Nume student: Pavel Alina-Elena

Coordonator : Ș.l.dr.ing. Cristian Aflori

Raport nr. 2

Proiectarea software a aplicației

Dezvoltarea aplicației se bazează pe entitațile din cadrul unei instituții de învațământ și relațiilor dintre acestea .Prelucrarea datelor, implementarea interacțiunilor dintre entitați și vizualizarea de către utilizator într-un mod cât mai lizibil a rezultatelor s-a obtinut prin implementarea unei arhitecturi stratificate.Componentele sunt organizate în straturi orizontale, fiecare strat îndeplinind un rol si o responsabilitate specifică în cadrul aplicației.

Modelul arhitecturii stratificate consta din patru straturi standard: de prezentare, de business,de persistență și baza de date. Una dintre caracteristicile puternice ale modelului arhitectural stratificat este “separations of concerns” între componente. Componentele dintr-un strat specific se ocupă numai de logica care se referă la acel strat.

In dezvoltarea serviciilor Web care sunt axate pe resursele sistemului și pe interacțiunile dintre acestea s-a utiliza modelul arhitectural REST. Partea de logica de bussines a aplicației este dezvoltata utilizând frameworkul Spring Boot pentru a oferi o procesare și o gestionare eficienta a endpointurile REST. Prin intermediul frameworkului am folosit injecția dependințelor ceea ce determina ca aplicația creată să fie slab cuplată, astfel usurand procesul de dezvoltare, testare și eventual modificare a aplicației .

Nivel bază de date : există o singură bază de date centrală care va furniza date pentru componenta de persistare .S-a utilizat un sistem de gestionare a bazelor de date relaționale de tip MySql. Este remarcat pentru prelucrarea rapidă, fiabilitatea dovedită, ușurința și flexibilitatea de utilizare. La acest nivel, se află baza de date împreună cu limbajul sau de procesare a interogărilor. De asemenea, avem relațiile care definesc entitatile din cadrul unei institutii de invatamant (eg. student, profesor, disciplina, laborator, catalog, prezenta,evaluare,programa scolara, situatie scolara,restanta, evenimente), relațiilor dintre acestea și constrângerile lor la acest nivel.

Nivel de persistare : este responsabil de a facilita accesul catre operatiile asupra datelor stocate in baza de date relationala. In cadrul acestui nivel am utilizat mecanisme de object-relational mapping care implementează responsabilitatea mapării obiectului la modelul relațional si design patternul , repository, componenta care încapsulează logica necesară pentru a accesa resursele de date asigurând întreținere eficienta și decuplând infrastructura utilizată pentru a accesa baza de date de la nivelul modelului de domeniu.

Nivel de bussines: raspunzator pentru executarea logicii de bussines pentru a răspunde tuturor evenimentelor utilizatorului si este responsabil de logica aplicatiei care se referă la extragerea, procesarea, transformarea , gestionarea , asigurarea coerenței și validității datelor. Acest lucru împiedică nivelul de prezentare să execute o logică falsă care ar duce la cereri incorecte de modificare a datelor. Serviciile Web vor expune resursele sistemului clientului prin maparea acestora prin scheme URL/URI. Toate apelurile API de către sisteme externe vor fi, de asemenea, gestionate central de catre acest nivel.

Nivel de prezentare: stabilește modul în care sunt prezentate informațiile utilizatorului final fara a fi responsabil de procesarile de pe nivelele anterioare. Am dezvoltat acest nivel utilizand frameworkul de TypeScript, Angular ,pentru a crea pagini web dinamice prim implementarea modelul Model-View-Controller, un model de proiectare software de arhitectură pentru implementarea interfeței cu utilizatorul (UI) a unei aplicații. Pentru a prelua sau modifica datele, nivelul de prezentare trebuie să efectueze apeluri API către serverul dezvoltat cu ajutorul frameworkului Spring Boot.

In cadrul aplicatiei , pentru implementarea functionalitatii de gestionare eficienta a prezentelor la orele de seminar/laborator de catre profesor prin trimiterea unui cod QR prin email studentilor a fost necesar implementarea unui serviciu in Spring Boot, apeland QRCode Api dezvoltat pentru Java necesitand doar includerea dependintei Maven in fisierul Pom.xml din cadrul proiectului.Endpoint-ul va genera pe baza unor parametrii de intrare (eg. numele disciplinei ,numarul laboratorului ,numele studentului) un cod ce contine un embedded link care va completa automat prezenta studentului in urma citirii acestuia de pe telefon sau utilizand o extensie Google Chrome .

De asemenea, a fost necesar implementarea unui serviciu de trimitere a emailurilor tuturor studentilor continand codul QR generat de serviciul descris anterior. Frameworkul Spring oferă abstractizare la nivel înalt, care simplifică procesul de trimitere a e-mailurilor utilizând interfata JavaMailSender. Pentru atasarea resurselor in linie, cum ar fi imaginea codului QR, in corpul mesajului a fost setat mesajului sa fie de tipul MimeMessage continand un content-id specific imaginii atasate.

Pentru afisarea resurselor de tip curs, laborator la nivelul de prezentarea a fost necesar implementarea unui serviciu de stocarea fișierelor în sistemul de fișiere si de expunere a documentelor la nivelul de bussines prin endpointuri REST.

Rezultate intermediare obtinute

Pentru serverul de backend s-a obtinut o arhitectura stratificate .Principalele endpointuri REST care expun resursele sistemului, entitațile din cadrul unei instituții de învațământ și relațiilor dintre acestea, prin schemele URL ce au fost implementate , urmand sa se adauge in functie de nevoi noi rute. De asemenea, logica pentru generarea codului QR , trimiterea e-mailurilor si stocarea resurselor de tip fisier si imagine au fost implementate si sunt expuse userului prin apeluri API. Verificarea corectitudinii raspunsurilor si statusului cererilor HTTP s-a realizat prin apelarea endpoint-urilor utilizand tool-ul de testare , Postman.

Partea de frontend disponibila utilizatorului final al aplicatiei este alcatuita din componente Angular specifice fiecarui scenariu de bussines.Fiecare componenta este alcatuita din mai multe module specifice actiunii userului (eg. file upload, modul de trimitere emailuri, vizualizare discipline in functie de programa scolara,diferite chart-uri, tabele centralizatoare). In cadrul aplicatiei se poate loga userul de tip student , care are posibilitatea de a vedea pagina de profilul, pagina de anunturi din cadrul facultatii, disciplinele pe care le-a parcurs pana acum, notele, anunturi si statusul prezentelor la fiecare disciplina, de asemena o componenta de calendar unde are setate deadline-uri pe care trebuie sa le respecte, pagina de statistici unde poate vedea evolutia lui in comparatie cu ceilalti colegi din an.

Alt user in cadrul aplicatiei este profesorul, are are posibilitatea de a vedea pagina de profilul,detalii despre continutul disciplinei , situatii centralizatoare ,posibilitatea de a uploada diferite materiale de curs, laborator ,auxiliare , componenta de calendar .

Provocari intampinate si solutii de rezolvare

Generarea datelor care urmau a fi prelucrate in cadrul aplicatiei a reprezentat o provocare datorita volumului de date persisteate. Incercarea de a modela entitațile din cadrul unei instituții de învațământ și a relațiilor dintre acestea a presupus stabilirea din faza de proiectare a unor dimensiuni mai mici a unor componente (eg. a numarului de materii dintr-un an , a numarului de studenti intr-o grupa) pentru a avea un volum de date rezonabil de inserat dar si in acelasi timp datele sa fie cat mai veridice . Pentru generarea intregului volum de datelor care urmat a fi inserat in baza de date relationala fost necesar realizarea unui program scris in limbajul Java pentru a realiza diferite prelucrari (eg. user-ul ce contine un camp de tip username necesar etapei de logare este identic cu emailul institutional care este de forma prenume.nume@student/profesor.tuiasi.ro in functie de tipul utilizatorului ,student sau profesor) sau inserarea multipla de date unde doar cativa parametrii difera(eg. pentru acelasi student a fost necesar de introdus in tabela ce reprezinta situatia scolara diferite note pentru materiile din programa scoalara).

Ideea initiala pentru procesul de monitorizare a prezențelor prin generarea de către cadrul didactic a unui cod QR pe care studenții îl vor putea scana de pe laptop, telefon, tableta a fost folosirea unei aplicatii care ofera servicii personalizate de generare a unor coduri de acest tip in functie de diferiti parametrii . In urma documentarii si incercarii de a gasi un API care sa satisface nevoile aplicatiei ,am constat ca ar trebui platit un abonament pentru a beneficia de aceste servicii sau existenta unor endpointuri care aveau nevoie de mai multe informatii decat cele pe care le putea furniza aplicatia in curs de dezvoltare.Astfel dupa o perioada de documentare am aflat ca exista o biblioteca pentru Spring Boot care necesita doar includerea dependintei Maven in fisierul Pom.xml al proiectului.

Astfel am creat un endpoint REST pe care utilizatorul apeland-ul cu minimul de informatii (eg.numele disciplinei, numele studentului , numarul laboratorului si ziua

curenta) se genereaza o imagine ce contine codul QR cu extensia .png ce se poate scana cu ajutorul mobilului sau al unei extensii Chrome.

Initial ,pentru functia de trimitere a e-mailurilor studentilor am vrut sa dezvolt serviciul utilizand un server de Node.js dar la final am optat pentru utilizarea interfetei JavaMailSender a frameworkul Spring ce oferă abstractizare la nivel înalt, care simplifică procesul de trimitere a e-mailurilor . Pentru atasarea resurselor in linie, cum ar fi imaginea codului QR, in corpul mesajului a trebuit setat continutul mesajului sa fie de tipul MimeMessage continand un content-id specific imaginii atasate. Set up-ul pentru aceasta functionalitate a fost o provocare din cauza dependintei Maven care trebuia adaugata in Pom.xml al aplicatiei dar in urma lifecycle-ului Mave de compilare , dependinta nu era scanata neputand astfel fi recunoscuta biblioteca in proiect. A fost necesar ca solutie finala download.ul jar-ului API-ului si incluzandul in path-ul proiectului.

Comunicarea dintre nivelul de prezentare si nivelul de bussines a presupus provocari. Afisarea dinamica a datelor in functie de tipul utilizatorului si de actiunile pe care acesta le executa a presupus documentare si intelegere a modului in care frameworkul Angular, ce implementeaza design patternul MVC si utilizeaza TypeScript ca si limbaj de programare, lucreaza cu fluxul de date. Pentru fiecare scenariu de bussines a fost creata cate o componenta in stratul de prezentarea ce apeleaza diferite componente de tip controller asemanatoare cu cele din nivelul de bussiness.