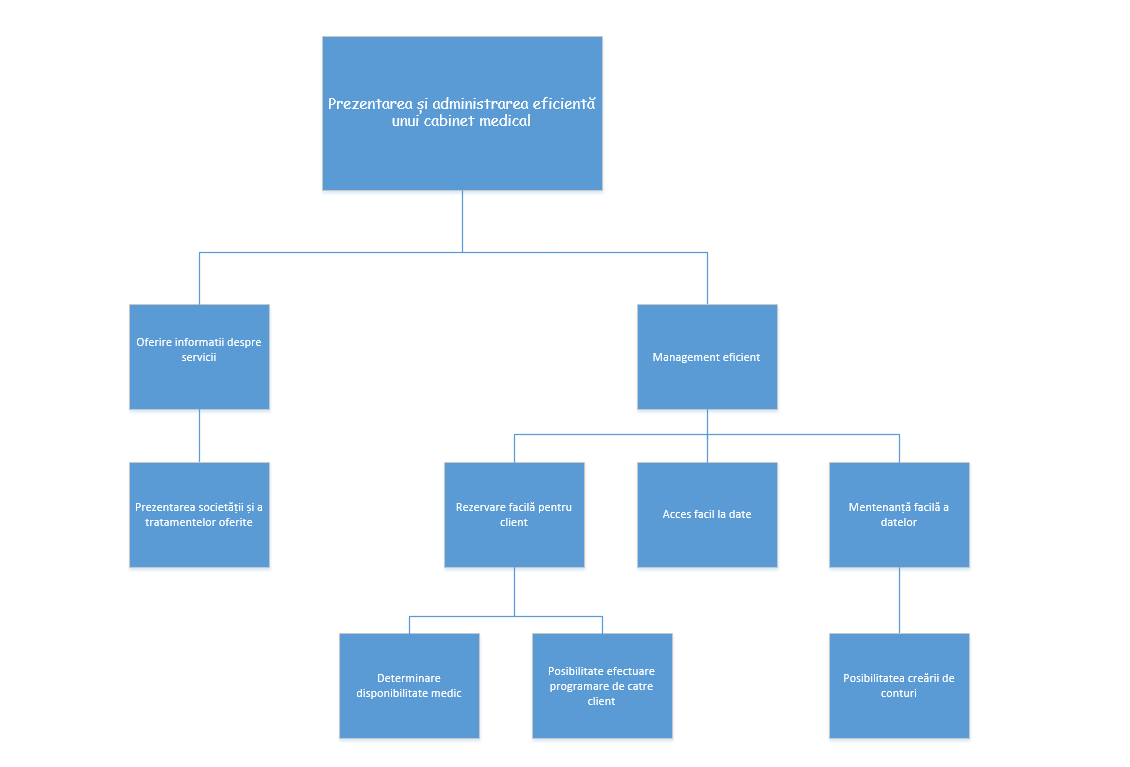
Necesitatea implementării acestei soluții are la bază și nevoia personalului medical de a avea la dispoziție informații în timp real despre datele medicale integrate ale pacientului și, de asemenea, dorința de a asigura o securitate sporită a datelor legate de pacient.

Pentru a ilustra grafic ansamblul de motivații voi folosi diagrama Fish-bone care vrea sa reprezinte legatura dintre un rezultat și factorii care au dus la apariția acestuia. Așasar, vom avea un “cap” al peștelui care constituie problema principala care trebuie rezolvată, iar apoi sunt prezentate, de-alungul “oaselor”, detalii legate de cauzele care au dus la aparitia problemei , pentru cele din urmă putând fi adăugate cauze suplimentare pe noi ramificații.

În vederea atingerii scopului principal și soluționării problemei avem nevoie de un set de obiective ce trebuie indeplinite care sunt reprezentate în diagrama de descompunere a principalelor obiective .

Figură 1. Diagrama “Fish-Bone”

Dupa cum reiese și din diagramă, principalul obiectiv este prezentarea și administrarea unui cabinet medical. În scopul realizării acestui obiectiv, trebuie să ne axăm pe aspectele principale care au legătură cu oferirea posibilității de a fi efectuate programări online prin intermediul unui cont, cu prezentarea în detaliu a serviciilor oferite de firmă și cu capacitatea de a efectua operații de creare, ștergere sau modificare a elementelor din baza de date.



Figură 2. Diagrama de descompunere a obiectivelor

## CONTEXT

Conform aspectelor prezentate în secțiunea 1.1, cabinetul stomatologic a fost înființat doar de un an, fapt care accentueaza nevoia de promovare și de integrare pe piață. Societatea își doreste dezvoltarea unui mediu de comunicare cu clientul care să faciliteze interacțiunea cu acesta, motiv pentru care este dispus să investească în crearea unui website.

Pe de altă parte, cabinetul are nevoie de o metodă de gestiune a datelor care să nu implice prezența documentelor fizice. Asadar, scopul nostru este de a furniza un sistem informatic care să satisfacă nevoile beneficiarului tocmai prezentate.

**Delimitarea Sistemului de Contextul Său**

Scopul acestei secțiuni este de a delimita și prezenta elementele care compun sistemul și, de asemenea, elementele exterioare graniței sistemului, dar cu care sistemul interacționează în prezent, lucru pe care îl vom realiza cu ajutorul metodologiei fațetelor.

1. **Fațeta subiect**

Datele care vor fi stocate și prelucrate prin intermediul aplicației online se referă la detalii cu privire la medicul care deține cabinetul stomatologic, la tratamentele pe care le oferă și, de asemenea, va fi nevoie de datele personale furnizate de pacient. Conform cerințelor Legii nr. 677/2001 pentru protecția persoanelor cu privire la prelucrarea datelor cu caracter personal și libera circulație a acestor date, cabinetul stomatologic are obligația de a administra în condiții de sigurantă și numai pentru scopurile specificate, datele personale pe care le furnizează pacientul sau persoana care face programarea. Totodată, informațiile cu caracter confidențial privind tratamentele aplicate pacientului pot fi furnizate numai în cazul în care acesta își dă consimțămânul.

Ca și stakeholderi pentru sistemul informatic prezent, identificăm doar medicul stomatolog, care este și managerul cabinetului și utilizatorii aplicației online sau potențialii clienți. În continuare, vom descrie stakeholderii și motivele pentru care sunt afectați în mod direct de aplicația care va fi dezoltată.

* Manager/medic
* Utilizează aplicația, în primul rând, pentru a prezenta detalii despre serviciile pe care le pune la dispoziția clienților și pentru a facilita efortul depus în realizarea programărilor, în ansamblu, pentru a eficientiza comunicarea cu viitorii pacienții. În al doilea rând, aplicația îl ajută să gestioneze datele despre pacienți, tratamente și programări. De asemnea, cu ajutorul aplicației, el poate determina aspecte precum tratamentele care sunt cele mai căutate sau orele la care majoritatea pacienților sunt disponibili.
* Utilizatorul website-ului
* Are posibilitatea ca, dintr-o simplă accesare a site-ului web să se informeze în legătură cu detaliile de care este interesat și, daca este mulțumit, sa faca o programare online prin care să precizeze data programării și ce tip de tratament dorește. Totodată, aplicația îi oferă posibilitatea de a lăsa un feedback referitor la seviciile prestate, daca este dornic să-și exprime părerea.

1. **Fațeta utilizare**

Aplicația online care urmează a fi dezvoltată este dedicată atât utilizatorilor Intrenetului, cât și medicului stomatolog care deține cabinetul medical.

Principalul scop al vizitatorilor site-ului web este de a se informa cu privire la tratamentele pe care le oferă medicul, detalii în legătură cu cabinetul și echipamentele folosite și, în cazul în care au fost convinși , dorința realizării unei programări.

Obiectivul managerului este de a gestiona datele legate de pacienți și de programări. Astfel, el va putea vizualiza, introduce, modifica sau șterge datele pacienților sau detalii legate de programări sau tratamente. Datorită folosirii aplicației, nu va mai fi necesară prezența unor documente fizice sau a unei persoane care să se ocupe de gestiunea datelor. Totodată, dacă medicul este interesat de rapoarte privind anumite aspecte precum numărul de clienți într-o săptămână sau totalitatea tratamentelor de un anumit tip într-o perioadă de timp, le poate realiza cu ajutorul aplicației.

Procesul prin care se trece de la starea inițială, de vizitator al site-ului, la client al cabinetului medical, este urmatorul : persoana X dorește să se intereseze ce implică o albire a dinților. Pentru aceasta, acceseaza Internetul și găsește site-ul web furnizat de noi. După ce citește prezentarea serviciului legată de albirea dinților și este mulțumit de preț, dorește sa efectueze o programare. Așadar, accesează butonul “ Programare”, moment în care este rugat să furnizeze niște detalii personale pentru a i se crea un cont. În urma creării contului, utilizatorul poate specifica detalii legate de tratamentul dorit și ora la care ar fi disponibil. Ulterior acestui proces, clientul va primi un e-mail de confirmare de la medic.

Din punctul de vedere al modului în care a fost gândită aplicația, vor exita 3 tipuri de utilizatori:

* Utizilatori la prima vizită
* Utilizatori care au deja cont
* Manager/medic
* Administrator

În funcție de categoria din care face parte, fiecare utilizator va fi supus unor politici

de acces prezentate în cele ce urmează:

* Utilizatorii la prima vizită: au acces la toate informațiile prezentate în meniul site-ului web, se urmarește atragerea lor în scopul creării unui cont
* Utilizatorii care au deja cont: pe lângă beneficiile celor care sunt la prima vizită, clienții care au cont au acces la realizarea de programări online și la oferirea de feedback
* Manager/Medic: va accesa toate detaliile în legătură cu pacienții, cu programările, va avea posibilitatea să introducă date în baza de date și, de asemenea, să creeze diferite rapoarte în funcție de nevoi
* Administratorul: se va ocupa cu gestiunea conturilor

1. **Fațeta IT**

În ceea ce privește domeniul IT, prin modul în care va fi dezvoltată aplicația se urmărește stabilirea unei comunicării facile cu clientul atât din punctul de vedere al prezentării informațiilor, cât și a transferului de date. Pentru ca utilizatorii sa aibă acces la aplicație va fi nevoie de un nume de domeniu asociat unui IP.

Limbajul de programare care va fi utilizat pentru dezvoltarea aplicației este C#, un limbaj dezoltat de Microsoft. Reprezintă un limbaj asemanator cu C++ și Java și se bucură de succes datorită calităților sale: este un limbaj simplu, cu productivitate mare în programare, este orientat pe obiecte și oferă suport complet pentru dezvoltarea de componente software. În C# se scriu, de asemenea, aplicații pentru sisteme complexe care funcționează sub o mare varietate de sisteme de operare, cât și pentru sisteme dedicate(embeded systems).

Limbajul C# funcționează având arhitectura .NET care este o componentă software care oferă un mediu ce permite dezvoltarea și rularea aplicațiilor și a serviciilor Web, independente de platformă. .NET suportă și alte limbaje, precum C++, Visual Basic sau Java.

Dacă ne referim la mediul de programare, vom folosi Microsoft Visual Studio 2015. Conform Wikipedia , „Visual Studio include un set complet de instrumente de dezvoltare pentru generarea de aplicații ASP.NET, Servicii Web XML, aplicații desktop și aplicații mobile. Visual Basic, Visual C++, Visual C# și Visual J# toate folosesc același mediu de dezvoltare integrat (IDE) care le permite partajarea instrumentelor și facilitează crearea de soluții folosind mai multe limbaje de programare. Aceste limbaje permit să beneficieze de caracteristicile .NET Framework care oferă acces la tehnologii cheie care simplifică dezvoltarea de aplicații web ASP și XML Web Services cu Visual Web Developer”.

Întrucât aplicația implică folosirea unui sistem de gestiune al bazelor de date, vom utiliza Microsoft SQL Server. Pentru a ne conecta și a avea acces la baza de date, vom folosi Entity Framework care este un framework ORM(Object/Relational Mapping) care permite dezvoltatorilor să lucreze cu date relaționale ca obiecte specifice domeniului, eliminând nevoia de scriere de mult cod pentru a accesa datele, așa cum se obișnuia să se facă. Utilizând Entity Framework, dezvoltatorii interoghează baza de date folosind LINQ, iar mai apoi preiau și manipulează datele obiecte puternic tipizate. Implementarea Entity Framework oferă servicii precum urmarirea modificărilor, identity resolution, lazy loading și traducerea interogărilor, astfel încât programatorii se pot concentra mai mult asupra logicii de afaceri specifice aplicației decât asupra aspectelor legate de accesul la date.

1. **Fateta dezvoltare**

Dezvoltarea sistemelor informatice cuprinde mai multe activităţi strâns legate între ele: analiza cerinţelor, proiectarea, implementarea, testarea şi întreţinerea, activităţi care sunt definite ca etape în ciclul de viaţă al produsului. Ciclul de viaţă are ca scop identificarea etapelor care apar în dezvoltarea de software. Aceste activităţi au o anumită succesiune în timp şi pentru îndeplinirea fiecăreia dintre ele trebuie adoptate tehnici adecvate.

Am considerat potrivit pentru aplicația noastră modelul cascadă, care este un model de dezvoltare software în care etapele sunt parcurse într-o anumită ordine, precum căderea unei cascade în care trecerea la etapa următoare implică finalizarea celei precedente. Modelul se aplică în general sistemelor în care sunt cunoscute de la început, în mod clar, cerințele pentru ca pe parcursul procesului sa nu mai fie nevoie de modificari , caz în care de încadrează și aplicația de față.

# 2. CERINȚE DE SISTEM

## 2.1 SURSE DE CERINȚE

Principalele surse de cerințe identificate sunt stakeholderii întrucât ei sunt implicați în mod direct în utilizarea aplicației și beneficiază de avantajele folosirii acesteia.

Așasar, în urma unei discuții detaliate cu medicul stomatolog care este și managerul cabinetului în legătură cu modul prezent de prezentare a afacerii și de gestionare a clienților, am identificat aspectele care trebuie îmbunatățite și modul în care se va realiza acest lucru. De asemenea, am analizat și necesitățile unor utilizatori care pot deveni viitori pacienți și vom incerca să integrăm nevoile acestora în aplicație, pentru ca aceasta să aibă succes.

În ciuda faptului că au fost unele cazuri în care stakeholderii nu au știut cu exactitate ce își doresc sau nu au reușit să-și exprime în mod clar nevoile, am încercat să ajungem la un consens și am definit cerințele aplicației.

## 2.2 ELICITAȚIA CERINȚELOR

Dacă ne referim, per ansamblu, la părțile implicate în proiect, pe lângă stakeholderi, putem să ne adăugăm și pe noi, ca persoane care dezvoltă proiectul.

Pe de o parte, beneficiul major pe care aplcația îl oferă cabinetului, este faptul că acesta poate să își promoveze activitatea și să crească numarul de clienți. Totodată, din punctul de vedere al utilizatorilor, li se va facilita procesul de a face programări și de a afla informații despre serviciile oferite de cabinet.

Pe de altă parte, avantajele din punctul nostru de vedere, al dezvoltatorlor, au legătură atât cu beneficiile financiare, cât și cu experiența dobândită. După finalizarea proiectului, putem folosi aplicația ca și model pentru alte platforme de acest gen , în scopul de a atrage clienții interesți de acest domeniu.

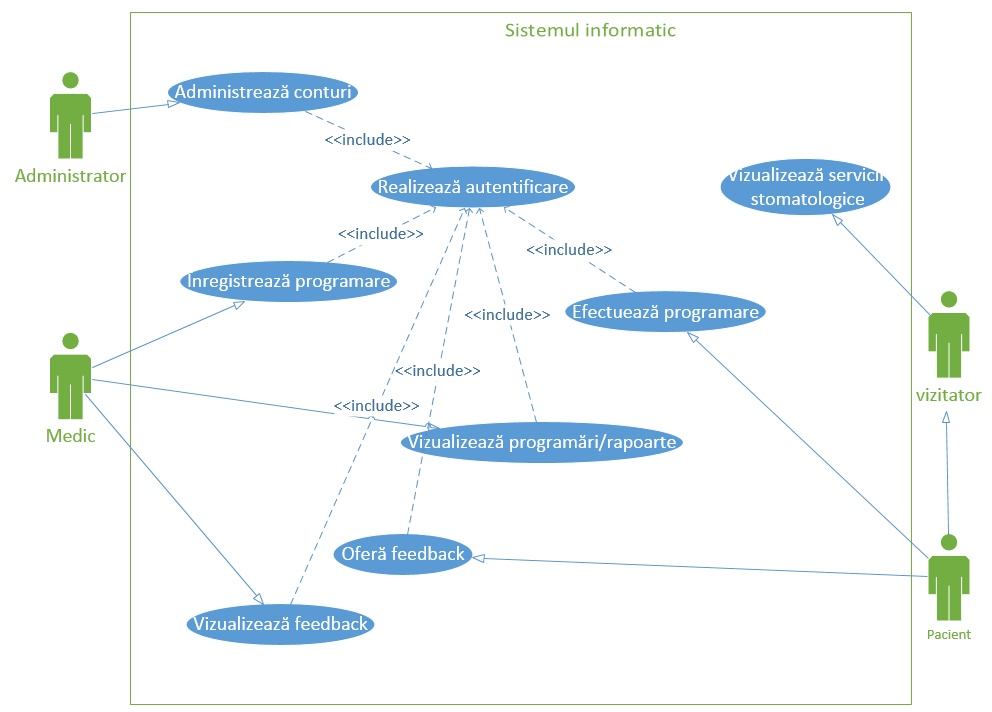
Ca și dezavantaje de care are parte beneficiarul, putem peciza faptul că nu are posibilitatea să dezvolte aplicația sau să modifice anumite trăsături, în acest sens având nevoie de noi pentru orice aspect care dorește să fie schimbat.

Din punctul nostru de vedere, în ciuda faptului că scopul nostru este de a dezvolta aplicația conform cerințelor și de a o întreține după ce este finalizată, partea care ține de mentenanță poate fi considerata un dezavataj, având în vedere faptul că în timpul consumat pentru întreținerea acestei aplicații, am putea dezvolta altele și extinde portofoliul de clienți.

Urmărind obiectivul de a înțelege mai bine necesitățile stakeholderilor și de a extrage nevoile acestora, am folosit metoda cazurilor de utilizare, a interviului și modelul de business.

**Metoda cazurilor de utilizare (use cases)**

Scopul acestei metode este de a identifica cine va interacționa cu sistemul și ce interfețe trebuie să ofere sistemul. Totodată, are un rol important în verficarea faptului că nicio cerință nu a fost uitată și că dezvoltatorii înteleg cerințele beneficiarului.

****

Figură 3. Diagrama cazurilor de utilizare

Mai sus este prezentată diagrama cazurilor de utilizare realizată prin intermediul UML(Unified Modeling Language) din Microsoft Visio.

După cum putem observa, actorii principali ai sistemului sunt :

* Vizitatorul- are doar posibilitatea de a naviga printre paginile site-ului web și de creare a unui cont.
* Pacientul- are aceleași facilități precum vizitatorul , însă este un actor care trebuie să dețină un cont creat anterior. El are acces la toate elementele interfeței: poate vizualiza detaliile despre serviciile și procedurile stomatologice puse la dispoziție, poate compara tarifele, are posibilitatea de a oferi feedback unității medicale precum și de a efectua programări online.
* Medicul- se ocupă de gestiunea internă a cabinetului stomatologic care implică următoarele: accesul la detaliile legate de programările rezervate, de pacienții înregistrați, posibilitatea de adăugare , modificare sau ștergere a seviciilor, a procedurilor și a prețurilor aferente, validarea feedback-ului înregistrat precum și vizualizarea unor rapoarte de care este interesat.
* Administratorul- rolul administratorului este de a gestiona conturile, având posibilitatea de a șterge, modifica sau crea conturi.

În urma documentării cu privire la operațiile de bază pe care beneficiarul le dorește de la sistem , s-au conturat următoarele cazuri de utilizare:

* Administrează conturi = gestionează conturile utilizatorilor(modificare, ștergere, adăugare utilizatori sau informații despre ei)
* Realizează autentificare = permite introducerea username-ului și a parolei pentru a putea fi efectuată programarea
* Vizualizează servicii = permite navigarea prin meniul website-ului
* Efectuează programare = permite pacienților să completeze formularul aferent unei programări
* Înregistrează programare = oferă medicului posibilitatea de a programa un pacient la seviciul pe care îl dorește
* Vizualizează programări/rapoarte = permite parcugerea de către medic a listei de programări și, dacă este nevoie, vizualizarea unor raporte în funcție de aspectele de care este interesat
* Oferă feedback = permite oferirea de feedback de către pacienți
* Vizualizează feedback = oferă posibilitatea medicului de a vizualiza și valida feedback-ul oferit de catre pacient

În cele ce urmează vom prezenta documentarea textuală a scenariilor de utilizare.

1. **Efectuează programare**

Acest scenariu descrie acțiunile pe care trebuie să le realizeze pacientul dacă dorește să facă o programare online.

**Actori**

Pacientul

**Prioritate**

Ridicată

**Pre-condiții**

Este necesar ca utilizatorul sa fie autentificat.

**Post-condiții**

Se va efectua o programare online și vor fi salvate detaliile aferente programării.

Pacientul primeste un e-mail de confirmare cu detaliile programării realizate.

Pacientul ramane conectat pe pagina principala.

**Cazuri de utilizare folosite**

S3-Autentificare

**Scenariu principal**

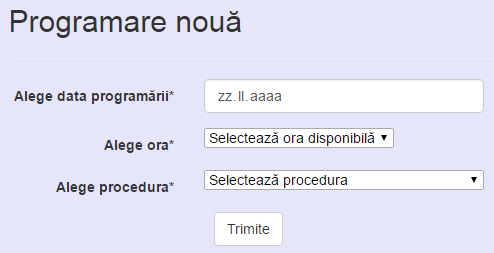
1. Autentificare – se va folosi cazul de utilizare S3.
2. Pacientul apasă pe butonul de programare din meniul principal.
3. Aplicația afișează pagina aferentă realizării unei programări.
4. Pacientul selectează data prin click pe câmpul “Alege data programării”.
5. Aplicația afișează un calendar.
6. Pacientul alege o dată din componenta calendar.
7. Pacientul selectează ora prin click pe campul “Ora”.
8. Aplicația afișează o listă cu orele disponibile pentru data aleasă.
9. Pacientul alege una din orele afișate și disponibile.
10. Pacientul alege o procedură pentru care dorește programare prin click pe butonul “Alege procedură”.
    1. Aplicația afișează o structură arborescentă cu serviciile și procedurile aferente disponibile.
11. Pacientul confirmă programarea prin click pe butonul “Trimite”.
12. Sistemul salvează datele aferente programării.
13. Pacientul este redirecționat pe pagina principală.

**Scenariu excepțional**

În cazul în care la pasul 10 se pierde conexiunea la internet:

1. Sistemul afișează un mesaj de eroare: “Lipsă conexiune la internet. Va rugăm să incercați din nou”.
2. Vizitatorul accesează din nou pagina aferentă realizării programării prin click pe butonul “Programare”.
3. Sistemul returnează formularul aferent programării.
4. Datele de autentificare ramân salvate, însă pacientul trebuie sa reintroducă datele referitoare la programare.

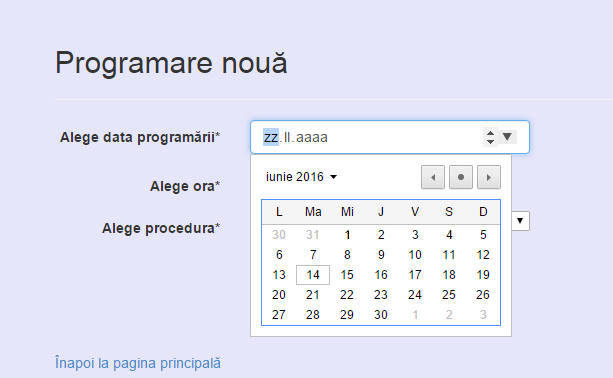
**Interfața cu utilizatorul**



**Criterii de calitate**

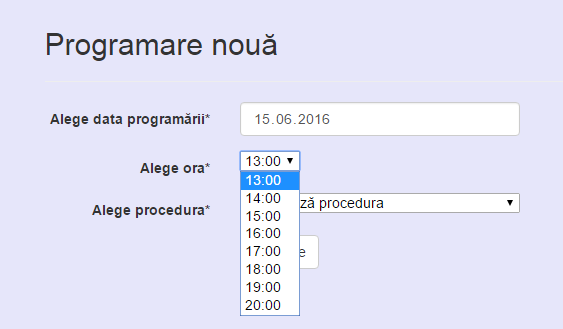
**Utilizabilitatea**

Componenta 1 :



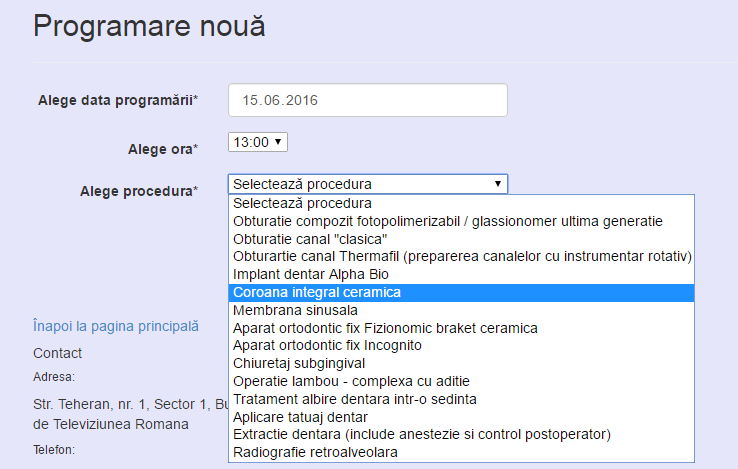
Sistemul oferă atât posibilitatea de a introduce data manual, cât și de a folosi un calendar. Pentru a ușura munca utilizatorului, data din ziua curentă este marcată implicit. De asemena, la introducerea manuală, nu se poate introduce o data anterioară celei curente. Totodată, campul oferă o valoare default care arată utilzatorului formatul în care trebuie să introducă data.

Componenta 2:



Pentru selectarea orei la care se dorește programarea, sistemul oferă o listă cu orarul la care medicul este disponibil. De asemenea, a fost setată o valoare default pentru camp,”Selectează ora disponibilă”, care oferă utilizatorului informații despre datele care ar trebui introduse.

Componenta 3 :



În procesul alegerii unei proceduri sistemul folosește o lista din care utilizatorul trebuie sa aleagă procedura pe care o dorește. Pentru a facilita alegerea unei proceduri, pacientul poate tasta litera cu care începe numele serviciului, moment în care sistemul selectează automat serviciile care încep cu litera respectivă.

Componenta 4 :



Datele introduse în formular sunt validate înainte de a putea fi trimise și în caz de eroare, apare un mesaj explicit. De asemenea, butonul „trimite” este colorat cu o nuanta mai închisă pe parcurlui introducerii datelor în formular, fapt care denotă ca este inactiv și devine activ doar în momentul în care toate câmpurile au fost completate în mod corespunzator.

1. **Înregistrează programare**

Scenariul de față descrie pașii pe care trebuie să îi urmeze medicul dacă dorește să realizeze o programare.

**Actori**

Medicul

**Prioritate**

Ridicată

**Tip scenariu**

Scenariu de tip B(scenariu de interacțiune)

**Obiective**

Înregistrarea unei programări

**Pre-condiții**

Este necesar ca medicul să fie înregistrat pentru a se putea autentifica și efectua programarea.

**Post-condiții**

Se va realiza o programare pentru un anumit pacient și vor fi salvate detaliile aferente programării.

Medicul rămâne conectat pe pagina principala.

**Cazuri de utilizare folosite**

S3-Autentificare

**Scenariu principal**

1. Autentificare – se va folosi cazul de utilizare S3
2. Medicul apasă pe butonul de programare din meniul principal.
3. Aplicația afișează pagina aferentă realizării unei programări.
4. Medicul introduce numele pacientului.
5. Dacă pacientul a fost înregistrat, sistemul afișează numele întreg și îl salvează în baza de date.
6. Dacă pacientul nu este înregistrat:
   1. Sistemul afișează mesajul: “Pacientul nu este înregistrat. Adaugă pacient nou”
   2. Sistemul oferă o lista cu campurile care trebuie completate pentru a adăuga un nou pacient.
   3. Medicul completează câmpurile
   4. Sistemul salvează datele introduse
7. Medicul dă click pe campul “Alege data programării”.
8. Aplicația afișează un calendar.
9. Medicul alege o dată.
10. Medicul dă click pe campul “ Alege Ora”.
11. Aplicația afișează o listă cu orele disponibile pentru data aleasă.
12. Medicul alege una din orele afișate.
13. Medicul accesează butonul “Alege procedură”.
14. Aplicația afișează procedurile stocate în baza de date.
15. Medicul selectează serviciul de care vrea să beneficieze.
16. Medicul apasă butonul “Trimite”.
17. Sistemul salveaza datele aferente programării.

**Scenariu excepțional**

În cazul în care la pasul 16 se pierde conexiunea la internet:

1. Sistemul afișează un mesaj de eroare: “Lipsă conexiune la internet. Va rugăm să incercați din nou”.
2. Medicul accesează din nou pagina aferentă realizării programării prin click pe butonul “Programare”.
3. Sistemul returnează formularul aferent programării.
4. Datele de autentificare ramân salvate, însă medicul trebuie sa reintroducă datele referitoare la programare.

**Criterii de calitate**

Activitățile se pot face într-un timp scurt și fără prea mult efort, fapt care crește utilizabilitatea aplicației.

1. **Vizualizare programări**

Acest scenariu descrie activitățile pe care le urmează medicul pentru a putea vizualiza programările care s-au făcut.

**Actori**

Medicul

**Prioritate**

Ridicată

**Tip scenariu**

Scenariu de tip B(scenariu de interacțiune)

**Obiective**

Vizualizarea programărilor efectuate în scopul unei organizări mai bune a timpului.

**Pre-condiții**

Este necesară existenta anterioară a unui cont cu rolul de doctor pentru a putea realiza autentificarea și a avea acces la interfața specifică medicului.

**Post-condiții**

Sistemul va afișa cu succes lista de programări cerută.

**Cazuri de utilizare folosite**

S3-Autentificare

**Scenariu principal**

1. Medicul se autentifică- se va folosi cazul de utilizare S3
2. Aplicația afișează pagina aferentă medicului
3. Medicul alege opțiunea “Listă de programări”
4. Medicul alege data pentru care dorește să vadă programările.
5. Sistemul afișează lista cu detaliile despre programările din data introdusă: ora, pacientul și tipul serviciului.

**Scenariu excepțional**

În cazul în care la pasul 2 se pierde conexiunea la internet:

1. Sistemul afișează un mesaj de eroare prin care notifică utilizatorul despre lispa conexiunii.
2. Informațiile despre autentificare rămân salvate.
3. Utilizatorul accesează pagina aferentă medicului.
4. Se repetă scenariul principal incepând cu pasul 2.

**Criterii de calitate**

Activitățile se pot face într-un timp scurt și fără prea mult efort, fapt care crește utilizabilitatea aplicației.

1. **Oferă feedback**

Senariul de utilizare prezent descrie pașii care trebuie urmați de către un pacient pentru a oferi feedback

**Actori**

Utilizatorul care are cont de pacient.

**Prioritate**

Ridicată

**Tip scenariu**

Scenariu de tip B(scenariu de interacțiune)

**Obiective**

Oferirea unui feedback

**Pre-condiții**

Este nevoie ca utilizatorul să fi fost înregistrat pentru a se putea autentifica și oferi feedback.

**Post-condiții**

Se va salva în baza de date feedback-ul oferit pentru ca mai apoi să poată fi afișat utilizatorilor.

**Rezultat**

Feedback înregistrat și în așteptare pentru a fi moderat

**Scenariu principal**

1. Pacientul foloște cazul de utilizare S3 pentru logare
2. Sistemul afișează pagina principală
3. Pacientul alege din meniul de sus opțiunea “Testimoniale”
4. Pacientul apasă butonul “Adaugă testimonial”
5. Sistemul afișează formularul pentru adăugare de feedback.
6. Pacientul completează campul:”Descriere”
7. Pacientul apasă butonul: “Trimite”
8. Sistemul salvează datele introduse de către pacient

**Scenariu excepțional**

În cazul în care la pasul 9 se pierde conexiunea la internet:

1. Sistemul afișează un mesaj de eroare prin care notifică utilizatorul despre lispa conexiunii.
2. Utilizatorul accesează din nou pagina aferentă oferirii de feedback.
3. Sistemul afișează formularul pentru adăugarea de feedback.
4. Datele de autentificare ramân salvate, însă trebuie sa introducă din nou detaliile despre feedback. .
5. Se reia scenariul principal de la pasul 6 .
6. **Autentificare**

Acest scenariu descrie procesul prin care un utilizator se autentifica în aplicație și, de asemenea, specifică drepturile de acces pentru diferiți utilizatori.

**Actori**

Actorul principal este utilizatorul care poate fi pacient, administrator sau medic.

**Prioritate**

Ridicată

**Tip scenariu**

Scenariu de tip B(scenariu de interacțiune)

**Obiective**

Autentificarea în scopul obținerii accesului la informații sau a posibilității de a face programări.

**Pre-condiții**

Este necesară întregistrarea anterioară pentru a obține un cont prin intermediul căruia să se poată realiza autentificarea și, de asemenea, o buna conexiune la internet

**Post-condiții**

Utilizatorul este autentificat și poate efectua operațiile de care este interesat.

**Rezultat**

Realizarea autentificării în cadrul aplicației

**Scenariu principal**

1. Sistemul afiseaza pagina de log-in
2. Utilizatorul introduce numele si parola
3. Aplicația validează datele introduse
4. Aplicația afișează pagina aferentă tipului de cont introdus.

**Scenariu excepțional**

După ce s-au introdus username-ul și parola, dacă acestea au fost introduse greșit:

1. Sistemul afișează mesajul de eroare: “Username sau parola greșită”
2. Se reia scenariul principal de la pasul 2.

Pentru o mai bună înțelegere a cazurilor de utilizare, voi prezenta, pe scurt, aspectele principale a celorlate cazuri de utilizare.

***Nume***: **Vizualizează servicii stomatologice**

***Actori***: vizitator

***Scenariu principal****:*

1. Vizitatorul selectează din meniu una dintre opțiunile : “Despre noi”,”Servicii”,”Tarife”,”Testimoniale”,”Contact”
2. Sistemul afișează pagina corespunzătoare opțiunii alese

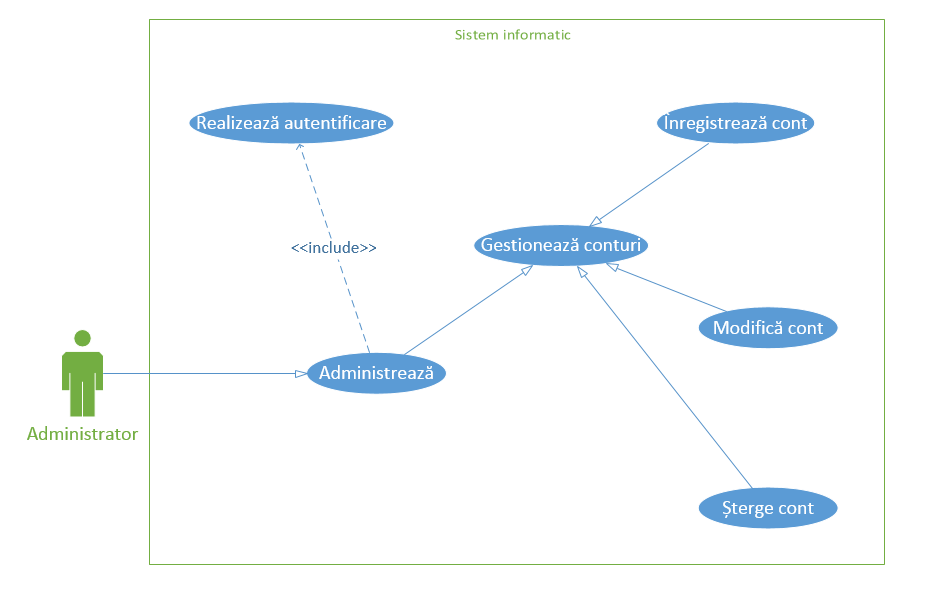
***Nume***: **Administrează conturilor**

***Actori***: administratorul sistemului

***Descriere***: acțiunile pe care le realizează administratorul când dorește adăugarea sau moficarea unui utilizator

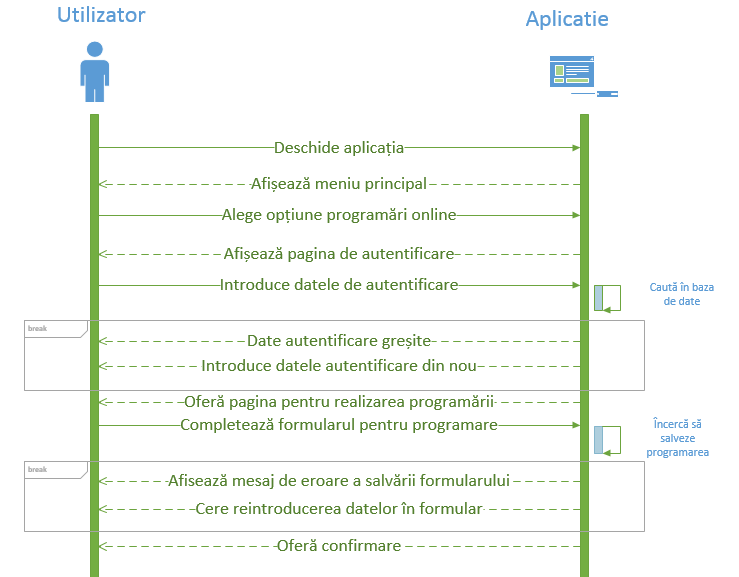
***Scenariu principal:***

1. Foloște cazul de utilizare S3 pentru logare cu drept de administrator
2. Administratorul alege una dintre opțiunile posibile: Înregistrează cont, Modifică cont, Șterge cont
3. Sistemul salvează modificările



Figură 4. Diagrama cazurilor de utilizare pentru cazul “Administrează”

Diagrama de secvență prezentată mai jos este folosită pentru documentarea scenariilor de utilizare. Această diagramă reprezintă un factor important deoarece reflectă interacțiunile actorilor cu sistemul în timp și , totodată, prezintă în mod clar diferitele tipuri de scenarii.



Figură 5. Diagrama de secvențe

**Metoda interviului**

Interviul este o metodă care are rolul de a facilita înțelegerea problemelor unei organizații și a posibilelor soluții din perspectiva stakeholderilor. Această tehnică poate fi folosită atât în analiza problemei cât și în faza de elicitație a cerințelor.

Întrebări generale și specifice cuprinse într-un interviu cu beneficiarul:

1. Care este modul principal de prezentare în mediul virtual al afacerii în momentul de față?

Avem o pagina de facebook unde sunt prezentate datele de contact.

1. Ce metodă este folosită pentru gestiunea programarilor și a pacienților ?

Am o agendă unde notez datele despre pacient și detaliile legate programări.

1. Cum credeți ca poate fi îmbunătățit modul actual de prezentare ?

Având în vedere importanța Internetului în ziua de azi, aș dori un site web în care să fie prezentate detaliile legate de serviciile pe care le ofer , datele de contact și, dacă se poate, aș dori ca utilizatorii sa aibă posibilitatea sa faca programări online.

3.1 Ce informații sunt necesare pentru efectuarea unei programări ?

Voi avea nevoie de datele generale ale pacientului : nume, prenume, număr de telefon, adresă de e-mail, de serviciul de care vrea să beneficieze și de ora la care este disponibil.

3.2 Doriți ca aplicația să trimită confirmarea programării ?

Da, aș dori să se trimită un e-mail fiecărui client cu confirmarea rezervarii.

1. Ati dori și gestionarea proceselor interne ale aplicatiei sau doar partea de client ?

Da, aș vrea să am o aplicație care să îmi permită să văd pacienții, tratamentele de care au avut parte, programările, fără să mai fiu nevoită să pierd timp prin căutarea lor în agendă.

4.1 Cum va avea loc colectarea de informații ?

Informațiile vor fi introduse prin intermediul formularelor, de către pacienți și de către mine.

1. Ce așteptări aveți cu privire la gradul de utilizabilitate al aplicației ?

Îmi doresc ca aplicația să permită realizarea programărilor cu ușurință, utilizatorii să poată găsi rapid informatiile de care au nevoie.

* 1. Care vor fi utilizatorii aplicației ?

Aplicația va fi utilizată atât de mine, care sunt și manager și medic , cât și de potențialii clienti, care vor fi vizitatorii site-ului web. De asemenea, va fi nevoie de un administrator care să se ocupe de baza de date, întrucât eu nu am cunoștintele necesare.

5.2 Care va fi limitarea de acces pentru fiecare categorie de utilizatori?

Utilizatorii la prima vizită vor avea acces doar la informațiile depsre servicii și la posibilitatea de a-și crea un cont , cei care sunt deja înregistrați, vor putea, pe lângă aspectele precizate mai sus, să efectueze programări online, iar eu voi avea acces la informațiile stocate în baza de date despre pacienți, servicii și programări.

**Modelul de business**

Scopul principal al modeului de business este să ne ajute să ințelegem organizația, oferindu-ne o vizualizare abstractizată a organizației și a proceselor sale. De asemenea, ne indică relațiile cu terții și modul de organizare în teritoriu a componentelor organizației. Modelul de business contribuie indirect la eficientizarea organizației si poate conduce la reingineria proceselor.

****

Figură 6. Modelul de business

**Workshop de cerințe**

Această metodă de elicitație reunește toți stakeholderii pentru o întalnire de scurtă durată, care are un scop precis, și anume, acela de a vedea părerea și nevoile fiecăruia. Printre avantajele generate de această metodă regăsim promptitudinea rezultatelor, senzația de transparență a procesului și posibilitatea ascultării fiecărei opinii.

În cazul de față, noi, dezvoltatorii, am organizat o întâlnire cu medicul care deține cabinetul stomatologic, 2 dintre pacienții acestuia și o persoană la întamplare care nu știa pana la acel moment de existența cabinetului.

Pe parcursul întâlnirii s-au efectuat sesiuni de brainstorming, iar la finalul ei, s-au împărțit câteva chestionare. Scopul acestui workshop a fost să vedem necesitățile utilizatorilor și părerile acestora în legătură cu aspectul și conținutul site-ului.

Concluziile în ceea ce privește dorințele utilizatorilor au fost următoarele:

* Existența unei interfețe prietenoase
* Specificarea detaliilor clare cu privire la serviciile orferite
* Prezența unor testimoniale
* Aspect simplu și clar
* Modalitate facilă de a realiza programarea online
* Informații precise legate de contact
* Necesitatea unor e-mail-uri de confirmare

## 2.3 FORMULAREA CERINȚELOR

Cerințele sunt împărțite în două categorii : cerințe funcționale și cerințe calitative. Cerințele funcționale reprezintă ceea ce poate să realizeze sau nu sistemul, ca reacție la un anumit input sau output sau în anumite situații, pe când cerințele calitative se referă la condițiile în care va realiza sistemul funcționalitățile.

Cerințe funcționale - sistemul trebuie să :

* Poată oferi utilizatorilor informațiile necesare despre serviciile oferite
* Aibă un meniu intuitiv, din care utilizatorul să extragă informația cu ușurință
* Ofere utilzatorului o modalitate de creare a unui cont
* Fie capabil să valideze datele introduse în formulare
* Ofere detalii în legătură cu disponibiliatatea medicului
* Fie capabil să genereze diferite rapoarte de care are nevoie medicul
* Să confirme programarea făcută
* Permită adăugarea facilă de utilizatori și date noi în baza de date
* Să nu permită introducerea unor date incorecte în formulare
* Ofere o siguranță crescută a datelor

Cerințe calitative :

* Aplicația trebuie sa aibă o utilizabilitate, mentenabilitatate și testabilitate crescută.

### 2.3.1 PROCESE ȘI ACTIVITAȚI

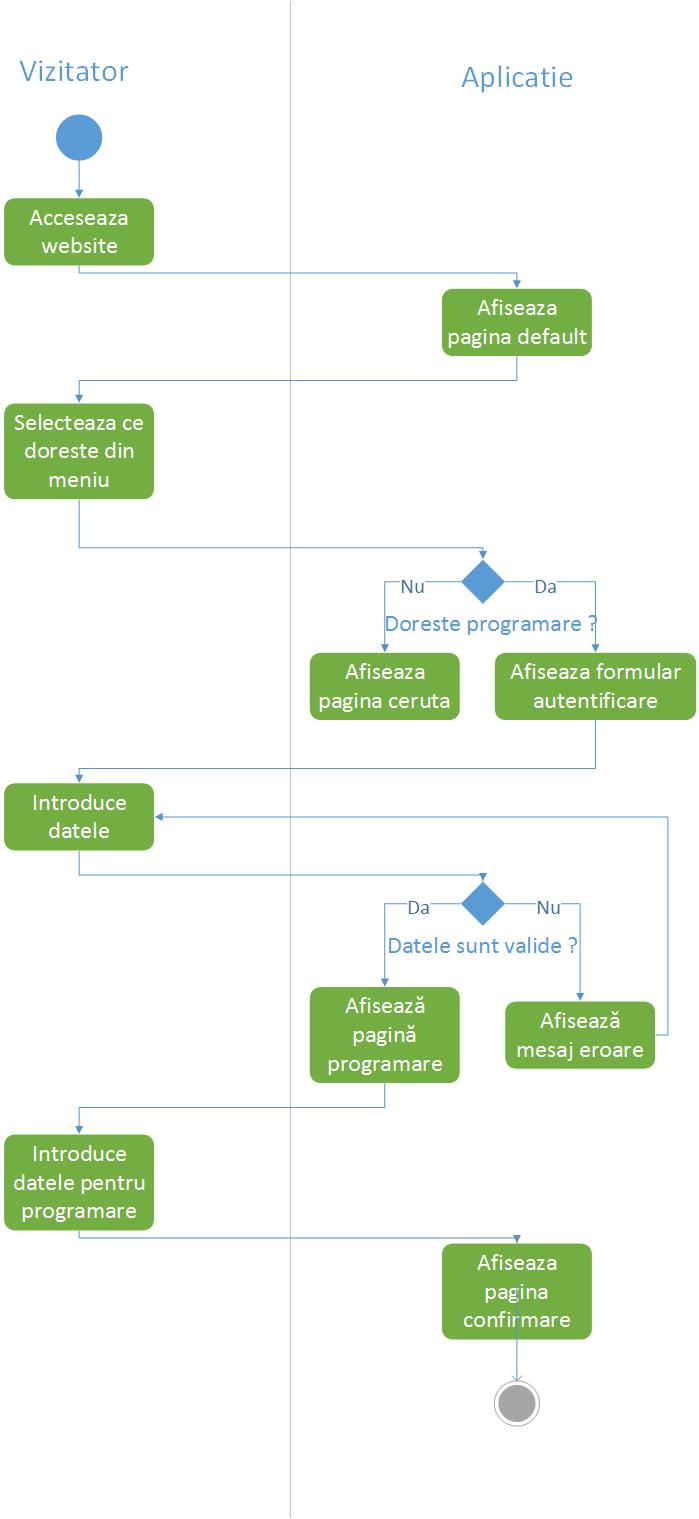
Obiectivul principal al aplicației este privit din două puncte vedere: partea de prezentare a cabinetului stomatologic și partea de administrare.

În ceea ce privește prezentarea, în momentul de față, afacerea are un cont de Facebook care conține detalii cu privire la adresa cabinetului și date de contact. De asemenea, metoda principală prin care medicul își promovează afacerea este răspândirea informației “din gură în gură”. Astfel, dacă pacienții au fost mulțumiți de serviciile primite și de modul în care au fost tratați, este evident faptul că vor recomanda acest cabinet altor persoane.

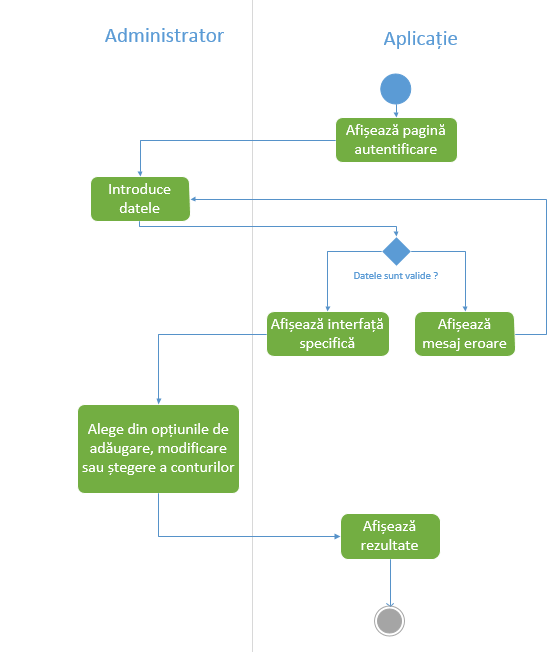
Din punctul de vedere al părții de administrare, momentan, gestiunea pacienților se face prin intermediul unei agende. Așadar, dacă un pacient sună la cabinet pentru a face o programare, medicul își notează datele personale, tratatamentul de care vrea să beneficieze clientul și detaliile despre programare într-o agendă.

Pentru a îmbunătăți sistemul existent, ne propunem să dezvoltăm o aplicație web care să cuprindă atât partea de prezentare, cât și cea de administrare. Astfel, aplicația va avea o interfață client care va conține un meniu unde sunt prezentate serviciile oferite de către cabinet și, de asemenea, va da posibilitatea clientului să facă programări online. Pentru aceasta, va fi necesar un cont care să conțină datele pacientului și, pentru a se putea realiza o programare, va trebui completat formularul cu datele necesare,date care vor fi stocate în baza de date. După efectuarea programării, clientul va primi un e-mail de confirmare în care doctorul poate accepta sau reprograma pacientul. Totodată, aplicația va avea și o interfață pentru administrator sau manager, persoane care o sa aiba acces la baza de date, putând face operații de adăugare sau modificare asupra tabelelor și, în cazul administratorului, poate gestiona conturile. În diagramele urmatoarele se observă principalele activități pe care le execută atât aplicația cât și medicul sau un vizitator al site-ului web.

Din punctul de vedere al vizitatorului, acesta acceseaza site-ul web, moment în care apare pagina principală de unde poate alege opțiunile dorite. Are la dispoziție meniul prin care poate naviga, iar după ce s-a interesat de serviciile și prețurile aferente și doreste o programare, poate accesa butonul “Programare” care îl va redirecționa spre pagina de creare a unui cont, în cazul în care utilzatorul nu este înregistrat, sau pe pagina de completare a datelor aferente programării. După trimiterea formularului, utilizatorul va fi redirecționat pe pagina principală a aplicației.



Figură 7. Diagrama de activități “Vizitator”



Figură 8. Diagrama de activități “Administrator”

# 3. MODEL DE DEZVOLTARE

După cum am precizat în secțiunea 1.2 dedicată fațetei dezvoltare, modelul care se pliază cel mai bine pe dezvoltarea sistemului prezent este modelul cascadă. Dacă luăm în considerare trăsăturile acestui model, putem afirma ca întregul proces de dezvoltare software este împărțit în faze separate, în care output-ul uneia se comportă ca și input pentru următoarea etapă. În continuare voi prezenta succind principalele etape ale modelului cascadă.

Cerințe

Design

Implementare

Testare

Mentenanță

Figură 9. Modelul cascasdă

În etapa de analiză a sistemului și a cerințelor sunt colectate toate necesitățile stakeholderilor pentru a stabili cum vor fi acestea cuprinse în aplicație. Cerințele sunt un set de funcționalități și constrângeri pe care beneficiarul se așteaptă să le aibă sistemul. Ele sunt extrase atât de la medicul stomatolog care deține cabinetul, cât și de la persoanele care vor vizita site-ul web. În urma finalizării acestei etape, se va genera un document care va cuprinde cerințele și va fi folosit în etapa următoare a modelului.

Faza de design este, de fapt , un proces care cuprinde mai mulți pași. Pentru început, încercăm să ne concentrăm pe anumite atribute ale sistemului cum ar fi structura de date, arhitectura de sistem, reprezentarea interfeței sau detalii legate de algoritmii folosiți. De asemenea, încercăm să transpunem cerințele în reprezentări ale software-ului care pot fi evaluate din punct de vedere al calității înaine să începem implementarea propriu-zisă. Scopul este de a ne asigura că design-ul satisface necesitățile utilizatorilor și este de bună calitate și, totodată, că nu există interpretări eronate ale unor cerințe.

În etapa de implementare, se folosește ca și input design-ul generat în faza anterioară, care va fi trasformat în cod ce poate fi compilat și executat. Aici vom scrie secțiuni de cod care sa fie ușor de înțeles și de modificat. De asemenea, fiecare secțiune va fi testată pentru a găsi eventualele erori și pentru a stabili dacă există discrepanțe între specificațiile din design și implementarea propriu-zisă.

Testarea are un rol foarte important în asigurarea calității software-ului. Având în vedere că metodele de verificare aferente etapei anterioare sunt limitate, aici vom testa din toate punctele de vedere atât codul, cât și aspecte legate de interfață și de funcționaltățile sistemului pentru a ne asigura că toate aspectele sunt conform cerințelor.

Etapa de mententanță implică folosirea sistemului de către client, motiv pentru care putem afirma că nu are un final bine definit. Având în vedere faptul că în timpul utilizării aplicației pot apărea diferite probleme care nu au fost prevăzute nici în cerințe sau design și nu au fost detectate în etapa de testare, dezvoltatorul trebuie să fie permanent disponibil să le revolve. Totodată, există posibilitatea ca beneficiarul să dorească implementarea unor noi functionalități ale sistemului sau modificarea celor existente, aspecte care vor fi tratate tot în cadrul etapei de mentenanță.

Modelul cascadă prezintă anumite avantaje și dezavantaje. Unul dintre principalele avantaje îl constituie controlul total asupra fazelor, faptul că procesul și rezultatele sunt foarte bine documentate. De asemenea, funcționează foarte bine pentru proiectele mici, unde cerințele sunt înțelese foarte bine și este ușor de însușit pentru membrii echipei de proiectare. Un alt aspect ține de partea de testare care este relativ usoară având în vedere că se poate face pe baza scenariilor prezentate în cerințe și, totodata , un avantaj este că erorile de design sunt detectate înainte ca sa se scrie cod efectiv, fapt care evită consumul de timp în cadrul fazei de implementare. Un beneficiu este și faptul că poate fi estimat un cost total al proiectului încă din faza de analiză, după ce au fost definite cerințele.

Principalul dezavantaj al modelului cascadă îl constituie faptul că cerințele nu pot fi schimbate în timpul procesului de dezvoltare. Beneficiarilor o sa li se pară de multe ori dificil sa exprime în mod cert necesitățile, fapt pentru care vor aprecia la nivel general funcționalitățile pe care și le doresc de la sistem. Un alt dezanataj îl reprezinta timpul alocat dezvoltării care poate fi relativ ridicat, având în vedere că trebuie finalizată fiecare etapă.

# GLOSAR

1. Ajax = prescurtare pentru Asynchronous JavaScript and XML, este o tehnică de programare pentru crearea de aplicații web interactive. Ajax face posibilă comunicarea cu serverul farî a fi nevoie să încarce o nouă pagină.
2. Bootstrap = este un framework dezvoltat inițial de Twitter, ajuns acum la versiunea 3, care permite realizarea de site-uri web responsive, care se adaptează la orice rezoluție de dispozitiv: desktop, tablete și telefoane mobile.
3. Feedback-ul = este o formă specifică de mesaj (reacție la mesajul inițial ) prin care se poate cunoaște impactul comunicării emitentului în comportamentul destinatarului (percepția, interpretarea și redarea mesajului inițial), se dă informatie despre consecințele unui anumit comportament și despre posibilitatea de adaptare și schimbarea situației.
4. Jquery = este o platformă de dezvoltare JavaScript, concepută pentru a ușura și îmbunătăți procese precum traversarea arborelui DOM în HTML, managementul inter-browser al evenimentelor, animații și cereri tip AJAX. JQuery a fost gândit să fie cât mai usor posibil si disponibil în toate versiunile de browsere importante existente.
5. MVC = este unul dintre modelele de programare ASP.NET; este un framework pentru a dezvolta aplicații web folosind design-ul MVC: Model- reprezintă datele la nivel logic, View - afișează datele, Controller = se ocupă de interacțiunea cu user-ul. Separarea MVC te ajută să administrezi aplicații complexe pentru că poți să te concentrezi pe rând pe fiecare aspect . Spre exemplu, te poți concentra pe view fără să depinzi de logica afacerii.
6. Mentenabilitatea = este proprietatea sistemului de a putea fi ușor menținut sau readus (restabilit) în stare de bună funcționare, într-un timp cât mai scurt. Activitățile legate de menținerea sau readucerea sistemului în stare de bună funcționare, se numesc activități de mentenanță.
7. Procedură = Serviciul este efectuat cu ajutorul mai multor proceduri, spre exemplu, pentru serviciul implantologie, există procedurile : implant dentar Nobel Biocare, adiție os Bio-Oss dinte Monoradicular, coroană integral ceramică etc.
8. Programare online = este termenul folosit atunci când dorești să faci o vizită medicului stomatolog pentru o anumită problema și trebuie să mergi la o anumită dată și oră când acesta este disponibil. Așadar, în loc ca pacientul să sune la cabinetul medical, intră pe website și completează formularul aferent programării unde este nevoit sa precizeze data, ora și serviciul de care vrea să beneficieze.
9. Utilizabilitatea = capabilitatea produsului software de a fi înţeles, învăţat, utilizat şi considerat atractiv de către utilizator, atunci când este folosit în condiţii specificate.
10. Serviciu stomatologic = un fel de tratament de care ai parte în momentul în care te hotărăști să faci o vizită unui medic stomatolog; exemple: endodonție, implantologie, tratamente estetice, ortodonție etc.

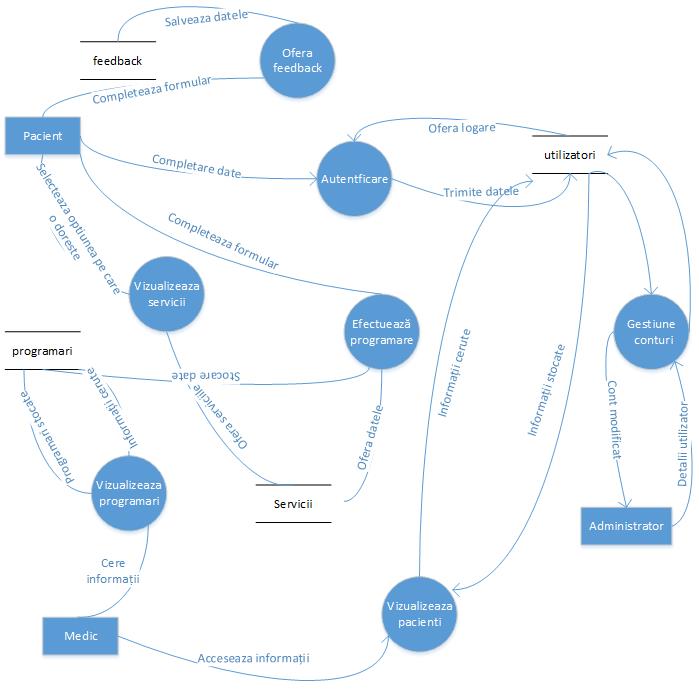
# PARTEA II – PROIECTAREA SISTEMULUI INFORMATIC

# 4. PROIECTAREA LOGICĂ

Primul aspect pe care îl vom lua în considerare în ceea ce privește proiectarea logica este legat de modul de prelucrare a datelor la nivel de sistem. Am ales să folosim un sistem de baze de date centralizat, adică atât datele cât și sistemul de gestiune sunt stocate pe un singur calculator. Această abordare prezinta urmatoarele avantaje: o bună integritate a datelor cu redundanță minimă(adică aceleași date nu vor apărea de mai multe ori în locuri diferite), asigurarea consistenței prelucrărilor și gestiunea facilă a tranzacțiilor.

**Diagrama flux de date**

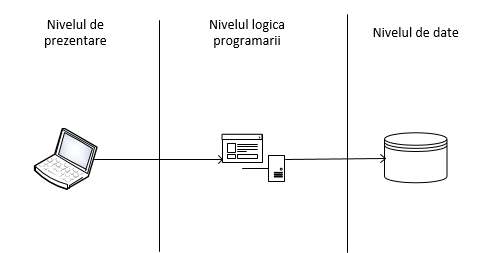
Diagrama fluxului de date (DFD) este o reprezentare grafica a “fluxului” de informații care circulă printr-un sistem și este folosită, de asemenea, pentru vizualizarea procesării informațiilor. Elementele diagramei sunt constiuite din: procese, care reprezintă niște activități pe care sistemul trebuie să le realizeze, activități ce primesc un input, prelucrează datele și produc un rezultat, surse de date care se referă la actorii care comunică cu sistemul și depozitele de date care stocheaza informațiile necesare proceselor.



Figură 10. Diagrama de flux de date

## 4.1 ARHITECTURA SISTEMULUI

Sistemul are la bază o arhitectură client/server pe trei nivele și anume : nivelul client sau prezentare, nivelul logica de programare sau server de aplicatii și nivelul de date. Acest tip de arhitectură a fost ales datorită urmatoarelor aspecte care sunt îmbunătățite față de celelalte opțiuni: performanță, scalabilitate, flexibilitate, mentenabilitate și reutilizabilitate.



Figură 11. Diagrama client-server

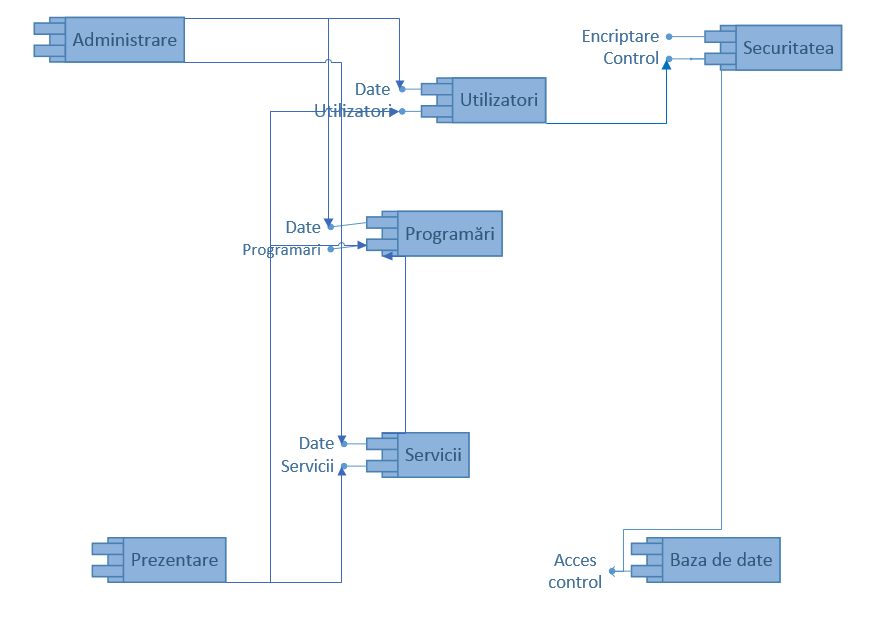
Primul nivel al arhitecturii client/server pe trei niveluri este reprezentat de interfaţa sistemului cu utilizatorul (ferestre de dialog, ferestre pentru introducerea datelor,administrarea afişării etc.), adică logica prezentării. Acest nivel rulează pe calculatorul utilizatorului și este reprezentat de browser-ul din care se accesează aplicația.

Nivelul de logica programului (server-ul de aplicații) este nivelul în care sunt înglobate diversele module de prelucrare a datelor. Acest nivel de mijloc asigură îmbunătățirea aspectelor amintite mai sus prin centralizarea logică a proceselor. Această centralizare face ca administrarea şi gestiunea modificărilor să fie mai uşoară prin localizarea funcţionalităţii sistemului astfel încât modificările sunt executate o singură dată şi plasate pe server-ul nivelului de mijloc pentru a fi disponibile de-a lungul întregului sistem.

Al treilea nivel, cel de baze de date, are ca și funcționalitate gestiunea bazei de date şi este folosit de către serviciile de fişiere şi date, referindu-se la logica accesului la baza de date şi validarea din partea server-ului. Componenta de gestiune a datelor garantează că datele sunt consistente de-alungul sistemului prin utilizarea unor caracteristici precum blocarea datelor, consistenţa şi replicarea acestora.

Diagramele de componente reprezintă relaţiile dintre componentele sistemului descris. O componentă este considerată independentă şi oferă altor componente acces la anumite funcţionalităţi prin intermediul unor interfeţe. În mod uzual diagramele de componentele sunt utilizate pentru descrierea la nivel înalt a structurii sistemului. În cadrul sistemului nostru, am identificat următoarele componente:

* Utilizatori - componenta care se ocupă cu activitățile pe care le poate realiza un utilizator
* Servicii – componenta care conține toate detaliile despre servicii și modul în care sunt folosite în aplicație
* Programări – componenta care se ocupă de gestionarea programărilor și tot ce implică realizarea unei programări
* Administrare – componentă care reprezintă de fapt tot ce implică partea de administrare și, după cum se observă, are acces la toate datele oferite de celelalte componente
* Prezentare – componentă care se referă la interfața cu utilizatorul și care poate să acceseze doar partea de prezentare a celorlalte componente, din motive de securitate
* Baza de date – componentă care stochează datele și se ocupă cu gestionarea tuturor datelor din cadrul sistemului
* Securitate – componentă importantă care are două intefețe: una de criptare și una de control al accesului de care este legată componenta “Utilizatori” pentru a permite sau a interzice accesul utilizatorilor la celelalte date. Partea de criptare se referă, în principiu, la storcarea parolelor utilizatorilor în baza de date pentru a nu putea fi accesate de nimeni.



Figură 12. Diagrama de componente

## 4.2 BAZA INFORMAȚIONALĂ

Luând în considerare aspectele menționate anterior în partea de analiză, am decis să structurăm datele în funcție de principalele funcționalități pe care beneficiarul dorește ca sistemul să le aibă . Astfel pentru activitatea de autentificare, vom stoca date referitoare la rolurile pe care le vor avea utilizatorii, gradul de acces aferent fiecăruia, precum și date personale necesare medicului . În ceea ce privește partea de prezentare a serviciilor oferite de cabinet, vom separa serviciile de procedurile aferente fiecăruia și vom specifica datele pe care trebuie să le cunoască pacientul cu prvire la proceduri. Activitatea de efectuare a unei programări va conține detalii din fiecare pare prezentată anterior , precum și detalii pe care le vor introduce utilizatoriul referitor la data în care dorește să fie programat. Ultimul aspect se referă la feedback-ul pe care îl va oferi pacientul, acesta având posibilitatea să ofere steluțe în funcție de diferite criterii stocate în baza de date.

# 5.PROIECTAREA TEHNICĂ

## 5.1 PROCESE ȘI ALGORITMI

Funcționalitatea cea mai importantă a aplicației noastre, cu ajutorul căreia website-ul se diferențiază de celelalte aplicații de acest gen este efectuarea de programare online. Pentru aceasta, în controller avem metoda Index care stă în spatele paginii care va afișa formularul. În pagină va fi afișată data, ora și procedura pe care urmează sa o selecteze utilizatorul, informații care vor fi salvate în baza de date cu ajutorul metodei SaveProg. Pentru a afișa o listă cu orele care sunt disponibile pentru o dată anume, am construit 2 liste, una cu orele la care lucrează, în general, medicul și una cu orele la care au fost deja făcute programări pentru o anumită dată. Mai apoi, din lista nu toate orele am eliminat cele ocupate și am afișat orele disponibile.

public ActionResult Index()

{

var x = new SelectList(db.Proceduras, "Id", "Denumire");

var prog = new ProgViewModel();

prog.Pacient\_Id = User.Identity.GetUserId();

prog.ServiciuProcedura = x;

prog.Data = DateTime.Now;

prog.Ora = \_allhours;

return View(prog);

}

public ActionResult FilteredValues(DateTime date)

{

var occupiedRows = db.Programares.Where(x => x.Data.Year == date.Year &&

x.Data.Month == date.Month &&

x.Data.Day == date.Day);

var occupiedHours = occupiedRows.Select(x => new SelectListItem() {

Text = x.Data.Hour.ToString() + ":00",

Value = x.Data.Hour.ToString()

});

var allhours = \_allhours.ToList();

foreach (var item in occupiedHours)

{

var toRemove = allhours.First(x => x.Value == item.Value);

allhours.Remove(toRemove);

}

return Json(allhours, JsonRequestBehavior.AllowGet);

}

O altă funcționalitate pe care o considerăm relevanta pentru utilizator este afișarea serviciilor, a procedurilor și a prețurilor aferente. Pentru aceasta am construit un view model care conține exact datele pe care vrem să le afișăm în pagină și anume : denumirea serviciului și o listă de proceduri, fiecare procedură având deja stocată în baza de date prețul aferent. Mai apoi, în controller, am creat o metodă care parcurge lista de servicii din baza de date și, pentru fiecare serviciu, selectează procedura aferentă.

public ActionResult Tarife()

{

var \_db = new StomaEntities();

var \_tarife = new List<TarifViewModel>();

foreach (var \_serviciu in \_db.Servicius.ToList())

{

var \_procList = new List<Procedura>();

var t = \_db.ServiciuProceduras.Where(x => x.Serviciu\_ServiciuID == \_serviciu.ServiciuID).Select(y => y.Procedura).ToList();

\_tarife.Add(new TarifViewModel() { DenumireServiciu = \_serviciu.Denumire, Procedura = t });

}

return View(\_tarife.ToList());

}

În view-ul aferent controller-ului, am parcurs fiecare item din model și am afișat denumirea serviciului și din datele despre procedură, denumirea și prețul acesteia.

@foreach (var item in Model)

{

<dl>

<dt>

@Html.DisplayFor(modelItem => item.DenumireServiciu)

</dt>

@foreach (var item1 in item.Procedura)

{

<dt>

@Html.DisplayFor(i => item1.Denumire)

@Html.DisplayFor(i => item1.Pret)

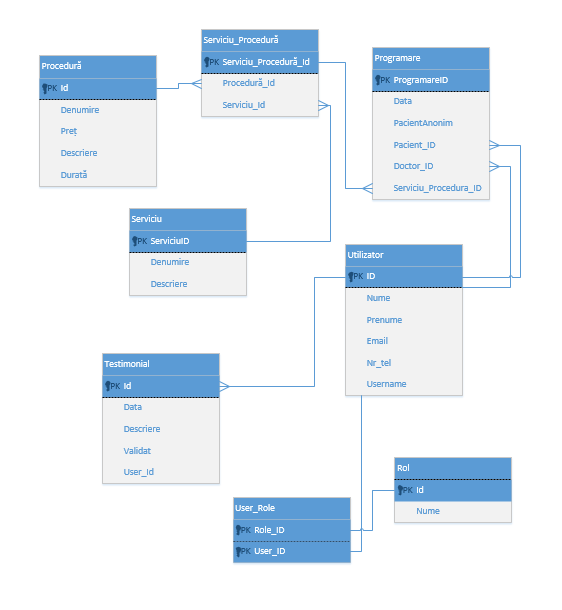
</dt>

}

</dl>

}

## 5.2 STRUCTURA FIZICA A DATELOR

În figura următoare este reprezentată baza de date, unde putem vedea structura fiecărei tabele. După cum am precizat în secțiunea despre baza informațională, tabelele sunt grupate astfel încât pe baza lor să se poată executa operații de manipulare a datelor care sa satisfacă cerințele beneficiarului. Relațiile dintre tabele au la bază chei primare și chei secundare, iar datele sunt inițializate în funcție de tipul definit. Tipurile de date folosite în majoritatea cazurilor au fost varchar, integer și date.

Figură 13. Structura bazei de date

## 5.3 TEHNOLOGII SPECIFICE

Pentru dezvoltarea aplicației web am ales să utilizăm framework-ul .NET care este un cadru (Framework) de dezvoltare software unitară care permite realizarea, distribuirea şi rularea atât a aplicaţiilor-desktop Windows cat şi aplicaţiilor WEB. Tehnologia .NET pune laolaltă mai multe tehnologii (ASP, XML, OOP, SOAP, WDSL, UDDI, WPF, LINQ) şi limbaje de programare (VB, C++, C#, F#) asigurând totodată atât portabilitatea codului compilat între diferite calculatoare cu sistem Windows, cât şi reutilizarea codului în programe, indiferent de limbajul de programare utilizat.

Ca limbaj de programare am ales limbajul C# care a fost dezvoltat de o echipă restrânsă de ingineri de la Microsoft, echipă din care s-a evidențiat Anders Hejlsberg. Principiile de bază ale programării pe obiecte (încapsulare, moștenire, polimorfism) sunt elemente fundamentale ale programării C#. În mare, limbajul moștenește sintaxa și principiile de programare din C++. Sunt o serie de tipuri noi de date sau funcțiuni diferite ale datelor din C++, iar în spiritul realizării unor secvențe de cod sigure (safe), unele funcțiuni au fost adăugate (de exemplu, interfețe și delegări), diversificate (tipul struct), modificate (tipul string) sau chiar eliminate (moștenirea multiplă și pointerii către funcții). Unele funcțiuni (cum ar fi accesul direct la memorie folosind pointeri) au fost păstrate, dar secvențele de cod corespunzătoare se consideră ”nesigure”.

ASP.NET este tehnologia Microsoft care permite dezvoltarea de aplicații web moderne, utilizând platforma Microsoft .NET cu toate beneficiile sale. Tehnologiile server ASP.NET sunt următoarele: Web Pages (cu sintaxă Razor), MVC (Model View Controller), Web Forms (ASP.NET tradițional). Pentru aplicația noastră am ales să folosim ASP.NET MVC. Model-View-Controller (MVC) separă o aplicație în trei componente principale: modelul, view-ul și contreller-ul. Framework-ul ASP.NET MVC oferă o alternativă la pattern-ul Web Forms pentru crearea de aplicații web; este un framework de prezentare ușor de testat care este integrat cu caracteristicile existente ASP.NET cum ar fi paginile master și autentificarea bazată pe membership.

Datorită faptului că în aplicația noastră avem nevoie de un SGBD (sistem de gestiune a bazelor de date), datele vor fi salvate într-o bază ce va fi de tipul Microsoft SQL Server. SQL Server este un DBMS (Data Base Management System) –sistem pentru gestiunea bazelor de date, produs de Microsoft. Suportă versiunea Microsoft de SQL (Structured Query Language) - limbaj structurat de interogări, cel mai comun limbaj pentru bazele de date. Este un sistem din clasa Enterprise ce se poate aplica bazelor de date de dimensiuni foarte mari. Ultima versiune aparută este Microsoft SQL Server 2008. Versiunea SQL Server 2008 , lansată pe 6 august 2008 cu nume de cod Katmai aduce și ea, aşa cum era de aşteptat, alte îmbunătăţiri. Acum oferă suport pentru stocarea datelor multimedia și adaugă noi tipuri de date (geometry, geography, hierarchy și mult așteptatul date fără datetime).

# PARTEA III – TESTAREA

## TESTAREA

Testarea este faza cea mai vitală pentru finalizarea unui proiect. Testarea, în principal, are ca scop verificarea modularității, a fluxului de date și a codului, astfel examinând complexitatea sistemului care este în curs de dezvoltare. Pentru această aplicație s-a efectuat o validare detaliată pentru toate aspectele care necesitau acest lucru. Testarea este partea esențială a dezvoltării sistemului pentru asigurarea calitații serviciilor, a proiectării și a codului.

**Obiectivele testării**

Obiectivul testării este aspectul estențial care ajută la efectuare unei testări corecte, aceasta reprezentând procesul de executare a unui program cu scopul de a decoperi erorile. Principalul obiectiv este să se proiecteze teste care ca descopere în mod sistematic diferite tipuri de erori într-un timp cat mai scurt. Testarea demostrează ca funcționalitățile sistemului sunt îndeplinite în mod corect și ca cerințele legate de performanță sunt indeplinite.

**Metodologia testării**

* Testarea Blackbox– primul tip de testare este la nivel de utilizatori unde toate aspectele aplicației au fost testate pentru a vedea dacă funcționează pentru utilizator.
* Testarea Whitebox– al doilea tip de testare este la nivel funcțional, unde fiecare modul al aplicației este testat. Acest test continuă în întreg proiectul, evoluând de la nivelul de testarea pe componente , la cel de testare integrată.
* Testarea pe componente– acest tip de testare se concentrează pe cele mai mici componente ale aplicației. După ce codul a fost dezvoltat, analizat și verificat să corespundă cu proiectarea la nivel de componentă, începe testarea pe fiecare componentă.
* Testarea integrată– testarea pe componente asigură faptul că fiecare componentă funcționează corect independent, dar acum trebuie testat sistemul ca și un întreg; trebuie dezvoltată o strategie care verifică faptul că întreg sistemul funcționează cum trebuie.

### 6.1 TESTAREA MANUALĂ

Pentru o testare de calitate, este important să existe o planificare riguroasă a testării încă din faza de proiectare sau dezvoltare. Pe măsură ce se conturează definițiile modulelor, entităților de date, obiectelor, claselor, funcțiilor, etc. este recomandabil să se scrie și “scenarii” de testare ale acestora, care, în industria software, se numesc test cases. Pentru fiecare din modulele următoare, am dezvoltat câte un test case.

**Modulele care vor fi testate:**

* Autentificarea unui utilizator
* Înregistrarea unui utilizator
* Efectuarea unei programări

**Autentificarea unui utilizator**

1. Click pe “Autentificare” din meniu → ar trebui să ma redirecționeze spre pagina unde să completez formularul



1. Click pe “Trimite” fără a completa username-ul sau parola → ar trebui să rămână aceeași pagină încărcată și să se afișeze un mesaj de eroare pentru fiecare căsuță



1. Completare username corect, parolă incorectă sau nulă, după care click pe “Trimite” → ar trebui să rămân în pagină și să se afișeze un mesaj de eroare



1. Completat username sau parolă greșită sau ambele greșite, după care click pe “Trimite” → ar trebui să rămân în pagină și să se afișeze un mesaj de eroare



1. Completat username corect, parolă corectă, după care click pe “Trimite” → trebuie să fiu corect autentificat, mă redirecționează pe pagina principală și apare în dreapta sus un text de salutare care conține numele de utiliator

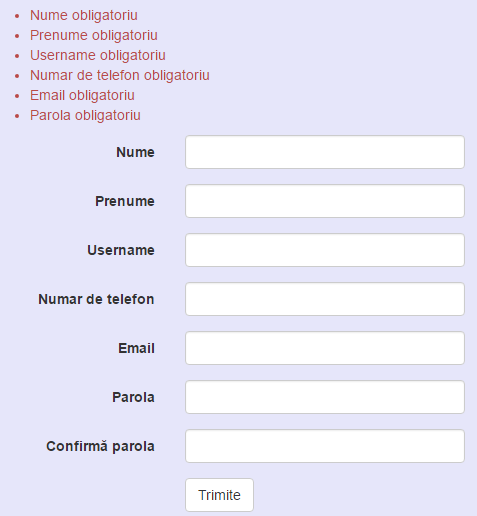


**Înregistrarea unui utilizator**

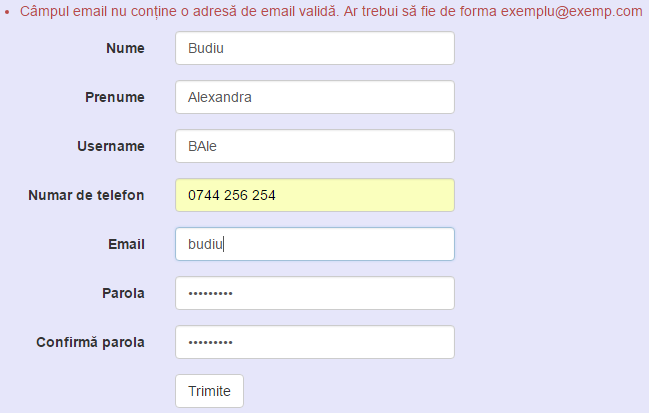
1. Click pe “Înregistrare” din meniu → ar trebui să ma redirecționeze spre pagina unde să completez formularul



1. Click pe “Trimite” fără a completa datele care se cer → ar trebui să rămână aceeași pagină încărcată și să se afișeze un mesaj de eroare cu toate căsuțele care trebuie completate



1. Completare una dintre căsuțe incorect, după care click pe “Trimite” → ar trebui să rămân în pagină și să se afișeze un mesaj de eroare

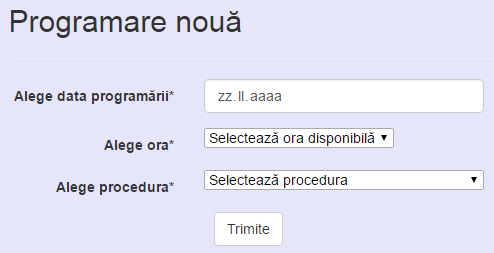


1. Completat toate datele corect, după care click pe “Trimite” → trebuie mă redirecționeaze pe pagina principală și apare în dreapta sus un text de salutare care conține numele de utilizator; automat sunt autentificat cu datele tocmai introduse

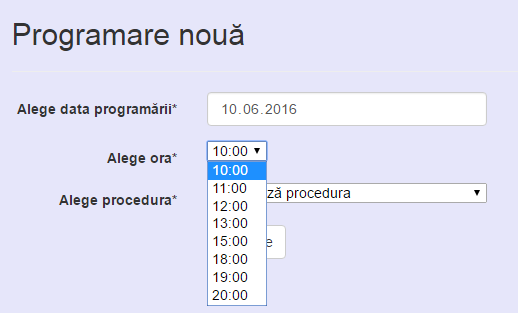


**Efectuarea unei programări**

1. Click pe butonul “Programare” din meniu → ar trebui să ma redirecționeze spre pagina unde să completez formularul; dacă apăs pe dată, va apărea un calendar, iar dacă apas pe oră, va apărea o lista cu toate orele la care lucrează, în general, medicul



1. Aleg data programării → ar trebui ca lista de ore să se schimbe și să afișeze doar orele la care medicul este disponibil în ziua respectivă



1. Aleg câte o valoare pentru toate câmpurile, după care dau click pe “Trimite” → datele despre programare se salvează și ma redirecționează pe paina de confirmare



# PARTEA IV – MENTENANȚĂ

## 8.OPERAȚII DE ÎNTREȚINERE

Întreținerea site-ului web este o activitate concepută pentru a devolta o aplicație care rulează fără probleme. Acesta poate fi realizată în mod regulat, sau programată după cum este necesar. Unul dintre motivele pentru a efectua lucrări de întreținere este de a actualiza site-ul web. Este important să se păstreze informații actualizate și să se asigure că sistemele utilizate pentru a rula site-ul sunt, de asemenea, actualizate. Având cele mai recente programe și pe partea de back-end, website-ul va fi păstrat mai sigur și le va permite oamenilor să țină pasul cu schimbările.

Serviciile de întreținere includ:

* administrarea bazelor de date
* introducerea în website a noilor servicii pe care firma le prestează: informații, oferte, poze, galerii foto etc.
* verificarea funcționalității site-ului
* implementarea noilor tehnologii web
* înscrierea în direcetoare web

În ceea ce privește mentenanța legată de website-ul nostru, vom lua în considerare următoarele aspecte:

* Îmbunătățirea design-ului

Este posibil ca pe viitor medicul să își dorească o schimbare a stilului în care sunt organizate paginile , să existe o interfață mai prietenoasă . Se pot adăuga diferite efecte care să atragă utilizatorii, prezența mai multor imagini cu cabinetul sau cu efectul tratamentelor efectuate

* Adăugarea de noi funcționalități

Având în vedere faptul că în prezent este un singur medic, se poate pune problema de extindere a afacerii în sensul angajării unor noi medici, ceea ce impune prezența unor schimbări în partea de back-end, actualizarea bazei de date etc.

* Integrarea aplicației pe alte platforme

Dacă medicul consideră ca website-ul are succes, utilizatorii își fac programări online și dorește extinderea acestuia în partea de mobile , echipa noastră ar putea dezvolta aplicația și în iOS sau Android

# CONCLUZIE

Importanţa Internetului în zilele noastre este deja cunoscută iar exploatarea oportunitaţilor oferite de acesta trebuie să fie o prioritate pentru o afacere nouă pe piață mai ales dacă concurenţa este deja prezentă online. Un website costă o singură dată şi rămâne permanent, în orice moment al zile, în atenţia publicului din orice zonă a lumii. Oricând, conţinutul unui website poate fi actualizat sau modificat după dorințele beneficiarului.

Scopul acestei lucrări a fost proiectarea unei aplicații web pentru un cabinet stomatologic și s-a urmărit schimbarea metodelor de gestiune din cadrul cabinetului și a celor de prezentare și oferire de informații despre serviciile prestate. Am implementat o parte de administrare, care ajută medicul să gestioneze serviciile, pacienții și aspectele care sunt prezentate în site-ul web și o interfață pentru utilizator care îi permite să navigheze prin paginile website-ului pentru a afla informațiile de care este interesat, să facă programări online și să ofere feedback.

Procesul de elaborare a lucrării de licență a avut un mare impact asupra fixării cunoștințelor dobândite pe parcursul facultății întrucât a cuprins elemente din cele mai importate materii predate. Totodată, am avut oportunitatea de a asimila noi cunoștinte referitoare la tehnologii diferite și de a pune în parctică informațiile pe care facultatea le-a oferit în mod teoretic. Un aspect foarte important a fost experiența dezvoltării unei aplicații de la partea de analiză la cea de implementare. Am aflat ce implică obiectivele unei aplicații, gradul ridicat de importanță a cerințelor și modul în care acestea pot fi extrase și cum aceste aspecte influențează partea de proiectare. De asemenea, am înțeles relevanța părții de testare și ponderea pe care o are în cadrul procesului de implementare. Am învățat ce implică gestiunea timpului în proiectarea unei aplicații și cum apariția unor erori poate influența foarte mult perioada de implementarea și, de asemenea, ideea că nu poți anticipa cu exactitate cât timp este necesar pentru a dezvolta o aplicație.

În concluzie, putem afirma că am îndeplinit obiectivele propuse și am reușit să dezvoltăm o aplicație care să satisfacă cerințele beneficiarului și, totodată, am avut oportunitatea de a experimenta procesul de proiectare a unei aplicației, parcurgând toți pașii necesari.

# BIBLIOGRAFIE

* Torgersen, Mads (October 27, 2008). "New features in C# 4.0". Microsoft. Retrieved October 28, 2008.
* Documentația oficială Microsoft pentru Visual Studio 2013
* „Code Behind vs. Code Inline”, Microsoft .NET Framework, Microsoft. Archived from the original on 11 November 2010. Retrieved 2010-11-22.
* Adrian Ni­a, Maria Ni­a, Nicolae Olaroiu, Rodica Pintea, Cristina Sichim(2008). “Introducere în .NET Framework”
* Whitten J., Bentley L. & Dittman, K., System Analysis and Design Methods (5 Ed.) Mc-Graw Hill: Boston.2001
* Trevor R.Wood, Implementation Specialist, “Solving Appointment/Arrival Management Dilemmas with Technology: for Sate Sales and Use Tax Agencies.
* “Web Application Architecture”, principles, protocols and practices by Leon Shklar, Richar Rosen
* <http://www.uav.ro/files/finalizare/Disertatie_NastasescuL.pdf>
* <http://www.rasfoiesc.com/educatie/informatica/internet/Medii-de-dezvoltare-web-NET-SI53.php>
* <https://prezi.com/sfonc_qmzbrp/thesis-online-patient-management-system/>
* <http://revistaie.ase.ro/content/28/ogrinja.pdf>