**UNIVERSITATEA DIN BUCUREȘTI**

**FACULTATEA**

**DE**

**MATEMATICĂ ȘI INFORMATICĂ**

**SPECIALIZAREA**

**BAZE DE DATE SI TEHNOLOGII SOFTWARE**

**Lucrare de disertatie**

**FlyBird – Sistem informatic si de gestiune al zborurilor**

**Coordonator științific Absolvent**

**Banu Iulia Teodora Vătafu Ștefan**

**București, Iulie 2023**

Cuprins

[Cuprins 2](#_Toc111905836)

[Abstract 1](#_Toc111905837)

[1.Introducere 2](#_Toc111905838)

[*1.1 Motivarea alegerii temei* 2](#_Toc111905839)

[*1.2 Prezentarea generala a proiectului* 3](#_Toc111905840)

[2. Arhitectura proiectului 5](#_Toc111905841)

[3.1 Frontend 7](#_Toc111905842)

[3.1.1 Angular 7](#_Toc111905843)

[3.1.2 Bootstrap 8](#_Toc111905844)

[4. Concluzii și implementări viitoare 11](#_Toc111905845)

[Bibliography 12](#_Toc111905846)

Abstract

Scopul acestui proiect este realizarea unui sistem informatic si de gestiune al zborurilor. Obiectivul este acela de a implemneta o aplicatie web completa, complexa si usor de manevrat pentru toate tipurile de utilizatori, dar si pentru administratorii acesteia.

Această aplicație web va fi împărțită în doua mari categorii: Frontend și Backend utilizându-se pentru fiecare diferite tehnologii care vor fi detaliate mai pe larg în capitolele ce vor urma.

Prima categorie, partea de la Frontend, are scopul de a realiza o interfata dinamica si usor de manevrat pentru utilizatori. Pentru realizarea ei se va folosi un framework de Java Script, Angular, ce se bazeaza pe Typescript si Bootstrap pentru adaugarea componentelor de interfata, dar cuprinde si o serie larga de librarii utile in implementarea de noi functionalitati.

A doua categorie a proiectului, Backend-ul, a fost realizata folosind limbajul de programare Java, folosind pattern-ul arhitectural Spirng MVC (Model-View-Controller) pentru a realiza RESTful Web Services. Împreună cu Spring MVC vom mai folosii tehnologii precum Hybernate care ne va ajuta la maparea domeniilor la o bază de date relațională și framework-ul Spring Security pentru autentificare și autorizare.

Ca și sistem de gestiune a bazelor de date relaționale vom utiliza MySQL care este un SGBD open-source.

1.Introducere

## *1.1 Motivarea alegerii temei*

Motivul alegerii unei aplicatii Web in domeniul aviatic ca si tema de disertatie a aparut datorita domeniului in care am studiat si datorita dorintei de a realiza o astfel de aplicatie utilizand diferite tehnologii care combinate sa produca un produs finit.

Mai mult decat atat aceste aplicatii sunt foarte des intalnite in zilele noastre si acest fapt se datoreaza avantajelor acestora:

* Compatibilitatea dintre sisteme – majoritatea aplicatiilor web trebuie sa indeplineasca o cerinta minima si anume existenta unui browser precum (Micrososft Edge, Google Chrome, Safari), astfel indiferent de sistemul de operare folosit aplicatiile web pot rula cu usurinta.
* Reducerea Costurilor – aplicatiile web pot reduce costurile atat pentru furnizor datorita costurilor mici pentru suport dar si pentru partea de intretinere cat si pentru utilizator deoarece sistemul acestuia are nevoie de specificatii minime pentru a putea utiliza aplicatia.
* Mult mai gestionabil – dupa cum am spus mai sus aplicatiile web necesita cerinte minimme din partea statiilor de lucru ale utilizatorului si impreuna cu avantajul ca acestea sunt instalate doar pe server, actualizarea sistemului este se poate face foarte rapid si foarte simplu.

## *1.2 Prezentarea generala a proiectului*

În această lucrare de disertație am ales să dezvolt un sistem informatic al unei clinici medicale, mai exact o aplicație web care se va focusa atât pe pacienți cât și pe partea de administrare.

Numele aplicației este NxTClinic și va avea o pagină principală în care se vor găsi informații generale despre clinică. Navigarea pe pagină se va face din bara de navigare din partea de sus a site-ului. Din această zonă, se poate naviga foarte usor către diferite secțiuni ale site-ului:

Din perspectiva unui pacient vom avea urmatoarele zone care vor contine diferite functionalitati:

* Detalii despre fiecare specializare medicală care se găsește în cadrul clinicii, acestea se vor putea accesa prin simpla apăsarea pe butonul Specialization care este în esență un dropdown button, reusind astfel să fie accesate categoriile pe care pacientul dorește să le vizualizeze;
* Următoarea secțiune este cea a medicilor unde se vor afla date despre fiecare medic precum: Nume, Prenume, Specializarea pe care acesta operează și o poză a acestuia. Această sectiune o sa fie vizibilă și accesibilă doar pentru utilizatorii care se vor înregistra și loga pe situl clinicii;
* A treia secțiune prezentă este Contact, unde se găsesc datele de contact (email, telefon, pagină de facebook, pagină de instagram) ale clinicii și locația acesteia pe harta, pinuită;
* A patra secțiune este secțiunea de abonamente și consultații care se va putea accesa în urma conectarii userului în aplicație. În aceasta pagină, utilizatorul poate să își programeze o consultație la unul dintre doctori sau iși poate achiziționa un abonament care va conține diferite beneficii în cadrul clinicii. După achiziționarea abonamentului sau a înregistrarii programării la o consultație acesta va primi un email care îi va oferii detaliile necesare;
* A cincea sectiune va face parte din pagina de profil a utilizatorului, unde se vor afla date personale despre acesta ca de exemplu: Nume, Prenume, Data nasterii, Oras, CNP, Numar de telefon, Email.
* Ultima zonă este reprezentată de pagina de autentificare și de înregistrare, unde pacienții își pot crea contul și ulterior se pot autentifica beneficiind de secțiunile menționate mai sus. To aici, pacientul va putea sa isi schimbe parola in cazul in care acesta a uitat-o.

Din perspectiva unui administrator vor exista toate paginile pe care le poate vizualiza și pacientul cu toate funcționalitățile acestora, însă va avea și o secțiune specială numită Administration care va reprezenta interfața de administrare a aplicației, specială doar utilizatorilor cu rol de admin, unde se vor putea efectua și vizualiza următoarele date:

* În secțiunea de AdministrationDoctors, se va putea vizualiza lista tuturor medicilor, unde se vor putea adăuga sau șterge diferiți medici și se vor putea face modificări asupra datelor acestora;
* Secțiunea AdministrationSubscription, unde se va putea vizualiza lista tuturor abonamentelor unuia sau mai multor clienți. În această listă, administratorul va putea face operații CRUD asupra abonamentelor;
* Secțiunea AdministrationMedicalExamination, unde se va putea vizualiza lista tuturor consultațiilor unuia sau mai multor pacienți. În această listă, administratorul va putea face operații CRUD asupra consultațiilor, desigur la solicitarea clienților;
* Va avea acees la datele de logare ale pacienților pe care se vor putea face diferite modificări precum: schimbarea parolei, schimbarea emialului;
* Vor exista dashborduri care vor prezenta diferite aspecte despre activitatea userilor pe platformă precum: numărul de logări ale acestuia într-o perioadă de timp;
* Va avea acces la secțiunea Specialization unde va putea face operații CRUD asupra enităților prezente.

Pe langă cele menționate mai sus, va exista posibilitatea filtrării entităților în tabelele descrise, astfel reusind să se eficientizeze și să se ușureze procesul de căutare făcut de către administrator în privința anumitor date.

2. Arhitectura proiectului



Figură 1. Arhitectura Apliației Web. Sursă Proprie.

În continuare vom analiza diagrama de mai sus:

Frontend-ul reprezintă componenta aplicației web, pe care utilizatorul o poate vedea și cu care poate interacționa. Această componentă a apliacției este împărțită la rândul ei în două sub-categorii: partea estetică (design-ul) și dezvoltarea implementării interfeței. Partea estetică a aplicației se referă la: meniuri, navigatia platformei, designul butoanelor, fonturile, caruselele cu imagini. În timp ce dezvoltarea aplicației se focusează pe modul în care de exemplu, funcționează butoanele și dacă acestea îndeplinesc corect funcționalitățile pentru care au fost create.

Frontendul nu face logică de business adică nu prelucrează datele. Scopul lui este doar de a colecta datele și de a le afișa utilizatorului sub o formă cât mai prietenoasă. Acesta va face un Request catre Backend acolo unde se face prelucrarea de date.

Pentru a se procesa datele avem nevoie de un server de Backend. Aceasta este componenta care stochează datele și care comunică cu baza de date, comunică cu diferite sisteme de fișiere și poate comunica chiar cu alte sisteme de backend și face toate prelucrările necesare. După aceasta etapa a prelucrărilor, serverul va trimite un Response către aplicația de Frontend, iar aceasta la rândul ei va afișa rezultatele către user.

3. Tehnologii folosite

3.1 Frontend

Pentru implementarea Frontend-ului ne vom ajuta de Visual Studio Code care reprezinta un editor de cod sursa foarte puternic reusind sa ruleze pe toate sistemele de operare. Acest tool este foarte utilizat datorita suportului cu care vine incorporat pentru JavaScript, TypeScript si Node.js. Cu toate acestea a devenit foarte populat si pentru limbaje de programare precum Java, C# sau Python datorita ecosistemului foarte bogdat de extensii pentru aceste limbaje. [[1](#Vis22)]

3.1.1 Angular

Angular este o platforma de dezvolatare construita de catre TypeScript. Acesta este proiectat sa trimita update-urile facute cat mai rapid, avand astfel posibilitatea sa profiti de acestea aproape instant.

Fiind o platforma Angular include:

* Un framework pentru construirea de aplicatii web scalabile;
* O colectie variata de librarii integreate care ajuta la implementarea diferitor caracteristici;
* O multitudine de instumente care te pot ajuta sa iti dezvolti, constuiesti si testezi codul.

Pentru o mai buna intelegere a modului cum functioneaza Angular vom caracteriza mai departe componentele de baza ale acestuia:

* Modulele – Aplicatiile Angular sunt modulare si astfel Angular are propriul sistem de modularitate numit NgModules. NgModules reprezinta containare pentru un anumit bloc de cod dedicat unui domeniu al aplicatiei, sau a unui workflow. Fiecare aplicatie Angular are cel putin un modul, care reprezinta radacina si care este denumita AppModule, reprezentand mecanismul de bootstrap care lanseaza aplicatia.
* Componentele – reprezinta blocurile din care este compusa aplicatia. Componenta include o clasa TypeScript reprezentata de o adnotare @Component(), un template HTML si stilurile. Clasa @Component() specifica urmatoarele informatii:

1. Un selector CSS care defineste modul in care componenta este folosita in template.
2. Un template HTML care indica cum sa fie randata componenta;
3. Un set de stiluri CSS care ajuta la design-ul elementelor HTML.

* Servicii si Dependency Injection – pentru date sau logica care nu sunt impartite intre componente, se va crea o clasa service. O astfel de clasa este imediat precedata de adnotarea @Injectable(). Aceasta adnotare ofera metadatele care permit altor furnizori sa fie injectati ca si dependinte in clasa ta. Cu ajutorul Dependency Injection componentele vor fi mult mai eficiente deoarece nu vor prelua datele de pe server si nu se vor conecta direct la consola aceste sarcini se vor delaga catre servicii. [[2](#Ang22)]



Figure 2. Diagrama Arhitecturala a unei Aplicati Angular [[3](#Ang221)]

3.1.2 Bootstrap

Bootstrap este un framework open source de frontend, care contine elemenete de baza pentru dezvoltarea aplicatiilor web receptive. Acest framework este format pe baza HTML, CSS si JavaScript.

Cu ajutorul Bootstrap se poate detecta dimensiunea ecranului pe care o foloseste utilizatorul, adaptand dimensiunile componentelor aplicatiei web automat. Pe langa aceasta functionalitate de baza, printre componentele incluse in interfetele Bootstrep se regasesc barile de navigatie, carusele de imagini si butoane.

Printre avantajele utilizarii acestui framework se regasesc:

* Fiind responsive, ajuta la performantele aplicatiei web deoarece tot procesul de optimizare este cu mult ameliorat.
* Compatibilitatea cu diferite browsere web;
* Fiind un framework foarte popular documentatia este bogata, iar datorita acestui lucru se pot gasi foarte usor exemple de cod care trebuie doar adaptate la necesitatile fiecaruia.

Exista si mici dezavantaje in folosirea Bootstrap cum ar fi:

* Necesitatea ca browserele sa fie updatate, deoarece daca se foloseste o versiune noua a frameworkului, iar versiunea browserului este mai veche vor exista discrepante.
* Daca se va utiliza un numar mare de functii va rezulta la dimensiuni ale fisierelor foarte mari ceea ce poate duce la ingreunarea proiectului. [[4](#Jor22)]

3.2 Backend

4. Concluzii și implementări viitoare

# Bibliografie

x

|  |  |
| --- | --- |
| [1] | Visual Studio Code. (2022, Aug.) Getting Started. [Online]. <https://code.visualstudio.com/docs> |
| [2] | Angular. (2022, Aug.) Introduction to Angular concepts. [Online]. <https://angular.io/guide/architecture> |
| [3] | Angular. (2022, Aug.) Diagrama Arhitectura Angular. [Online]. <https://angular.io/guide/architecture> |
| [4] | Jordana A. (2022, July) What Is Bootstrap? [Online]. <https://www.hostinger.com/tutorials/what-is-bootstrap/> |

x