**UNIVERSITATEA BABES¸ -BOLYAI CLUJ-NAPOCA**

**FACULTATEA DE MATEMATICAˇ S¸ I INFORMATICAˇ**

**SPECIALIZAREA MATEMATICAˇ - INFORMATICAˇ**

# LUCRARE DE LICENT¸ A˘

**[Titlu lucrare]**

## Conduca˘tor s, tiint, ific

**[Grad, titlu s, i nume coordonator]**

**Lector universitar, Dr. Dragos Radu**

*Absolvent Butean Rares-Ionut*

2021

ABSTRACT

Abstract: un rezumat ˆın limba engleza˘ cu prezentarea, pe scurt, a cont, inutului pe capitole, punaˆnd accent pe contribut, iile proprii s, i originalitate

Due to the recent pandemic event, nearly all schools, universities and IT related businesses migrated to an online, work from home environment. This abrupt and unanticipated change required a rapid mobilization, which led to an adoption of a multitude of applications to serve the same purpose – a competent procedure of communication and work tracking.

[Application name / This solution /...?] manages to bring the forementioned(?) [proposed / suggested / presented] demands to a single web application in an easy-to-use manner. Supporting communication via text, audio and video, departments, announcements, calendar, and assignments, [Application name / This solution /...?] creates an apt working environment in both education and business.

Keywords: Enterprise Resource Planning, WebAPI, intranet.

[Main chapters summary]

# Cuprins

1. [Introducere](#_bookmark0) 1
2. [Titlul capitolului](#_bookmark1) 2
3. [Titlul capitolului](#_bookmark5) 5

[3.1 Sectiune](#_bookmark6) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

[3.1.1 Subsectiune](#_bookmark7) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

[3.2 Sectiune](#_bookmark8) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6

[3.2.1 Subsectiune](#_bookmark9) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6

1. [Titlul capitolului](#_bookmark10) 7
2. [Concluzii](#_bookmark11) 9

[Bibliografie](#_bookmark12) 10

**Capitolul 1 Introducere**

Introducere: obiectivele lucra˘rii s, i descrierea succinta˘ a capitolelor, prezentarea te- mei, prezentarea contribut, iei proprii, respectiv a rezultatelor originale s, i ment, ionarea (daca˘ este cazul) a sesiunii de comunica˘ri unde a fost prezentata˘ sau a revistei unde a fost publicata˘.

Ca rezultat al contextului pandemic actual, procesul de digitalizare a fost grabit semnificativ. Acest fapt a pus institutiile din domeniul educatiei si al businessului in ipostaza de a folosi numeroase aplicatii deja existente, al caror ansamblu serveste la crearea unui spatiu de munca si unei colaborari eficiente.

[Vb despre google, google scholar, zoom, teams, webspace-urile prezente, care probabil nu erau pregatite pentru context full online, fapt care a dus la folosirea mai multor aplicatii, functionalitati necesare (share screen, prezenta participant, etc.) si aparitia unor probleme: folosirea mai multor aplicatii simultan, stocarea de informatii impartita in mai multe aplicatii]

[Application name / This solution /...?] contribuie procesului de digitalizare prin integrarea unui sistem de comunicare intr-un mediu ce ofera o buna colaborare si sincronizare membrilor unei institutii de invatamant sau ale unui business.

[Application name / This solution /...?] este un website ce implementeaza comunicarea prin text si apeluri audio-video, programarea sedintelor, calendar cu evenimente, crearea anunturilor de interes general sau tintite spre un department. Adaptabilitatea [Application name / This solution /...?] la un mediu de business sau unuia educativ reiese din implementarea unui sistem de monitorizare a orelor de lucru, respectiv a unuia de atribuire a temelor.

Motivatia dezvoltarii [Application name / This solution /...?] vine din redundanta creata in urma utilizarii mai multor aplicatii cu scopul de a stabili o modalitate de colaborare eficienta in mediul online.

# Capitolul 2

## Tehnologii folosite

[Intro in tipul aplicatiei / Comparatie cu exemple de aplicatii asemanatoare / …?]

[Application name / This solution /...?] este o aplicatie web construita dupa modelul client-server, in care componenta server furnizeaza fiecarui client conectat la aplicatie un serviciu web sau o resursa.

Cele doua componente comunica prin intermediul tiparului de mesagerie cerere-raspuns (request-response). Clientul trimite o cerere, iar in urma procesarii si validarii acesteia conform regulilor bine stabilite, primeste un raspuns de la server.

Pentru a formaliza schimbul de date, componenta server a [Application name / This solution /...?] expune un web API (Application Programming Interface) – un tip de interfata software ce faciliteaza schimbul de informatii prin apelarea metodelor numite endpoint-uri.

Ca beneficiu secundar, un web API ascunde informatiile interne cu privire la modul de functionare a serverului prin expunerea publica exclusiv a partilor ce pot fi utilizate in dezvoltarea frontend-ului aplicatiei. Mai mult decat atat, web API-ul folosit in [Application name / This solution /...?] este supus filtrului CORS (Cross Origin Resource Sharing), fapt ce blocheaza orice alta aplicatie neautorizata sa utilizeze endpoint-urile construite.

La bazele dezvoltarii [Application name / This solution /...?] se afla urmatoarele tehnologii si framework-uri:

* Java
* REST API
* Spring Framework
* MySQL
* Angular
* HTML
* CSS
* TypeScript
* OAuth 2.0
* Git
* (?) WebRTC
* (?) Server VPN

### 2.1 Java

Dezvoltat de catre James Gosling in cadrul Sun Microsystems (in present detinut de Oracle) si lansat in anul 1995, Java este un limbaj de programare de nivel inalt, orientat pe obiecte, al carui design ofera o cuplare scazuta.

Unul dintre cele mai mari avantaje ale limbajului este executarea aplicatiilor compilate in Java virtual machine (JVM) ce permite rularea pe orice computer ce suporta Java, indiferent de arhitectura acestuia. Acest lucru este posibil datorita compilarii codului sursa intr-o reprezentare intermediara numita Java bytecode.

Prin acest beneficiu si prin constructia limbajului, Java nu reuseste sa impresioneze prin performantele obtinute in categoriile viteza de rulare si pornire, scalabilitate in sisteme multi-core, utilizarea memoriei si eficienta functiilor trigonometrice, in special comparate cu un limbaj de viteza mare, precum C/C++.

Cu toate acestea, conform GitHub, Java este al doilea cel mai utilizat limbaj de programare la anul scrierii, in special utilizat in dezvoltarea aplicatiilor web de tip client-server.

In astfel de aplicatii, viteza nu este un factor ce determina alegerea limbajului ales. Convenabilitatea si abordabilitatea utilizarii acestui limbaj, alaturi de abundenta documentatiei aflata in multiple resurse, atat digitale cat si fizice, constituie cauza alegerii acestui limbaj de programare in dezvoltarea componentei de business logic a aplicatiei prezentate.

Versiunea folosita la dezvoltarea [Application name / This solution /...?] este Java 17, versiunea actuala cu suport pe termen lung, lansata in Octombrie 2021.

*CAPITOLUL 2. TITLUL CAPITOLULUI*

### 2.2 REST API

Representational state tranfer, abreviat REST, este un stil arhitectural software creat de catre Roy Fielding in anul 2000, menit sa ghideze si formeze dezvoltarea arhitecturii World Wide Web-ului.

REST impune un set de constrangeri arhitecturale, care, aplicate intr-un intreg, accentueaza scalabilitatea interactiunii intre componente, generalitatea interfetelor, independenta implementarii componentelor, marirea securitatii, si incapsularea sistemelor legacy.

Constrangerile REST sunt:

1. Arhitectura Client-Server

- Principiul separarii problemelor reprezinta fundatia arhitecturii Client-Server. Impartirea interfetei utilizatorului de stocarea datelor imbunatateste portabilitatea pe mai multe platforme. Mai mult decat atat, componentele pot fi dezvoltate independent unele de altele.

Shape, rectangle

Description automatically generated

Figura 2.2.1. Client-Server

1. Stateless

- Comunicarea rezultanta din interactiunea client-server trebuie sa fie de natura stateless, astfel incat fiecare solicitare emisa de client catre server este nevoita sa contina toata informatia necesara intelegerii acesteia, fara a se avantaja de context. Prin urmare, stadiul sesiunii este in intregime in posesia clientului. Aceasta constrangere ofera proprietatile de vizibilitate, fiabilitate si scalabilitate.

Shape

Description automatically generated

Figura 2.2.2. Client-Stateless-Server (?)

1. Cache

- Pentru a imbunatati eficienta retelei, constrangerile cache adaugate formeaza

client-cache-stateless-server. Acestea impun informatiei unui raspuns sa fie implicit sau explicit ca si stocabil in cache sau nu. In caz afirmativ, un cache al clientului primeste dreptul de a refolosi informatia acelui raspuns mai tarziu, pentru cereri echivalente. Constrangerile cache maresc eficienta, scalabilitatea si performanta vazuta de utilizator prin reducerea timpului de raspuns.

Shape

Description automatically generated

Figura 2.2.3. Client-Cache-Stateless-Server (?)

1. Interfata uniforma

- Prin aplicarea principiului de generalitate definit in ingineria software, arhitectura sistemului este simplificata si vizibilitatea interactiunilor marita. Implementarile sunt decuplate de serviciile pe care le produc, ceea ce incurajeaza evolutia independenta. In schimb, aceasta constrangere degradeaza eficienta, deoarece informatia este transmisa intr-o forma standardizata in locul uneia specifice nevoilor aplicatiei. Interfata REST este optimizata pentru cazurile obisnuite de utilizare a Web-ului. Pentru a obtine o interfata uniforma, mai multe constrangeri arhitecturale trebuie aplicate. REST este definit de patru astfel de constrangeri: identificarea resurselor, manipularea resurselor prin reprezentari, mesaje auto-descriptive si favorizarea folisirii datelor hypermedia, din cauza avantajelor intrinseci ale arhitecturii pe acest tip de date.

Diagram

Description automatically generated

Figura 2.2.4. Uniform-Client-Cache-Stateless-Server (?)

1. Stratificarea sistemului

- Aceasta constrangere permite arhitecturii sa fie compusa din straturi ierarhice prin restrangerea comportamentului in asa fel incat niciun component nu poate interactiona cu alte componente in afara celor vecine. Prin restrictionarea cunostintelor sistemului la un singur apare limitarea complexitatii generale a sistemului si promovarea independentei substraturilor. Astfel se faciliteaza inlocuirea sau extinderea straturilor, care pot fi folosite si pentru incapsularea serviciilor legacy si protejarea noilor servicii de clientii legacy

Diagram

Description automatically generated

Figura 2.2.5. Uniform-Layered-Client-Cache-Stateless-Server

1. Code-On-Demand

- Prin aceasta constrangere, REST ofera clientului extinderea functionalitatii prin descarcarea si executarea codului sub forma de applet-uri sau script-uri. Datorita acestui fapt se reduce numarul de functionalitati ce trebuie pre-implementate, iar sistemul devine mai extensibil. Prin urmare e optionala. Code-on-demand este adesea folosit in arhitecturile in care sistemele includ multiple limitari organizationale.

Diagram

Description automatically generated

Figura 2.2.6. Arhitectura REST

*CAPITOLUL 2. TITLUL CAPITOLULUI*

### 2.3 Spring Framework

Spring Framework este un framework si inversion of control container (?) (IoC container) open source pentru platforma Java. Caracteristicile principale ale framework-ului pot fi folosite de orice aplicatie Java. Desi Spring Framework nu impune un model de programare specific, cel mai utilizat este modelul Enterprise JavaBeans (EJB).

Versiunea folosita in [Application name / This solution /...?] este Spring 5.3.8, aceasta fiind ultima versiune stabila.

Fraza introducere in module [TODO]

### 2.3.1 Spring Boot

Extensia Spring Boot este solutia „convention-over-configuration” pentru crearea aplicatiilor bazate pe Spring, stand-alone, la nivel de productie, preconfigurate.

Printre caracteristicile de baza ale acesteia se afla:

* Incorporarea directa a Tomcat sau Jetty, nu necesita lansarea de fisiere WAR
* Furnizarea de Project Object Models (PMOs) pentru simplificarea configuratiei Maven sau Gradle
* Configurarea automata
* Furnizarea de caracteristici production-ready cum ar fi metrici, health checks[[1]](#footnote-1) si configuratie externalizata
* Nici o necessitate pentru configuratia XML

Versiunea implementata in [Application name / This solution /...?] este versiunea 2.5.4.

### 2.3.1.1 Spring Security

### Modulul Security este un framework ce furnizeaza procesele de autentificare si autorizare, precum si alte procese de Securitate pentru aplicatiile enterprise.

### Proiectul a fost inceput in 2003 de catre Ben Alex sub numele de “Acegi Security”. Acegi a fost incorporat in portofoliul Spring ca si Spring Security, avand primul release public in anul 2008.

### Flow-ul procesului de autentificare este:

|  |  |
| --- | --- |
| Browser-ul trimite credentialele autentificarii | |
| „*Authentication Mechanism*” preia informatiile | |
|  | Un obiect „*authentication request*” este construit |
|  | Obiectul „*authentication request*” este trimis catre „*AuthenticationManager*” |
|  | *„AuthenticationManager” este responsabil pentru trimiterea request-urilor printr-un lant de „AuthenticationProviders”* |
|  | *„AuthenticationProvider*" va cere un „*UserDetailsService*” pentru a furniza un obiect „*UserDetails*” |
|  | Obiectul „*UserDetails*” va fi folosit in constructia obiectului „*Authentication*” complet populat |
| Daca „*Authentication Mechanism*” primeste inapoi obiectul „*Authentication*” atunci va considera request-ul valid, va plasa „*Authentication*” in „*SecurityContextHolder*” si va cauza request-ul original sa fie reincercat  Daca, pe de alta parte, „*AuthenticationProvider*” respinge request-ul, atunci „*Authentication Mechanism*” va cere utilizatorului sa reincerce. | |
| *„AbstractSecurityInterceptor” a*utorizeaza request-ul recreat si arunca o exceptie Java | |
| *„ExceptionTranslationFilter”* traduce exceptia aruncata de *„AbstractSecurityInterceptor” in coduri de eroare specifice HTTP* | |
|  | Cod de eroare 403 – daca utilizatorul a fost logat insa ii lipeste autorizarea accesului  Trimite un „*AuthenticationEntryPoint*” daca utilizatorul nu a fost logat |

### Mecanismul de baza a modulului Spring Security din interiorul [Application name / This solution /...?] a fost augmentat de OAuth 2.0 (prezentat in subcapitolul 2.X [TODO]).

### 2.3.1.2 Spring Data

Spring Data aduce un model de programare consistent pentru accesul de date in timp ce conserva caracteristicile specifice bazei de date subiacente.

Acesta faciliteaza folosirea tehnologiilor de access de date, bazelor de date relationale si non-relationale si serviciilor de date.

Caracteristicile principale ale modulului sunt:

* Crearea de obiecte tip Repository
* Abstractizarea procesului de object mapping
* Implementarea claselor domain de baza cu proprietati fundamentale
* Suport pentru audit transparent
* Integrarea codului repository definit de programator

Modulul adaposteste numeroase subproiecte implementate in [Application name / This solution /...?], precum Spring Data JDBC sau Spring Data JPA.

# Capitolul 3 Titlul capitolului

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nos- trud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

## Sectiune

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nos- trud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

### Subsectiune

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nos- trud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

*CAPITOLUL 3. TITLUL CAPITOLULUI*

## Sectiune

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nos- trud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

### Subsectiune

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nos- trud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

# Capitolul 4 Titlul capitolului

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nos- trud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tem- por incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tem- por incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tem- por incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tem- por incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla

*CAPITOLUL 4. TITLUL CAPITOLULUI*

pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tem- por incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tem- por incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tem- por incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

# Capitolul 5 Concluzii

Concluzii ...

# Bibliografie

[Har93] Peter Harwood. The title of the work. Master’s thesis, The school of the thesis, The address of the publisher, 7 1993. An optional note.

//

https://en.wikipedia.org/wiki/API

https://en.wikipedia.org/wiki/Client–server\_model

<https://githut.info/>

Fielding, Roy Thomas. *Architectural Styles and the Design of Network-based Software Architectures*. Doctoral dissertation, University of California, Irvine, 2000.

//

[Jos93] Peter Joslin. *The title of the work*. PhD thesis, The school of the thesis, The address of the publisher, 7 1993. An optional note.

[Min] Ministry for Primary Industries. Kina sea urchin regions in NZ. [http:](http://fs.fish.govt.nz/Page.aspx?pk=7&sc=SUR)

[//fs.fish.govt.nz/Page.aspx?pk=7&sc=SUR](http://fs.fish.govt.nz/Page.aspx?pk=7&sc=SUR). Online; accessed 29 January 2014.

[RB12] Narayan Ramasubbu and Rajesh Krishna Balan. Overcoming the challen- ges in cost estimation for distributed software projects. In *Proceedings of the 34th International Conference on Software Engineering*, ICSE ’12, page 91–101.

IEEE Press, 2012.

[RRT15] Romain Pierre Julien Robbes, D Ro¨ thlisberger, and E´

Tanter. Object-

oriented software extensions in practice. *Empirical Software Engineering*, 20(3), 2015.

[Som10] Ian Sommerville. *Software Engineering*. Addison-Wesley Publishing Com- pany, USA, 9th edition, 2010.

1. evaluare a aplicatiei in conformitate cu obiectivele sale si aderenta la metodologia propusa [↑](#footnote-ref-1)