România
Ministerul Apărării Naționale
Academia Tehnică Militară *"Ferdinand I"*

Facultatea de Sisteme Informatice și Securitate Cibernetică
Specializarea: Calculatoare și sisteme informatice pentru apărare și securitate națională

PLATFORMĂ PENTRU COACHING ONLINE

Coordonator Științific
Cpt. lect. dr. ing. Iulian Aciobăniței

Absolvent
Sd. sg. maj, Miron-Onciul Tudor

Conține _____ file
Inventariat sub numărul _____
Cu poziția din indicator _____
Cu termen de păstrare _____

București
2024

Abstract

In the increasingly fast paced rhythm of today's society, the world is undergoing continuous digitalization, which leads to a tendency of a sedentary lifestyle and a lack of physical activity. An effective solution to counteract this trend is to use technology to promote sports in the online world. Adopting a healthy and active lifestyle is the main pillar in improving overall well-being, preventing various health problems.

This paper presents the FitStack application, which is an online platform, designed to create a strong link between coaches and their clients. Users have the option to consume sports content such as exercises, workouts or fitness programs. Additionally, the concept of gamification provides to users a pleasant experienceon the platform, facilitating the accelerated progress through claiming prizes and accumulating points that can secure an advanced position in the global leaderboard. Users can opt for a premium account which grants access to sports content provided by the coaches and facilitates communication with them through the chat service.

The online coaching application is available on iOS and Android mobile platforms, due to using Flutter framework for implementation. The user interface is intuitive, adhering to the best practices of UI/UX concepts. FitStack is designed to host a significant number of users due to its microservices-based architecture, which provides scalability, performance, efficiency, and fault tolerance.

The application provides a secure payment module of subscriptions integrated with the PayPal platform. Additionally, the application uses HTTPS to enssure encryption and confidentiality of data during server-client communication. The platform uses the Kubernetes' provided module, Cert-manager to automate the issuance and management of certificates verified by recognised certification authorities.

Rezumat

În ritmul tot mai alert al societății actuale, lumea se află într-o continuă digitalizare, ceea ce duce la o tendință crescută din partea oamenilor la sedentarism și la un stil de viață lipsit de activitate fizică. O soluție eficientă pentru contracararea acestei tendințe este folosirea tehnologiei pentru promovarea sportului în mediul online. Adoptarea unui stil de viață sănătos și activ reprezintă pilonul principal în îmbunătățirea bunăstării generale, prevenind diverse afecțiuni și probleme de sănătate majore.

Această lucrare prezintă aplicația FitStack, o platformă de coaching online, menită să creeze o legătură puternică între antrenori și clienții lor. Utilizatorii au posibilitatea de a consulta conținut sportiv precum exerciții, antrenamente și programe de fitness. De asemenea, conceptul de gamificare oferă utilizatorilor o experiență plăcută în cadrul platformei, facilitând progresul accelerat prin câștigarea premiilor și acumularea de puncte care pot asigura o poziție avansată în clasamentul general. Utilizatorii au posibilitatea de a opta pentru un cont premium, care le permite accesul la conținutul sportiv adăugat de antrenori și facilitează comunicarea cu aceștia prin serviciul de mesagerie.

Aplicația de coaching online este disponibilă pe platformele mobile Android și iOS datorită folosirii framework-ului Flutter în implementare. Interfața este una intuitivă, respectând bunele practici ale conceptelor de UI/UX. FitStack este proiectată pentru a fi disponibilă pentru un număr considerabil de utilizatori, datorită arhitecturii sale bazate pe microservicii, care oferă scalabilitate, performanță, eficiență și toleranță la defecte.

Aplicația oferă un modul securizat de plată al abonamentelor prin integrarea cu platforma PayPal. De asemenea, aplicația utilizează HTTPS pentru a asigura criptarea și confidențialitatea datelor în procesul de comunicare între client și server. Platforma utilizează modulul furnizat de Kubernetes, Cert-manager, pentru automatizarea emiterii și gestionării certificatelor verificate de autorități de certificare recunoscute.

Cuprins

Listă de Imagini

Listă de Abrevieri

JWT ..... Json Web Token

JSON ..... JavaScript Object Notation

OAuth ..... Open Authorization

CRUD ..... Create, Read, Update, Delete

ACID ..... Atomicity, Consistency, Isolation, Durability

HTTP ..... Hypertext Transfer Protocol

HTTPS ..... Hyper Text Transfer Protocol Secure

TLS ..... Transport Layer Security

SSL ..... Secure Sockets Layer

UI/UX ..... User Interface/User eXperience

REST ..... Representational State Transfer

API ..... Application Programming Interface

UML ..... Unified Modeling Language

JPA ..... Java Persistence API

DOM ..... Document Object Model

1 Introducere

1.1 Importanța temei

În contextul tehnologic actual, fitnessul și activitățile sportive tind să fie omise de către majoritatea oamenilor, în ciuda faptului că nu îmbunătățesc doar aspectul fizic, ci au și un impact major asupra sănătății mintale. Sportul este esențial în viața oricărui om, având efecte pozitive asupra încrederii de sine, reducerii stresului, îmbunătățirii somnului și a activității cerebrale. Acestea sunt doar câteva dintre beneficiile sportului asupra corpului uman, acționând ca un mecanism de prevenție a multor probleme de sănătate, precum obezitatea, problemele cardiace, tensiunea arterială crescută etc.

Dezvoltarea lumii digitale nu aduce doar dezavantaje în domeniul sportului, deoarece aplicațiile cu tematică fitness sunt din ce în ce mai utilizate. Avansul tehnologic permite crearea aplicațiilor de calitate dedicate sportului, acestea fiind capabile să deservească cât mai mulți utilizatori în mod eficient și performant, toate acestea prin intermediul unor interfețe clare, fluide și plăcute din punct de vedere estetic.

În momentul de față există o gamă vastă de aplicații de fitness, fiind foarte grea alegerea celei mai bune pentru scopurile pe care le are clientul. Există o multitudine de tipologii de utilizatori, de la avansați, la oameni care își doresc ceva specific precum alergare, yoga, exerciții de mobilitate etc., până la cei care sunt complet neinițiați în lumea sportului. O aplicație de fitness ideală trebuie să satisfacă cerințele oricărui utilizator, indiferent de locul în care se află în călătoria lor sportivă. O astfel de aplicație este foarte greu de implementat, deoarece nu sunt ușor de prezis așteptările fiecărui utilizator de la aplicație.

O abordare este și crearea unei aplicații dedicate unei categorii specifice de oameni, care doresc să evolueze pe un singur plan, precum powerlifting, alergare, antrenamente de forță, exerciții cu greutatea corporală etc. Un dezvoltator al acestor aplicații trebuie să țină cont și de faptul că unii utilizatori nu beneficiază de echipamentul necesar găsit în mod obișnuit într-o sală de fitness, fiind necesară adăugarea unor exerciții și antrenamente care se pot executa doar cu greutatea corporală sau cu echipament minim, cum ar fi o bară, un covor de yoga, un prosop etc.

Aplicațiile de fitness, deși trebuie să țină cont de toate aceste aspecte menționate anterior, ele trebuie să aibă un singur scop final, și anume, progresul utilizatorilor. Deoarece unii clienți sunt mai puțin motivați, o funcționalitate importantă o reprezintă adăugarea unei metode de responsabilizare a utilizatorilor prin notificări cu mesaje motivaționale sau de atenționare, sau prin implicarea antrenorilor în aplicație.

1.2 Obiectivele lucrării

Platforma de coaching online FitStack este creată pentru întrunirea intereselor atât ale clienților care necesită metode eficiente, diverse și bine definite pentru a se antrena, cât și ale antrenorilor care au nevoie de modalități simple și directe de adăugare a conținutului sportiv. Această aplicație îndeplinește nevoile acestor două categorii de utilizatori, într-un cadru rapid, eficient, securizat și scalabil,platforma fiind capabilă să gestioneze un număr considerabil de clienți.

Pe piață există mai multe aplicații care oferă posibilitatea realizării antrenamentelor și exercițiilor fizice. Platforma de coaching online creează mult mai mult decât atât, facilitând stabilirea unei conexiuni puternice între două categorii de utilizatori care se află într-o relație de producător și consumator, respectiv antrenor și client. De asemenea, fiecare activitate sportivă este monitorizată în cadrul aplicației prin intermediul istoricului și statisticilor generate pe baza acestuia.

Aplicația este dezvoltată în jurul conceptului de gamificare, însemnând împrumutarea unor concepte întâlnite în mod uzual în jocurile video. Aceste mecanisme vor fi folosite în alte contexte, cu scopul de a crea o experiență mai plăcută, oferind utilizatorilor premii și recompense în funcție de performanța acestora.

Aplicația este disponibilă atât pentru sistemele de operare Android și iOS, cât și pentru majoritatea browserelor de pe piață. Prezența pe multiple platforme are ca scop creșterea popularității și atragerea unei game extinse de potențiali utilizatori. Interfața aplicației este interactivă, simplă și intuitivă, ceea ce o face ușor de folosit de către clienți.

Platforma FitStack își propune să ofere servicii eficiente, securizate și tolerante la defecte, prin proiectarea aplicației în jurul șablonului arhitectural de tip microservicii. Acest lucru face ca aplicația să fie mai scalabilă și mai capabilă să deservească un număr considerabil de clienți.

1.3 Metodologia de cercetare

Metodologia de cercetare pentru aplicația de coaching online FitStack a

constat în crearea unui model bazat pe șablonul arhitectural de tip microservicii. Acest obiectiv a dus la proiectarea unui sistem complex care a fost conceput să fie scalabil, eficient, performant și să poată deservi un număr ridicat de utilizatori. Proiectarea aplicației s-a realizat prin implementarea incrementală a fiecărei sarcini stabilite la începutul dezvoltării proiectului, astfel având la fiecare iterație o versiune funcțională dar incompletă a aplicației.

Fiecare iterație a fost urmată de o etapă de testare riguroasă în care au fost verificate noile funcționalități, atât în mod izolat, cât și în colaborare cu restul componentelor deja existente în sistem. Aplicația se bazează pe prioritizarea testării celor mai de sus componente din ierarhie, deoarece se dorește un sistem funcțional și verificat încă de la începutul dezvoltării, în care modulele sunt interoperabile.

Pe parcursul implementării sarcinilor aplicației au existat probleme atât din perspectiva arhitecturii complexe a sistemului, cât și din lipsa de experiență în tehnologiile și framework-urile alese pentru dezvoltarea aplicației. Aceste probleme au fost depășite prin intermediul surselor publice și a documentațiilor propuse.

Sarcinile principale din cadrul aplicației de coaching online Fitstack au fost testate cu ajutorul unui eșantion alcătuit din persoane apropiate și prieteni, de la care am obținut un feedback legat de interfața utilizatorului, astfel îmbunătățind și eliminând problemele majore care au fost omise în procesul de dezvoltare al platformei. Aceștia au menționat că platforma are funcționalități interesante și cu adevărat utile în procesul de realizare a antrenamentelor.

1.4 Rezultatele obținute

Implementarea platformei de coaching online FitStack are ca scop final crearea unui produs software de calitate, testat, capabil să gestioneze un număr considerabil de utilizatori. Aplicația este multi-platformă, disponibilă pentru platformele Android și iOS, având o interfață intuitivă, simplistă și fluidă. Tehnologiile utilizate în componentele de backend, frontend și în procesul de deploy fac ca platforma de coaching să fie robustă din toate punctele de vedere.

Funcționalitățile produsului software final sunt:

- Autentificare și înregistrare în aplicație cu cont normal sau prin intermediul platformei Google
- Resetare parolă
- Executarea antrenamentului
- Executare program de fitness

- Crearea propriului antrenament
- Filtrarea antrenamentelor și programelor de fitness în funcție de cuvinte cheie și de nivelul de dificultate
- Posibilitatea de a câștiga premii și de a obține puncte în funcție de performanța utilizatorului pe parcursul antrenamentului
- Vizualizare statistici despre antrenamente
- Vizualizare clasament atât global cât și clasament format din contactele Google
- Posibilitatea de a trece la cont premium prin plata online cu PayPal
- Urmărire antrenor
- Adăugare conținut sportiv (exerciții, antrenamente, programe de fitness)
- Vizualizare profil
- Adăugare și verificare de certificări sportive
- Posibilitatea schimbului de mesaje între antrenori și urmăritorii acestora

1.5 Rezumatul lucrării pe capitole

Capitolele din cadrul lucrării oferă o analiză detaliată a procesului de dezvoltare a platformei de coaching online, printr-o structură clar definită.

Primul capitol reprezintă introducerea în contextul lucrării, unde este prezentată tema și motivația alegerii acesteia, continuând cu metodele folosite în procesul dezvoltării metodologiei de cercetare. De asemenea, sunt menționate și rezultatele obținute în urma implementării aplicației.

Cel de-al doilea capitol conține informații relevante cu privire la tehnologiile utilizate în contextul dezvoltării platformei. Aici, conținutul este împărțit în trei componente, fiind menționate tehnologiile folosite pentru dezvoltare, monitorizare și deploy a aplicației.

Al treilea capitol conține un studiu în care sunt detaliate atât avantajele, cât și dezavantajele celor mai populare aplicații din domeniul fitness. În primul subcapitol sunt enumerate și descrise în detaliu un set de aplicații cunoscute de pe piață, urmat de o analiză a detaliilor financiare. Ultimul subcapitol constă în compararea aplicațiilor din mai multe considerente.

Penultimul capitol este format din descrierea implementării aplicației de coaching online, prin definirea cerințelor atât funcționale, cât și nefuncționale,

urmată de prezentarea arhitecturii soluției. În continuare, sunt detaliate funcționalitățile modulelor componente și tehnologiile utilizate în implementarea acestora. Acest capitol se încheie prin enumerarea componentelor principale de securitate, urmată de analiza rezultatelor testelor efectuate și interpretarea acestora.

Ultimul capitol subliniază ideile principale abordate în această lucrare, completând cu direcțiile pentru continuarea cercetării.

2 Tehnologii utilizate

2.1 Tehnologii pentru dezvoltare

Platforma de coaching online are ca scop principal furnizarea datelor într-o manieră rapidă și sigură pentru un număr mare de utilizatori. De aceea, este nevoie de o suită de *framework*-uri și instrumente robuste care să faciliteze aceste cerințe.

2.1.1 Tehnologii Backend

2.1.1.1 Spring

Baza infrastructurii pentru componenta Backend a platformei de coaching online este formată din *framework*-ul Spring și de seturile de instrumente software cu care se integrează. Spring oferă suport pentru o gamă largă de scenarii, fiind structurat în module, printre care se numără Spring Boot, Spring Cloud, Spring Security, Spring MVC, și multe alte module complementare. Această modularitate permite adaptarea la nevoile specifice ale oricărui proiect.

Spring Boot

Spring Boot[42] este un *framework open-source* ce are la bază limbajul de programare Java[1]. Spring Boot simplifică proiectarea atât a aplicațiilor ce se bazează pe microservicii, cât și a aplicațiilor web, integrându-se cu o gamă largă de extensii software.

Spring Boot aduce numeroase beneficii precum:

- Abordare opinionată: Spring Boot furnizează o configurație implicită pentru a elimina scrierea de cod de tip șablon, accelerând procesul de dezvoltare al aplicației.
- Aplicații de sine stătătoare: aplicațiile dezvoltate prin intermediul *framework*-ului Spring Boot pot rula de unele singure deoarece este furnizat un

[1] https://docs.oracle.com/en/java/

server web în mod implicit. Tomcat[2], Jetty[3] sau Undertow[4] sunt opțiunile oferite de platforma Spring Boot.

- Injectarea dependințelor: Spring Boot aduce conceptul de delegare a controlului, eliminând necesitatea de a gestiona manual crearea și configurarea obiectelor. Acest aspect aduce foarte multe beneficii, în special la partea de testare, deoarece modulele aplicației sunt decuplate, putem aplica teste în mod izolat.

- Tranziție simplificată către mediul de producție: *framework*-ul Spring Boot furnizează funcționalități implicite pentru mediul de producție: colectarea metricilor, monitorizarea ”sănătății” și externalizarea configurației.

Aceste aspecte fac ca *framework*-ul Spring Boot să fie o soluție robustă și eficientă pentru crearea unei aplicații web.

Spring Cloud

Spring Cloud[38], o componentă a platformei Spring, este utilizată pentru a construi platforma de coaching online, care folosește un pattern arhitectural de tip microservicii.

Spring Cloud este proiectată cu scopul de a simplifica procesul dezvoltării și gestionării sistemelor distribuite și complexe. Acest *framework* oferă o gamă variată de instrumente specifice dezvoltării aplicațiilor în *cloud* bazate pe microservicii, precum:

- serviciu de înregistrare și descoperire a microserviciilor: integrarea cu serviciul Netflix Eureka

- servicii de rutare: permite configurarea API Gateway-urilor pentru rutarea traficului către serviciile aferente

- încărcarea echilibrată a serviciilor

- Întreruperi de Circuit: patternul Circuit Breaker[5] este folosit pentru prevenirea rulării unei aplicații care este predisusă să eșueze, cu scopul de a nu consuma inutil resursele mediului de producție

Prin urmare, Spring Cloud oferă instrumentele necesare pentru a răspunde rapid și eficient la nevoile dinamice ale utilizatorilor.

[2] https://tomcat.apache.org/

[3] https://jetty.org/

[4] https://undertow.io/

[5] https://learn.microsoft.com/en-us/azure/architecture/patterns/circuit-breaker

Spring Security

Un element care trebuie să fie nelipsit într-o aplicație este securitatea și accesul autorizat la resursele sistemului. Spring Security, un modul robust devenit standard de facto în asigurarea securității aplicațiilor Java, furnizează mecanisme prin care se impune autentificarea și autorizarea pentru a avea acces la diferite servicii REST din cadrul aplicației. Spring Security suportă o multitudine de metode de autentificare precum: HTTP Basic[6], LDAP[7], OAuth 2.0[8], JWT[9] și multe altele.

Spring Data JPA

Spring Data JPA[39]este un instrument ce face parte din modulul Spring Data, destinat abstractizării nivelului de date prin tehnologia JPA[10]. Java Persistence API (JPA) furnizează mecanisme care implementează operațiuni CRUD asupra bazei de date, prin intermediul unor API-uri. JPA folosește patternul de repozitoriu, care presupune crearea unei interfețe care comunică cu baza de date și implementează metode la nivel centralizat.

Una dintre caracteristicile esențiale ale instrumentului JPA este funcționalitatea de a crea operații CRUD asupra bazei de date doar prin definirea semnăturii în interfața repozitoriului. JPA permite extinderea funcționalităților standard ale interfeței pentru crearea metodelor mai complexe asupra bazei de date.

JPA aduce cu sine o serie de beneficii importante:

- Reducerea codului tip șablon: JPA permite crearea metodelor de bază pentru operațiunile CRUD, doar prin generarea interogărilor pe baza convenției de nume.
- Ușor de învățat: Abordarea intuitivă face ca tehnologia JPA, împreună cu *framework*-ul Spring, să fie o opțiune populară pentru dezvoltatori.
- Flexibilitate ridicată: JPA permite atât interogări simple, cât și complexe.

2.1.1.2 Node.js

Node.js[26] este un mediu de runtime bazat pe limbajul de programare JavaScript[11]. Node.js folosește V8 Engine, care este nucleul browserului Google Chrome, permițând acestei tehnologii să fie foarte performantă.

[6] https://en.wikipedia.org/wiki/Basic_access_authentication
[7] https://en.wikipedia.org/wiki/Lightweight_Directory_Access_Protocol
[8] https://oauth.net/2/
[9] https://jwt.io/
[10] https://www.ibm.com/docs/en/was-liberty/nd?topic=liberty-java-persistence-api-jpa
[11] https://developer.mozilla.org/en-US/docs/Web/JavaScript

Paradigma non-blocantă pe care se bazează Node.js aduce numeroase beneficii platformei de coaching online, răspunzând rapid la nevoile utilizatorilor. Motivul principal pentru care este folosit serverul de Node.js îl reprezintă numărul de conexiuni ridicate pe care acesta poate să le suporte în mod concurent, fiind o funcționalitate esențială pentru serviciul de schimb de mesaje în timp real între clienții platformei.

De asemenea, un avantaj pe care îl aduce Node.js este faptul că se bazează pe un limbaj de programare, JavaScript, care are multă notorietate pe piața IT, permițând astfel programatorilor să dezvolte nu doar aplicații frontend, ci și aplicații pentru backend.

Node.js folosește un instrument robust pentru gestionarea pachetelor software, și anume NPM[12], care este unul dintre cele mai mari registre de module software. Acesta este un instrument cu o foarte mare utilitate, întrucât doar prin simpla comandă ”npm install” va instala toate pachetele necesare de care are nevoie aplicația pentru a putea funcționa, eliminând acest efort suplimentar al dezvoltatorului.

Socket.IO

În mod obișnuit, în cadrul aplicațiilor web, clientul va iniția comunicarea către server urmând ca acesta să răspundă cererii HTTP și să furnizeze datele cerute. O cerință des întâlnită în cadrul aplicațiilor web este ca serverul să poată iniția comunicarea către client, pentru a furniza date în timp real către utilizatori.

Există o serie de metode prin care poate fi obținută această funcționalitate[13]:

- HTTP Short Polling: reprezintă o tehnică prin care clientul va interoga serverul la intervale scurte de timp (sub un minut) pentru a vedea dacă au apărut modificări asupra datelor. În cazul în care există modificări, la următoarea interogare, serverul va furniza setul de date modificat, iar în caz contrar va răspunde cu un mesaj gol. Această tehnică aduce foarte multe defecte precum: interogări inutile, procesare inutilă atât din partea clientului cât și din partea serverului, întârzieri ale datelor, etc.

- HTTP Long Polling: funcționează similar cu short polling, dar serverul va menține conexiunea deschisă cu clientul până vor fi date disponibile sau până trece o perioadă de timp definită, când conexiunea va fi încheiată, urmând să se reia pasul inițial. Deși este mai performant decât short polling-ul în ceea ce privește furnizarea datelor în timp real, această metodă aduce suprasolicitarea serverului în cazul în care vor exista multe conexiuni deschise, efect ce ar putea fi devastator pentru un serviciu expus la un număr mare de utilizatori.

[12] https://docs.npmjs.com/about-npm

[13] https://www.geeksforgeeks.org/what-is-long-polling-and-short-polling/

- Web Sockets[14]: protocol care este proiectat peste TCP și care permite comunicarea *full-duplex*. Această tehnologie elimină limitările metodelor menționate anterior, făcând posibilă comunicarea în timp real. Pentru ca acest protocol să poată funcționa, se va folosi *HTTP Upgrade header*[15] pentru a face trecerea de la protocolul HTTP la protocolul WebSockets, creând astfel canalul de comunicare în timp real. Un dezavantaj al acestei metode îl reprezintă faptul că unele dispozitive nu suportă acest protocol.

Socket.IO[36] este o bibliotecă *open-source* care permite schimbul de mesaje bidirecționale între server și client, facilitând o latenta redusă pe parcursul comunicării. Socket.IO este proiectat pentru a îmbunătăți funcționalitățile aduse de WebSockets, făcând comunicarea în timp real o cerință ușor de implementat.

Socket.IO aduce numeroase beneficii, cum ar fi:

- suport pentru o gamă largă de limbaje de programare
- comunicare bazată pe evenimente: biblioteca Socket.IO este concepută pentru o arhitectură *event-driven* [16] fiind potrivită pentru o aplicație bazată pe microservicii
- opțiuni alternative: în cazul în care dispozitivele angrenate în comunicare nu suportă protocolul WebSockets, biblioteca Socket.IO furnizează metode alternative precum HTTP long/short polling pentru a se realiza schimbul de mesaje

2.1.1.3 Python

Python[32] este un limbaj de programare cunoscut pentru popularitatea sa în domeniul IT, fiind ușor de învățat și de utilizat de către dezvoltatori. Limbajul beneficiază de o sintaxă simplă și de o abordare elegantă a principiilor de programare orientată pe obiect. Datorită numărului considerabil de biblioteci disponibile, Python este folosit în multiple domenii precum inteligența artificială, dezvoltare web, *web scraping*, analiza datelor și multe altele.

Python este un limbaj interpretat, deoarece în loc de compilatorul clasic folosit de limbajele de programare C/C++, acesta utilizează un interpretor care execută codul linie cu linie. Deși această abordare este mai lentă și consumă mai multă memorie, avantajul său constă în faptul că scriptul este rulat linie cu linie, iar în final are o dimensiune mai mică pe disc, fiind salvat doar fișierul care conține codul sursă, nu și executabilul.

[14] https://en.wikipedia.org/wiki/WebSocket

[15] https://en.wikipedia.org/wiki/HTTP/1.1_Upgrade_header

[16] https://aws.amazon.com/event-driven-architecture/

Flask

Flask[30] este un *framework* bazat pe limbajul de programare Python și este o alegere populară în rândul dezvoltatorilor software pentru aplicațiile web, deoarece oferă o abordare minimalistă, flexibilă și robustă. Filozofia din spatele *framework*-ului este simplitatea, făcând Flask o soluție preferată de orice dezvoltator, fie el începător sau avansat. Flask furnizează un server web care facilitează testarea rapidă și prototipizarea fără a fi nevoie de alte configurații pentru integrarea cu mecanisme externe. Serverul inclus în instalarea *framework*-ului ușurează procesul de depanare și accelerează dezvoltarea, fiind o opțiune potrivită pentru abordarea dezvoltării iterative. Un beneficiu suplimentar este documentația extensivă și comunitatea activă care este în continuă căutare de modalități pentru a îmbunătăți platforma prin contribuții la extinderea funcționalităților *framework*-ului *open-source*.

Scikit-learn

Scikit-learn[35] este o bibliotecă *open-source* bazată pe limbajul de programare Python și este cel mai folosit mecanism de implementare a aplicațiilor ce folosesc concepte de învățare automată, facând regresia, clasificarea sau clusterizarea să pară sarcini simple ce pot fi implementate cu doar câteva linii de cod. Scikit-learn oferă o abstractizare peste conceptele matematice complexe care guvernează domeniul învățării automate, făcând posibilă implementarea aplicațiilor doar cu câteva noțiuni elementare. Biblioteca oferă soluții simple pentru învățarea supervizată și nesupervizată, standardizarea datelor și reducerea dimensionalității. De asemenea, această bibliotecă a devenit un standard de facto în dezvoltarea aplicațiilor bazate pe învățare automată datorită integrării sale cu biblioteci dedicate manipulării datelor precum NumPy[17], SciPy[18] sau Matplotlib[19].

2.1.1.4 Mecanisme de stocare a datelor

Datele reprezintă cea mai importantă resursă dintr-o aplicație, deoarece acestea conțin informații private despre un utilizator. Există multe lucruri care trebuie luate în considerare când vine vorba de date, precum securitatea, deservirea rapidă a acestora, stocarea, metode de recuperare în cazul unor evenimente neprevăzute în sistem și multe altele.

Pe piață există nenumărate mecanisme de stocare, fiind esențială alegerea unei alternative care să fie potrivită pentru datele manipulate în cadrul aplicației.

[17] https://numpy.org/doc/stable/

[18] https://docs.scipy.org/doc/scipy/

[19] https://matplotlib.org/stable/index.html

PostgreSQL

Baza de date PostgreSQL[31] este un sistem de gestiune a bazelor de date *open-source*, relațională, bazată pe obiect. PostgreSQL reprezintă pilonul principal al platformei de coaching online, deoarece stochează majoritatea datelor într-un mod structurat, fiind cunoscut pentru integritatea, securitatea și suportul solid pentru tranzacții de tip ACID[20].

Există o multitudine de beneficii pe care le aduce sistemul de management al bazelor de date PostgreSQL, precum:

- performanță și scalabilitate crescută;
- consistență și fiabilitate: PostgreSQL este recunoscută pentru suportul consistent pentru tranzacțiile ACID;
- suport nativ pentru date de tip JSON;
- securitate crescută prin autentificarea pe bază de rol și criptarea datelor cu algoritmul SSL;
- extensibilitate crescută: oferă suport pentru crearea tipurilor de date personalizate;
- licență *open-source*.

MongoDB

MongoDB[25] este o bază de date de tip NoSQL care stochează informațiile sub forma unui document JSON, fiind cunoscută pentru rapiditatea și flexibilitatea prin care gestionează un flux mare de date nestructurate. Stocarea datelor sub forma de obiecte JSON face ca MongoDB să fie una dintre cele mai rapide baze de date pentru operații de scriere și citire. MongoDB este o bază de date gratuită, gestionată complet în *cloud*, fiind ideală pentru aplicații web ce necesită furnizarea rapidă a datelor.

Platforma MongoDB oferă mecanisme native de monitorizare și recuperare a datelor, prin intermediul MongoDB Ops Manager. Acesta permite dezvoltatorilor să configureze alerte personalizate pentru a putea acționa rapid în cazul unor evenimente neprevăzute. MongoDB furnizează mecanisme prin care se pot efectua *snapshot*-uri la baza de date, astfel se poate reveni la o stare anterioară funcțională a bazei de date în cazul unei anomalii. MongoDB oferă mecanisme de scalare pe orizontală prin intermediul *shard*-urilor[21], care nu oferă doar scalabilitate crescută, ci și redundanță, fiind posibilă recuperarea datelor în cazul eșecului unui *shard*.

[20] https://en.wikipedia.org/wiki/ACID

[21] https://en.wikipedia.org/wiki/Shard_(database_architecture)

Platforma de coaching online Fitstack foloste acest mecanism de stocare a datelor in cadrul

MinIO

MinIO[24] este un mecanism *open-source* folosit pentru stocarea obiectelor, integrându-se nativ cu platforma Kubernetes. MinIO oferă API-uri compatibile cu Amazon Web Services S3[22] și suportă toate caracteristicile acestuia, fiind foarte fluidă trecerea dintr-un mediu de producție gestionat în *cloud*, într-un mediu de producție privat care are control absolut asupra datelor. MinIO poate fi instalat în orice mediu de producție, fie că este vorba despre *cloud* public, *cloud* privat sau infrastructură de tip *bare-metal*[23]. MinIO este utilizat în principal pentru stocarea datelor nestructurate, precum fișiere media, fișiere de jurnalizare, fișiere de *backup*, imagini Docker, etc.

MinIO este o soluție modernă ce oferă numeroase beneficii unei organizații:

- **Scalabilitate**: MinIO poate scala cu ușurință prin adăugarea de noduri în clusterul de Kubernetes, fiind o soluție potrivită în cazul în care sunt gestionate cantități mari de date.
- **Eficiență**: MinIO folosește o arhitectură distribuită care permite procesarea paralelă a datelor.
- **Securitate**: MinIO oferă mecanisme de criptare a datelor.
- **Flexibilitate**: MinIO se poate instala oriunde, oferind posibilitatea de stocare a cantităților mari de date.
- **Compatibilitate**: MinIO este compatibil cu Amazon S3 API.
- **Costuri reduse**: MinIO este un mecanism de stocare *open-source*, fiind gratuit de instalat și configurat.
- **Interfață intuitivă**: MinIO permite utilizarea unei interfețe intuitive pentru configurare și monitorizarea datelor.
- **Personalizare**: MinIO poate fi customizat în funcție de cerințele proiectului.
- **Integrare**: MinIO se integrează cu majoritatea limbajelor de programare precum Go, Java, Python, JavaScript, etc.

[22] https://aws.amazon.com/s3/

[23] https://en.wikipedia.org/wiki/Bare-metal_server

2.1.1.5 Patternuri software arhitecturale

Dezvoltatorii se întâlnesc deseori cu anumite probleme când vine vorba de proiectarea unei arhitecturi care să manipuleze datele în mod eficient. Aceste patternuri arhitecturale vin în ajutorul inginerilor software deoarece oferă soluții standardizate care se potrivesc în situații similare[15].

Layered Pattern

Patternul arhitectural bazat pe straturi este unul dintre cele mai utilizate de catre dezvoltatori, intrucat functionalitatile sunt impartite in module diferite in cadrul aplicatiei.

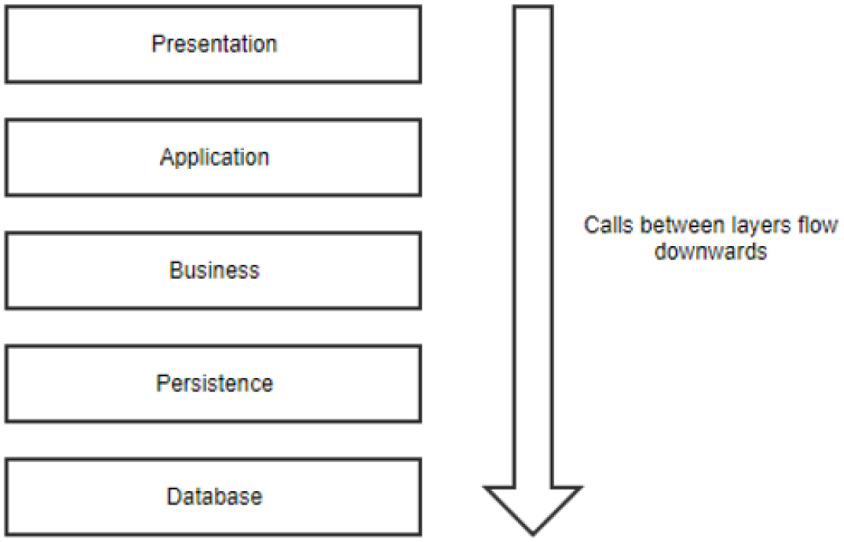

Fig. 2.1: Layered Arhitectural Pattern[24]

Fiecare strat are un rol bine definit în cadrul arhitecturii, acestea comunicând între ele în mod unidirecțional. Această arhitectură aduce cu sine o serie de beneficii importante:

- Testare simplificată: Izolarea serviciilor în module diferite care au funcționalități specifice face posibilă simplificarea testării. Prin această segregare se pot realiza teste unitare asupra modulelor, celelalte servicii cu care comunică fiind simulate.
- Organizare crescută: Acest pattern oferă o imagine clară dezvoltatorilor asupra modului în care ar trebui să arate aplicația.
- Separarea responsabilităților: Fiecare strat va avea o responsabilitate bine definită, făcând astfel codul mult mai intuitiv și ușor de înțeles.
- Scalabilitate și performanță: Straturile pot scala în mod independent unul față de celălalt.

[24] https://www.redhat.com/architect/5-essential-patterns-software-architecture

Microservice Pattern

Patternul arhitectural de tip microservicii oferă o soluție pentru o arhitectură bazată pe mai multe module, care, deși pot fi dezvoltate și instalate în producție independent, sunt proiectate pentru a colabora, oferind scalabilitate crescută sistemului. Acest pattern respectă principiile de design dictate de *Separation of concerns*[25], deoarece separă aplicația in module decuplate, fiecare având scopul ei bine definit în cadrul infrastructurii.

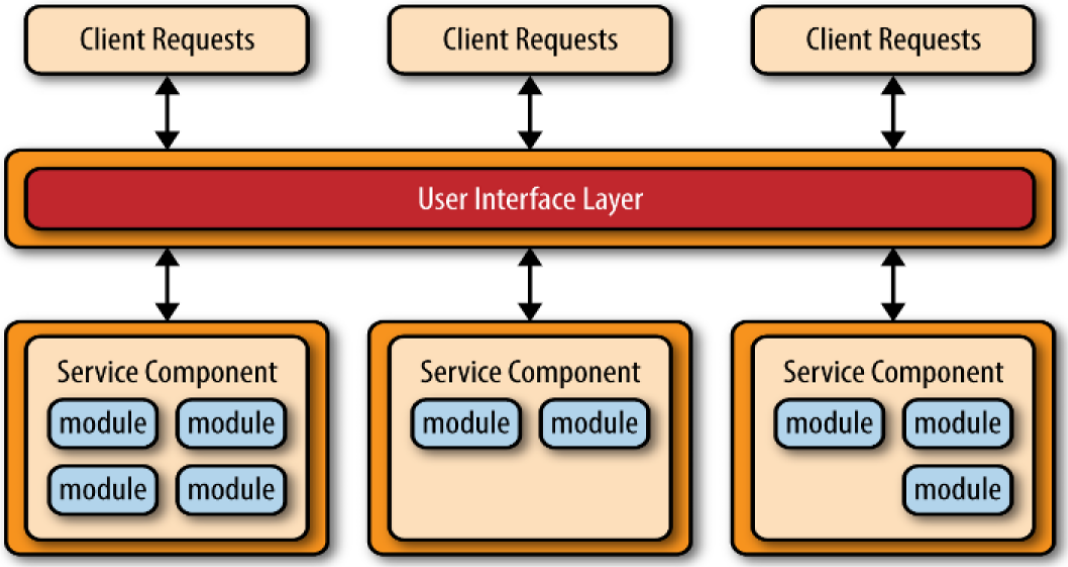

Fig. 2.2: Microservices Pattern[26]

Monolith Vs Microservicii

Această abordare este una modernă care vine ca răspuns la dezavantajele aplicațiilor de tip *monolith*[27]. Adjectivul *monolithic*[28] provenit din limba engleză se traduce ca ”prea mare”, sugerând din nume că acest tip de arhitectură are o multitudine de defecte, fiind o soluție învechită pentru cerințele unei aplicații cu un număr ridicat de utilizatori.

În următorul tabel2.1 vor fi prezentate o serie de avantaje și dezavantaje atât pentru arhitectura de tip microservicii, cât și pentru cea de tip monolith[12].

[25] https://en.wikipedia.org/wiki/Separation_of_concerns
[26] https://www.redhat.com/architect/5-essential-patterns-software-architecture
[27] https://www.techtarget.com/whatis/definition/monolithic-architecture
[28] https://dictionary.cambridge.org/us/dictionary/english/monolithic

	Monolith	Microservicii
Avantaje	- folosit pentru aplicații mici - simplu de proiectat - dezvoltare rapidă - instalare în producție simplificată, deoarece există doar un singur artefact - depanare simplificată	- scalabilitate crescută - flexibilitate - reziliență - mentenanță crescută - se pot folosi tehnologii diferite pentru servicii - decuplarea infrastructurii
Dezavantaje	- mentenanță scăzută - scalibiltate scăzută - se paote folosi doar o singuă tehnologie pentru dezvolatre - instalarea in producție durează mult - toleranța la defecte scazută	- complexitate crescută - overhead ridicat pentru comunicarea între microservicii - instalare în producție dificilă, deoarece există mai multe artefacte - monitorizare si depanare dificilă - costuri crescute - testare dificilă

Tabel 2.1: Avantaje/dezavantaje arhitectura monolith/microservicii

În urma studiului comparativ prezentat în tabelul anterior, se pot trage câteva concluzii:

- Arhitectura de tip microservicii se pretează în cazul aplicațiilor complexe, scalabile, în care se dorește utilizarea mai multor tehnologii pentru dezvoltare, venind cu o serie de dezavantaje legate de configurare, testare, monitorizare și costuri.
- Arhitectura de tip monolit se folosește pentru aplicații simple, cu un număr mic de utilizatori, având avantajul testării, depanării și monitorizării simplificate.

Deși sunt mai greu de gestionat, aplicațiile bazate pe microservicii sunt necesare în contextul actual, deoarece organizațiile își doresc maximizarea profitului, deci un număr cât mai mare de utilizatori.

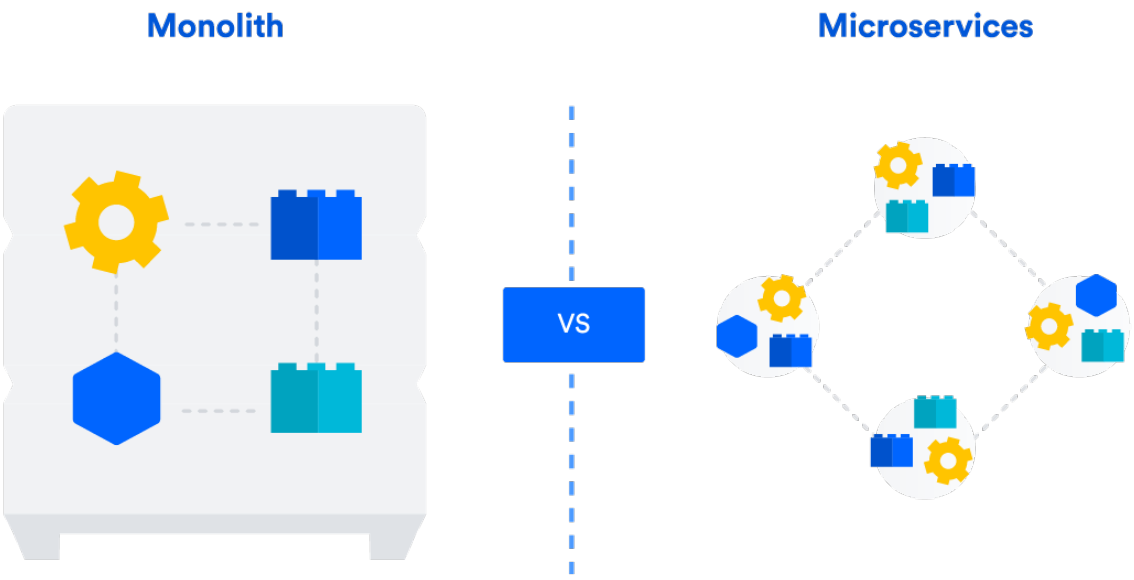

Fig. 2.3: Monolith Vs Microservice[29]

Publisher Subscriber Pattern

Sablonul arhitectural de tip publisher-subscriber[30] funcționează pe baza unui dispecer care redirecționează mesajele către sursele abonate la el. Actiunea de a trimite mesaje către dispecer se numește *publish*, iar actiunea de a citi mesajele se numește **subscribe**. Acest pattern arhitectural se regăsește în platforma de coaching online la nivelul serviciului de mesagerie în timp real, deoarece se folosește de resursa din Azure Pub-Sub for Socket.IO[22], care propune o soluție scalabilă, eficientă și rezilientă. Această soluție este proiectată pe baza bibliotecii Socket.IO[36], care furnizează schimb de mesaje în mod asincron și în timp real.

[29] https://www.atlassian.com/microservices/microservices-architecture/microservices-vs-monolith

[30] https://www.redhat.com/architect/pub-sub-pros-and-cons

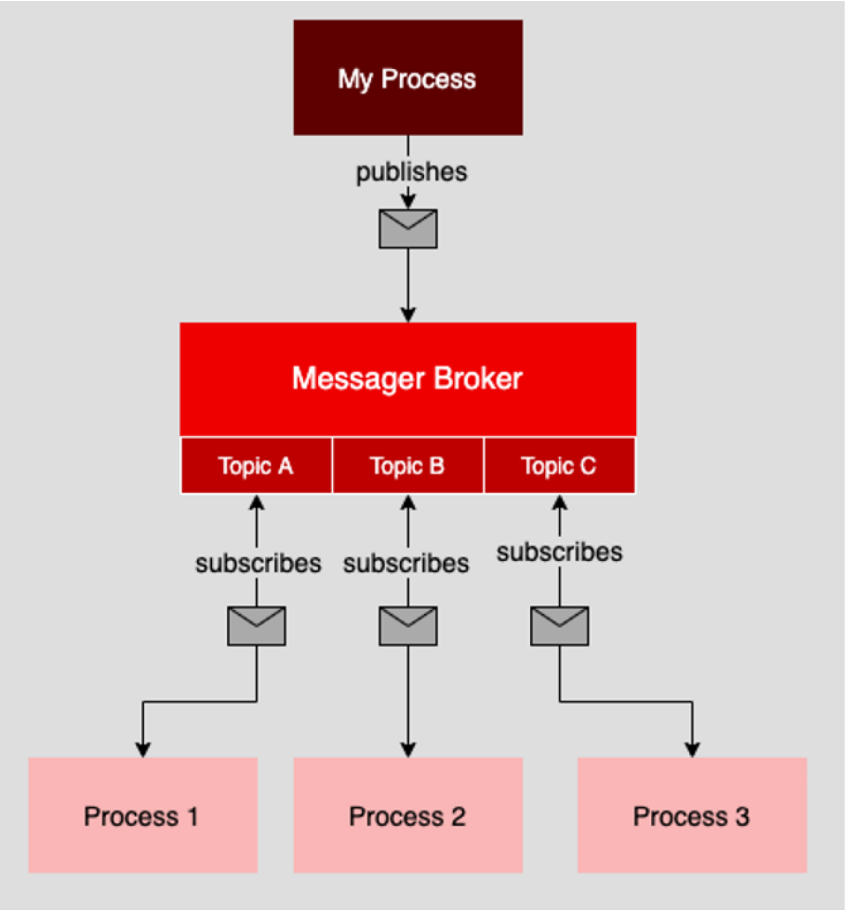

Fig. 2.4: Publisher-subscriber pattern[31]

2.1.2 Învățarea automată

Inteligența artificială si învățarea automată au cunoscut o creștere exponențială în domeniul IT, fiind din ce în ce mai integrate în aplicații, deoarece automatizează procese repetitive care pentru oameni ar putea fi consumatoare de timp și resurse. Aceste tehnologii avansate aduc beneficii semnificative, inclusiv eficiență crescută și analiză rapidă a datelor.

2.1.2.1 Regresia

Regresia este un concept fundamental al învățării automate prin intermediul căruia se stabilește o relație de asociere între caracteristici și etichetă, fiind folosită în practică pentru prezicerea unor informații numerice cum ar fi prețul caselor pe baza unor detalii despre suprafața imobilului, zona, numărul de camere, etc. Regresia se încadrează în categoria învățării supervizate[32], deoarece antrenarea este realizată pe un set de date etichetat.

[31] https://www.redhat.com/architect/pub-sub-pros-and-cons
[32] https://en.wikipedia.org/wiki/Supervised_learning

Există mai multe tipuri de regresie care sunt folosite în situații diferite, în funcție de natura și complexitatea datelor:

- Regresie liniară simplă: relația dintre etichete și o singură caracteristică este modelată ca fiind liniară.
- Regresie liniară multiplă: relația dintre etichetă și un set de caracteristici este modelată ca fiind liniară.
- Regresie polinomială: extinde modelul de regresie liniară pentru a surprinde relația neliniară dintre date.
- Regresie logistică: în acest caz, răspunsul este unul categorial, determinat de probabilitatea ca un element să aparțină unei clase.

În cadrul platformei de coaching online, este folosită regresia liniară multiplă pentru a surprinde relația dintre etichetă, care este numărul de calorii, și caracteristicile care sunt genul, vârsta, greutatea, înălțimea și durata exercițiului.

Formula pentru regresia liniară multiplă este următoarea:

$$y_i = \beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \cdots + \beta_k x_{i,k} + \epsilon_i \tag{2.1}$$

În cadrul formulei anterioare, y_i este predicția modelului de regresie, $\beta_1, \beta_2, ...\beta_n$ reprezintă ponderile calculate în urma procesului de antrenare a modelului, $x_{i,1}, x_{i,2}, ...x_{i,n}$, sunt caracteristicile pentru care se dorește să se facă o predicție, β_0 reprezintă *bias*-ul si ϵ_i este zgomotul sau eroarea. Prin antrenarea modelului de regresie liniară multiplă se vor calcula ponderile și *bias*-ul, care au ca scop prezicerea unei valori cât mai aproape de realitate pe baza unui set de caracteristici.

2.1.2.2 Descrierea corpusului

Alegerea și curățarea unui set de date reprezintă un proces premergător celui de creare a modelului de învățare automată și joacă un rol crucial în determinarea rezultatelor și performanțelor acestuia.

Setul de date ales pentru dezvoltarea modelului de regresie a fost selectat din colecția cuprinzătoare de corpusuri disponibile pe Kaggle[33], platformă cunoscută pentru calitatea datelor, care sunt preluate în mare parte din studii științifice și proiecte de cercetare.

Setul de date ales cuprinde 15,000 de înregistrări, fiecare furnizând informații complete despre gen, vârstă, înălțime, greutate, durata exercițiului, ritmul cardiac și temperatura corporală. Aceste caracteristici sunt însoțite de eticheta care

[33] https://www.kaggle.com/docs

este numărul de calorii arse în timpul exercițiilor efectuate de persoanele care au luat parte la acest studiu.

Acest set de date este unul complex, fiind nevoie doar de schimbarea caracteristicii categoriale ca gen din valorile posibile ”female” si ”male” in valorile 0 si 1 pentru a putea include si acest camp in calcularea modelului de regresie. Există o asociere puternică între durata exercițiului fizic și numărul de calorii arse, cu o valoare maximă de corelație între acestea, fiind un element esențial în crearea modelului de regresie.

2.1.3 Tehnologii Frontend

2.1.3.1 Flutter

Flutter[14] este un *framework open-source*, creat pentru dezvoltarea aplicațiilor multi-platformă, fiind una dintre cele mai folosite unelte din ultima perioadă. Acesta este un *framework* creat de Google în anul 2017, reprezentând una dintre cele mai simple metode de a proiecta aplicații cross-platform. Flutter este compatibil cu iOS, Android, Windows, Linux, macOS și majoritatea browserelor și este folosit atât pentru dezvoltarea aplicațiilor native, cât și pentru implementarea interfețelor web.

Flutter este dezvoltat pe baza Dart[34], care este un limbaj de programare orientat pe obiect și este foarte ușor de folosit, datorită structurii și sintaxei simplificate. Acest limbaj este cunoscut pentru eficiență și rapiditate, fiind o bază solidă pentru un *framework* dedicat dezvoltării interfețelor interactive și fluente.

Aplicațiile Flutter rulează pe o mașină virtuală care permite hot reload[35], fără a fi nevoie de a recompila codul de fiecare dată pentru a fi vizibile în interfață modificările din cod. De asemenea, Flutter are o gamă largă de API-uri parte terță cu care se poate integra direct din biblioteci furnizate de *framework*.

Arhitectura *framework*-ului Flutter

Arhitectura *framework*-ului Flutter constă în trei elemente principale:

- **Flutter framework**: reprezintă cea mai vizibilă parte a tehnologiei Flutter. Dezvoltatorii interfețelor multi-platformă interacționează cu tehnologia Flutter prin intermediul acestui strat, deoarece dispune de o gamă largă de biblioteci care, la rândul lor, furnizează programatorilor toate elementele vizuale de care au nevoie pentru implementarea unei interfețe robuste.

[34] https://dart.dev/guides

[35] https://docs.flutter.dev/tools/hot-reload

- **Engine**: Reprezintă nucleul *framework*-ului Flutter și este implementat în mare parte în limbajul de programare C++. Acest strat este responsabil cu crearea imaginii pe toate platformele suportate de Flutter. De asemenea, este responsabil cu implementarea API-ului central al *framework*-ului Flutter, incluzând grafica, layout-ul textului, I/O pentru fișiere și rețea, arhitectura de pluginuri și mediul de funcționare Dart și metode de depanare și compilare.
- **Embedder**: Componenta funcțională ce ajută *framework*-ul Flutter să poată funcționa pe orice sistem de operare suportat.

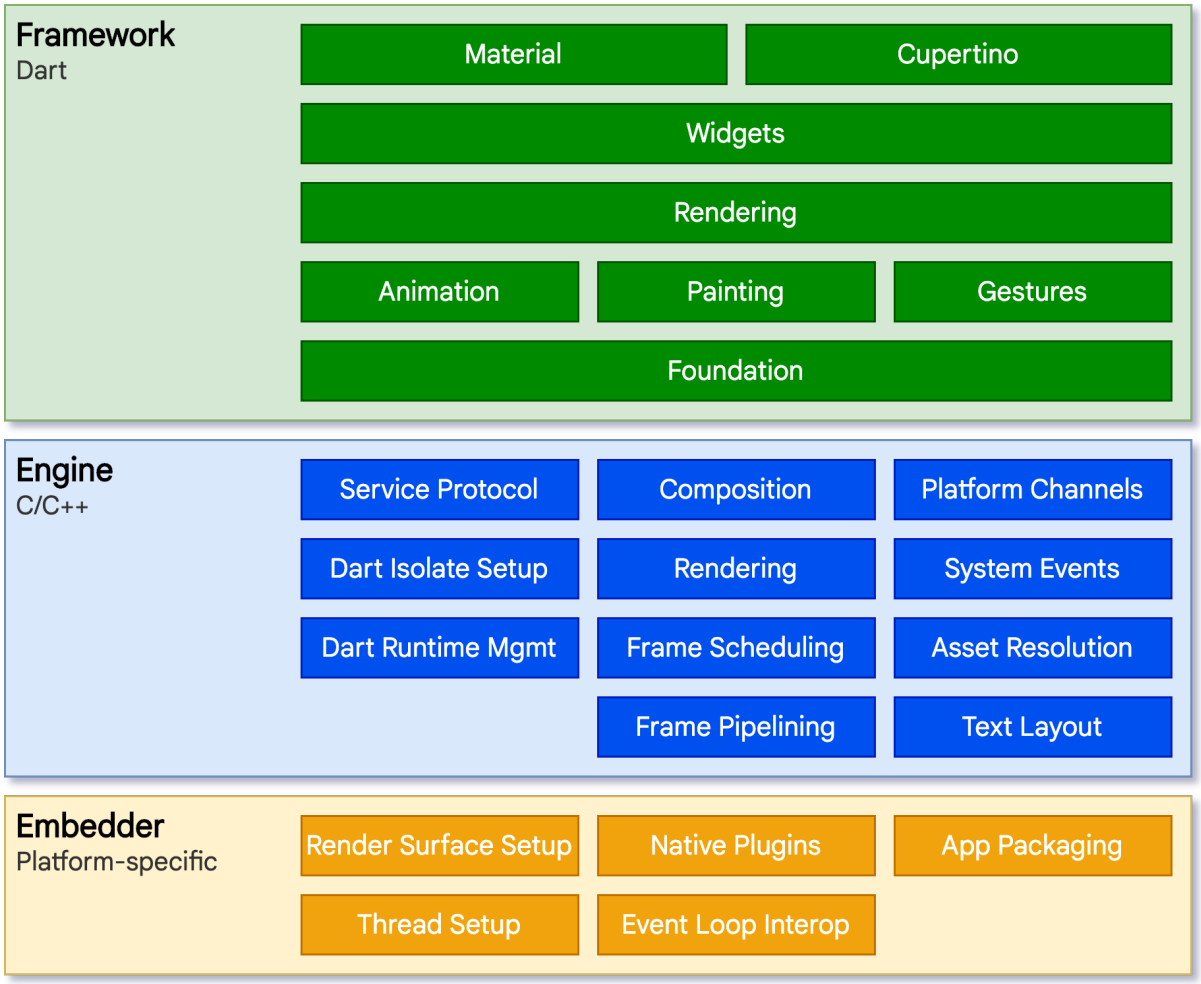

Fig. 2.5: Arhitectura Framework-ului Flutter[36]

Widget-uri

Un *widget* reprezintă o componentă cheie când vine vorba despre dezvoltarea aplicațiilor implementate cu ajutorul *framework*-ului de dezvoltare Flutter. Un

[36] https://docs.flutter.dev/resources/architectural-overview

widget este folosit pentru a descrie o componentă vizuală a aplicației, precum: butoane, text, imagini, etc.

Aplicațiile Flutter sunt structurate prin intermediul unei ierarhii de *widget*-uri cunoscută și ca *Widget Tree*. La cel mai înalt nivel se află MaterialApp[37], care definește structura aplicației Flutter, urmând ca celelalte *widget*-uri adiționale să se incuibareze unele în altele pentru a crea o interfață robustă și complexă.

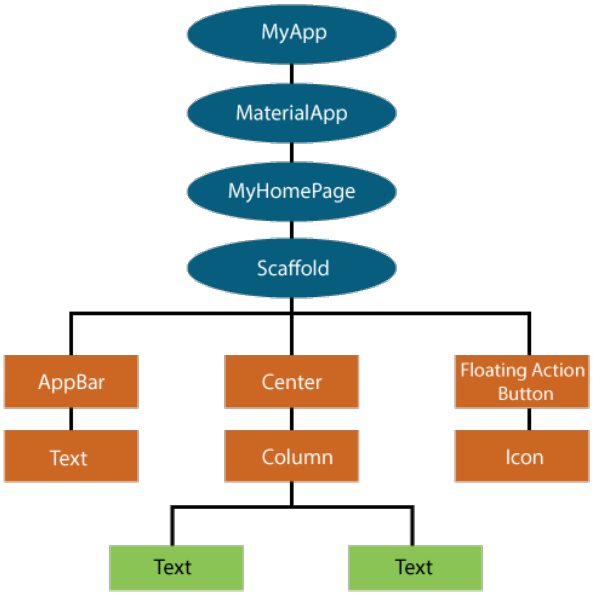

Fig. 2.6: Widget Tree Flutter[38]

Un element important când vine vorba de *widget*-uri este starea acestora, care poate fi:

- **Stateless**: Un widget stateless este imutabil, ceea ce înseamnă că nu își schimbă starea pe parcurs. Acestea sunt folosite atunci când nu se dorește ca un widget să aibă o stare internă care să modifice elementele vizuale din cadrul ferestrelor aplicației.
- **Stateful**: Acestea își pot schimba starea internă pentru a modifica anumite elemente din pagină în cazul unor evenimente.

2.1.3.2 React.js

React.js este o bibliotecă bazată pe limbajul de programare JavaScript și este cunoscută pentru notorietatea și creșterea sa exponențială în popularitate pe piața IT. Această bibliotecă a fost creată de compania Facebook[39] în anul 2011 și a fost inițial concepută pentru uz intern în cadrul organizației. În anul 2013,

[37] https://api.flutter.dev/flutter/material/MaterialApp-class.html
[38] https://www.javatpoint.com/flutter-widgets
[39] https://ro.wikipedia.org/wiki/Facebook

React a devenit o bibliotecă *open-source* și a cunoscut o creștere exponențială, devenind una dintre cele mai populare soluții de dezvoltare a interfețelor web.

Componente

Dezvoltatorii interfețelor web au fost inițial reticenți în ceea ce privește biblioteca React.js, deoarece aceasta folosește atât taguri HTML, cât și cod JavaScript, în același fișier. Cu toate acestea, popularitatea sa a crescut rapid, deoarece React este conceput pentru a crea componente reutilizabile în cadrul aplicației. Această abordare, deși combină elemente de HTML cu cele de JavaScript, se dovedește a fi foarte facilă pentru dezvoltatori, crescând productivitatea și modularizarea aplicației. Arhitectura bazată pe componente contribuie, de asemenea, la creșterea nivelului de întreținere a codului.

Componentele dezvoltate de programatori vor fi încadrate unele în altele, fiind mai ușor de vizualizat sub formă arborescentă.

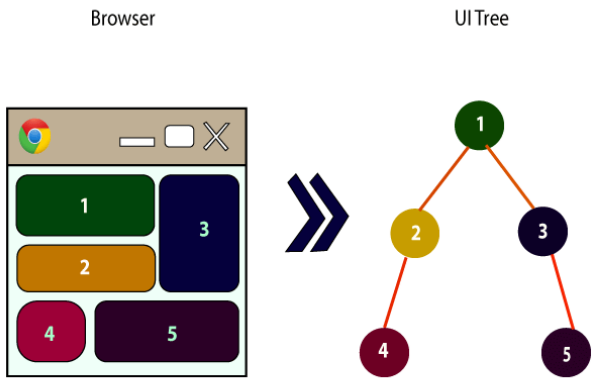

Fig. 2.7: Arbore componente React.js[40]

DOM Virtual

Un alt motiv pentru care biblioteca React.js este o soluție populară este reprezentat de conceptul de DOM[41] Virtual. Acesta este un concept în care o versiune virtuală a DOM-ului este menținută în memorie și este sincronizată periodic cu DOM-ul real. Această abordare permite natura declarativă a bibliotecii React.js, care reconciliază starea interfeței cu DOM-ul.

[41] https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model

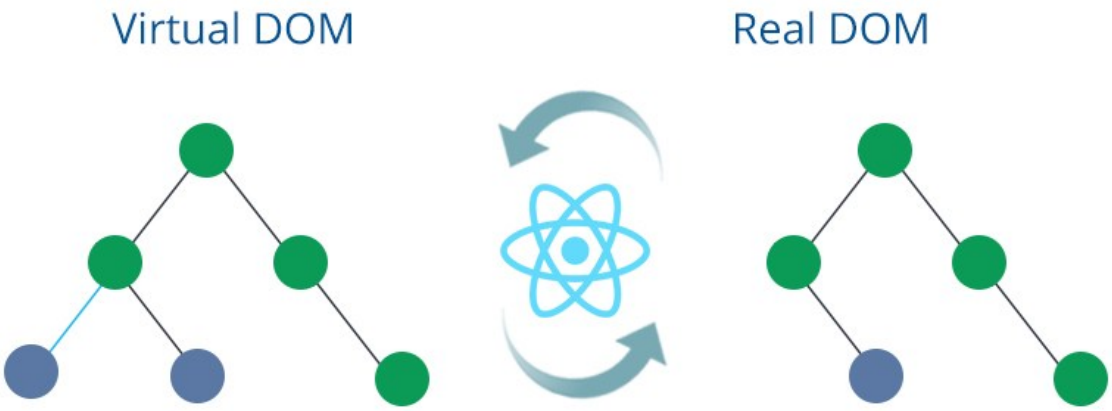

Fig. 2.8: DOM Virtual[42]

2.2 Tehnologii pentru monitorizare

Monitorizarea reprezintă un proces esențial în contextul oricărui mediu de producție, deoarece nu furnizează doar date agregate și alerte despre sistem și starea acestuia, ci oferă și o imagine de ansamblu a organizației, fiind un real ajutor în procesul depanării infrastructurii în cazul unor evenimente neprevăzute. Monitorizarea oferă informații importante despre utilizarea procesorului, utilizarea memoriei persistente și efemere, erori, excepții, *bandwidth*, starea podurilor și numărul lor, starea generală a clusterului de Kubernetes și multe alte detalii importante.

2.2.1 Metrici și monitorizare în Azure

Platforma Microsoft Azure[18] furnizează servicii de monitorizare[19], oferind o serie considerabilă de metrici, alerte și metode de vizualizare a datelor prin intermediul unor grafice.

Azure permite configurarea a trei tipuri de metode de monitorizare gestionate complet de aceștia:

- Metrici native: Azure oferă posibilitatea configurării unor metrici native pentru monitorizarea resurselor. Un avantaj al acestei alternative este faptul că este foarte rapidă, nu necesită configurări și este gratuită, dar ofera un nivel mic de personalizare a mecanismului de monitorizare.

- Metrici personalizate: această abordare permite un nivel mai mare de personalizare, întrucât metricile sunt colectate de la agenți de monitorizare

[42] https://blog.arrowhitech.com/virtual-dom-its-definition-and-benefits-that-you-must-know/

furnizați de Azure[43]. Un avantaj este nivelul ridicat de personalizare a metricilor, dar prevede un cost suplimentar.

- Metrici Prometheus[44]: Prometheus este un instrument *open-source* de colectare a metricilor și alertare. Azure oferă suport nativ pentru acest instrument, fiind nevoie doar de activarea opțiunii de monitorizare cu Prometheus pentru un cluster de Kubernetes. De asemenea, acesta se integrează cu platforme de vizualizare a datelor precum PromQL[45] sau Grafana Dashboards[46] care este o unealtă *open-source* de vizualizare a datelor cu suport nativ pentru platforma Azure. Această alternativă oferă un nivel ridicat de personalizare a mecanismelor de monitorizare, dar prevede un cost suplimentar.

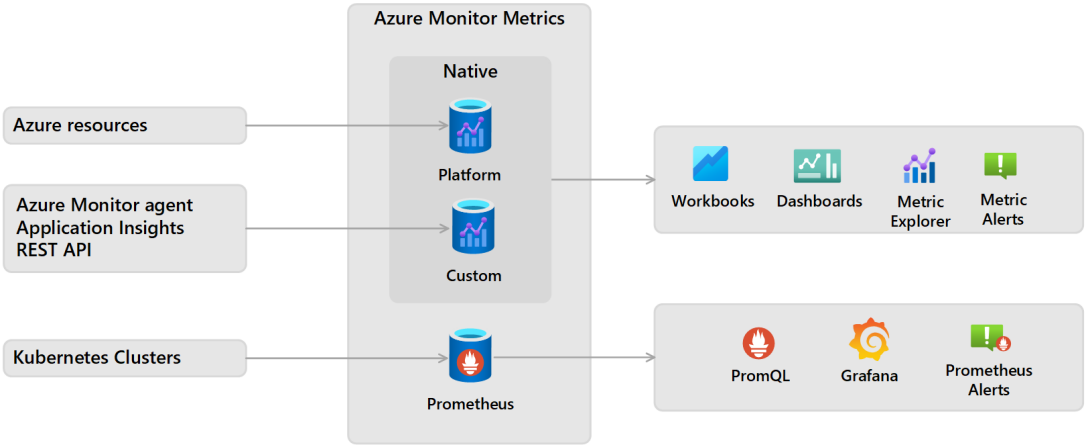

Fig. 2.9: Monitorizarea metricilor cu platforma Azure[19]

2.3 Tehnologii pentru deployment

2.3.1 Docker

Docker[6] este o platformă *open-source* proiectată pentru dezvoltarea aplicațiilor într-un mediu izolat. O aplicație care funcționează prin intermediul platformei Docker va avea toate dependințele necesare pentru a rula, ceea ce înseamnă că ea poate fi executată pe orice platformă care are Docker instalat local.

Pentru a înțelege mai în profunzime tehnologia din spatele platformei Docker trebuie menționați o serie de termeni cheie:

- imagine: o imagine Docker este o împachetare standardizată a aplicației cu toate fișierele, binare și bibliotecile necesare pentru a putea funcționa.

[43] https://learn.microsoft.com/en-us/azure/azure-monitor/agents/agents-overview
[44] https://prometheus.io/docs/introduction/overview/
[45] https://prometheus.io/docs/prometheus/latest/querying/basics/
[46] https://grafana.com/grafana/dashboards/

- container: folosit pentru împachetarea aplicațiilor în unități software standardizate. Acestea includ toate dependențele necesare pentru a putea rula, permițând portabilitatea între medii de lucru diferite. Containerele sunt instanțe ale imaginilor Docker în momentul rulării.

- registru: locație centralizată la nivelul căreia se stochează imaginile Docker sub formatul următor: nume:eticheta. Registrele pot exista atât local, cât și în *cloud*, un exemplu fiind platforma DockerHub[47] care este cel mai mare registru de containere.

Platforma Docker este o tehnologie inovatoare în domeniul dezvoltării software, deoarece aduce numeroase avantaje față de soluția precedentă, și anume, mașinile virtuale. În următorul tabel2.2 voi evidenția diferențele principale între containere și mașini virtuale[48].

	Mașina Virtuală	Container
Sistem de operare	folosește sistemul de operare al dispozitivului	propriul sistem de operare
Copy code Portabilitate	foarte portabil	Mai puțin portabil
Viteza	pornire/oprire rapidă	pornire/oprire lentă
Resurse Folosite	puține resurse folosite	multe resurse folosite
Avantaje	folosit pentru aplicații scalabile și portabile	folosit pentru aplicații izolate

Tabel 2.2: Diferențe Containere/Mașini Virtuale

2.3.2 Kubernetes

Kubernetes[16] este o tehnologie *open-source* folosită în domeniul dezvoltării software ce are ca scop orchestrarea aplicațiilor containerizate, facilitând scalabilitatea, reziliența și portabilitatea. Termenul de Kubernetes provine din limba greacă și înseamnă navigator sau pilot.

Pentru a înțelege mai bine tehnologia din spatele platformei Kubernetes trebuie menționați o serie de termeni esențiali:

- Namespace: mediu în care sunt izolate resurse Kubernetes

- Pod: reprezintă cea mai granulară unitate software ce poate fi gestionată de serviciul Kubernetes. Un pod este un grup de unu sau mai multe containere care vor avea acces la resurse partajate ca memoria și interfața de rețea.

[47] https://www.docker.com/products/docker-hub/

[48] https://cloud.google.com/discover/containers-vs-vms

Un pod va fi configurat pentru a asigura funcționarea containerelor pe care le gestionează.

- ReplicaSets: un replica set este folosit pentru asigurarea faptului că la un moment dat rulează un număr definit de replici de pod-uri. Această resursă are ca scop asigurarea scalabilității și a rezilienței.
- Deployment: reprezintă o metodă prin care dezvoltatorul poate gestiona Pod-urile și ReplicaSet-urile prin declararea unui fișier YAML, la nivelul căruia se stabilesc detalii cum ar fi: imaginile care vor rula în cadrul pod-urilor, porturile deschise, numărul de replici etc.
- Service: reprezintă o abstractizare software peste resursa *deployment* folosită prin expunerea funcționalității unui pod, atât la nivelul clusterului de Kubernetes cât și în internet.
- Ingress: resursa din cadrul platformei Kubernetes care permite accesul extern clusterului prin intermediul protocolului HTTP/HTTPS. Ingress-ul va trimite mai departe traficul către serviciile existente în cadrul clusterului de Kubernetes.

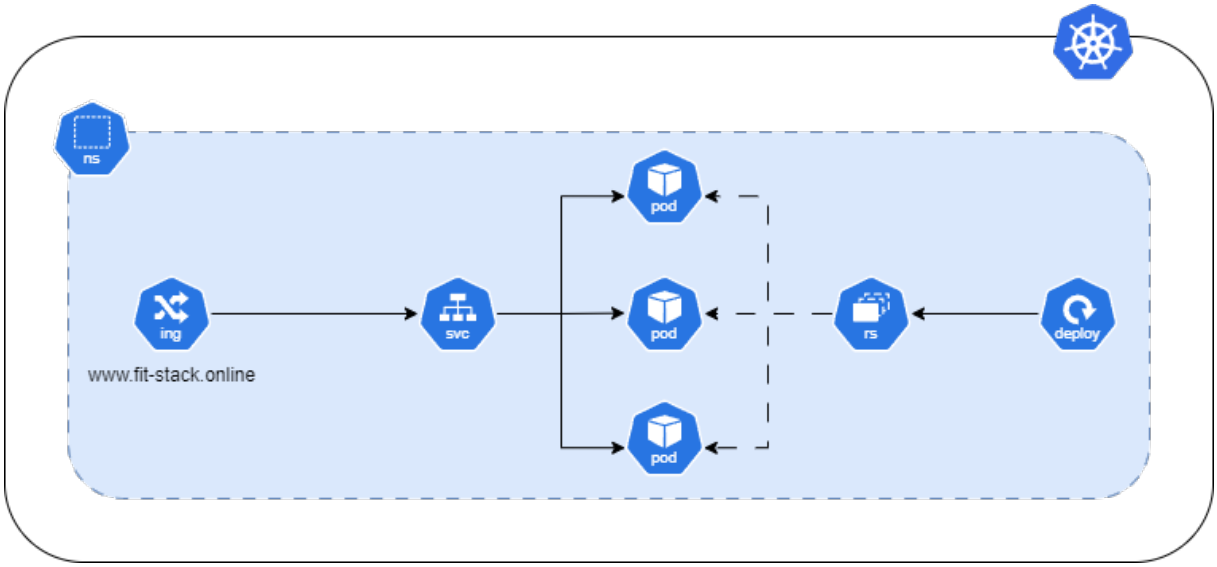

Fig. 2.10: Infrastructura Kubernetes

2.3.3 Pipline CI/CD

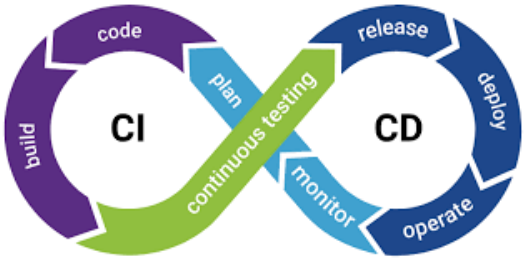

Fig. 2.11: Integrare Continuă/Livrare Continuă

CI/CD, echivalent în limba română cu Integrare Continuă/Livrare Continuă, reprezintă o modalitate eficientă de automatizare și integrare a unei noi versiuni a aplicației în mediul de producție, în momentul în care dezvoltatorul modifică starea repozitoriului Git prin adăugarea unei noi versiuni pe ramura principală. Această abordare este frecvent utilizată în mediile reale de producție, deoarece timpul de nefuncționare al aplicației scade substanțial, făcând trecerea de la o versiune veche la una nouă a aplicației un proces continuu, neafectat de greșeala umană. Un alt avantaj semnificativ al CI/CD este livrarea rapidă a unui produs software testat. Înainte ca noua versiune a aplicației să fie integrată în mediul de producție, codul este supus unor metode riguroase de evaluare, astfel încât eventualele probleme sunt identificate mult mai devreme în ciclul de dezvoltare. Această adăugire evidențiază avantajele rapide ale CI/CD, cum ar fi capacitatea de a satisface rapid cerințele utilizatorilor și de a crește satisfacția clienților.

Acest proces amplu are două părți componente:

- Integrarea continuă (CI)
- Livrare continuă (CD)

2.3.3.1 Integrare Continuă

CI, sau Integrarea Continuă, stă la baza procesului de livrare a unui cod de calitate în mediul de producție, fiind responsabilă de următoarele etape:

- compilare: verificarea dacă codul poate fi compilat cu succes, fără erori sau excepții.
- testare: rularea automată a testelor unitare pentru a verifica funcționalitatea corectă a aplicației.
- împachetare: încărcarea unei noi versiuni a aplicației într-un registru Docker ca imagine Docker.

Etapele prezentate anterior sunt interdependente, astfel încât, dacă toate acestea se finalizează cu succes, codul poate fi reintegrat în ramura principală a repozitoriului Git prin intermediul comenzii *push* sau, în cazul în care dezvoltatorul a adăugat noi funcționalități într-o ramură secundară, se poate crea un *pull request* pentru ca managerul de proiect să dea un ultim verdict.

Pentru realizarea procesului secvențial de integrare continuă, s-a optat pentru utilizarea platformei GitHub Actions[13], datorită notorietății și ușurinței cu care poate fi configurat fluxul de lucru. Fiecărui microserviciu din componenta aplicației îi este atașat un fișier în care este configurat fluxul de lucru pentru a asigura calitatea codului pentru fiecare modul în parte. Imaginile Docker, care conțin software-ul testat și toate dependențele necesare pentru a putea rula, sunt încărcate pe platforma online Docker Hub [5]. Acesta este cel mai mare registru de containere din lume, fiind o soluție sigură și foarte bine documentată.

Analiză comparativă a tehnologiilor de integrare continuă

Cand vine vorba despre alegerea unei platforme de integrare continua, exista multe aspecete care trebuie luate in considerare, cum ar fi:

- Ușor de configurat și de folosit
- Software *open-source*
- Gestionare interna sau în *cloud*
- Sisteme de operare suportate
- Monetizare
- Extensii software
- Compatibilitate cu aplicații de control a versiunii

Îm următorul tabel2.3 vor fi evidențiate principalele diferențe și asemănări între cele mai populare platforme de integrare prezente la momentul actual.

	GitHub Actions	Jenkins	CircleCI	AWS Code-Pipeline
Avantaje	-integrare nativă cu GitHub [37] -fișiere de configurare YAML [37] -gestionat in *cloud*[37]	-foarte customizabil -ecosistem mare de extensii software [37]	-ușor de customizat -gestionat in cloud[37] -suportă testarea paralelă[37]	-serviciu gestionat în totalitate de AWS[37] -integrare cu alte extensii disponibile in AWS[37]
Compatibil cu aplicații de control a versiunii	GitHub	Git, SVN, Mercurial[37]	GitHub, BitBucket, Gitlab[37]	GitHub, BitBucket, CodeCom-mit[37]
Software *open-source*	Da	Da	Da	Da
Monetizare	Gratuit pentru repozitoriuri publice, planuri de plată pentru cele private [37]	Gratuit	Premium si gratuit [37]	planuri de plata de tip pay-as-you-go[49]
Extensii Software	GitHub Marketplace contine mii de extensii create de dezvolatori [37]	peste 1800 de extensii disponibile [37]	peste 2500 de extensii [37]	sistem limitat de extensii

Tabel 2.3: Soluții de integrare continuă populare

Platforma GitHub Actions a fost aleasă datorită ușurinței de configurare, integrării facile cu o gamă largă de extensii furnizate de comunitate, vitezei sale ridicate, gestionării în *cloud* și poziționării sale ca o extensie naturală a aplicației GitHub, care este repozitoriul în care este publicat codul pentru platforma de coaching online.

[49] https://docs.amberflo.io/page/aws-style-pay-as-you-go-payg

2.3.3.2 Livrare Continuă

Procesul de livrare continuă reprezintă mijlocul prin care noile versiuni ale aplicației sunt încărcate din registrul de containere în mediul de producție în mod automat, garantând minimizarea timpului de nefuncționare al platformei.

Pentru acest proces sunt folosite două abordări principale:

- Reactivă: în această situație, repositoriul de Git va accesa mediul de producție pentru a instala și configura imaginile Docker. Aceasta este o modalitate reactivă, deoarece mediul de producție reacționează la cererile repositoriului de Git.

- Proactivă: această abordare presupune ca mediul de producție să interogheze la un interval definit de timp atât repositoriul de Git, cât și registrul de containere. Scopul acestei metode este ca mediul de producție să acționeze proactiv în căutarea noilor schimbări din aplicație și să se actualizeze cu noua versiune.

În următorul tabel2.4 vor fi evidențiate principalele avantaje și dezavantaje ale abordărilor prezentate anterior.

	Reactiv	Proactiv
Avantaje	- crește viteza și eficiența procesului de *deploy* - ușor de configurat - documentație exhaustivă	- ușurează procesul de instalare în producție și depanare - colaborare îmbunătățită - fiabilitate crescută - opțiuni de rollback pentru a reveni la o variantă funcțională a infrastructurii - Infrastructure as Code (IaC)[50] - securitate îmbunătățită
Dezavantaje	- securitate scăzută - nu există opțiunea de a reveni la o versiune anterioară - colaborare limitată - nu există versionarea infrastructurii - depanare limitată	- poate fi greu de configurat - necesită ca echipa de devops să aibă competențe avansate

Tabel 2.4: Avantaje/Dezavantaje abordare Reactivă/Proactivă Deploy

Potrivit informațiilor din tabel, abordarea proactivă aduce o multitudine de beneficii, fiind utilizată în platforma de coaching online.

Aceasta abordare aduce cu sine conceptul de ”GitOps”[33]. GitOps este o abordare care combină conceptele esențiale ale practicilor DevOps cu avantajele platformei de versionare Git, creând un mediu de dezvoltare și livrare continuă modern și eficient. Git este văzut ca o sursă de adevăr pentru mediul de producție, asigurând că *framework*-urile GitOps mențin constant infrastructura declarată în repozitoriu. Un beneficiu important este faptul că GitOps introduce conceptul de versionare a infrastructurii, simplificând trecerea de la o versiune la alta în cazul unei erori și îmbunătățind astfel toleranța la defecte a platformei.

FluxCD [9] este un *framework* conceput pentru dezvoltatorii care preferă abordarea proactivă, menținând mediul de producție sincronizat cu cel de dezvoltare. FluxCD oferă numeroase beneficii, inclusiv posibilitatea de integrare cu diverse alte platforme pentru monitorizare și securitate. Acest *framework* funcționează în clustere Kubernetes, oferind un control robust și flexibil asupra

[50] https://learn.microsoft.com/en-us/devops/deliver/what-is-infrastructure-as-code

implementării și gestionării aplicațiilor.

FluxCD funcționează prin intermediul unor controlere instalate la nivelul clusterului Kubernetes. Aceste controlere reprezintă poduri în cadrul clusterului, configurate pentru a monitoriza modificările survenite în repozitoriul Git, Helm sau registrul de containere. Prin urmare, aceste controlere sunt activate și gestionează automat actualizările corespunzătoare dacă apare o modificare în declarația infrastructurii și în versiunile containerelor sau a helm chart-urilor.

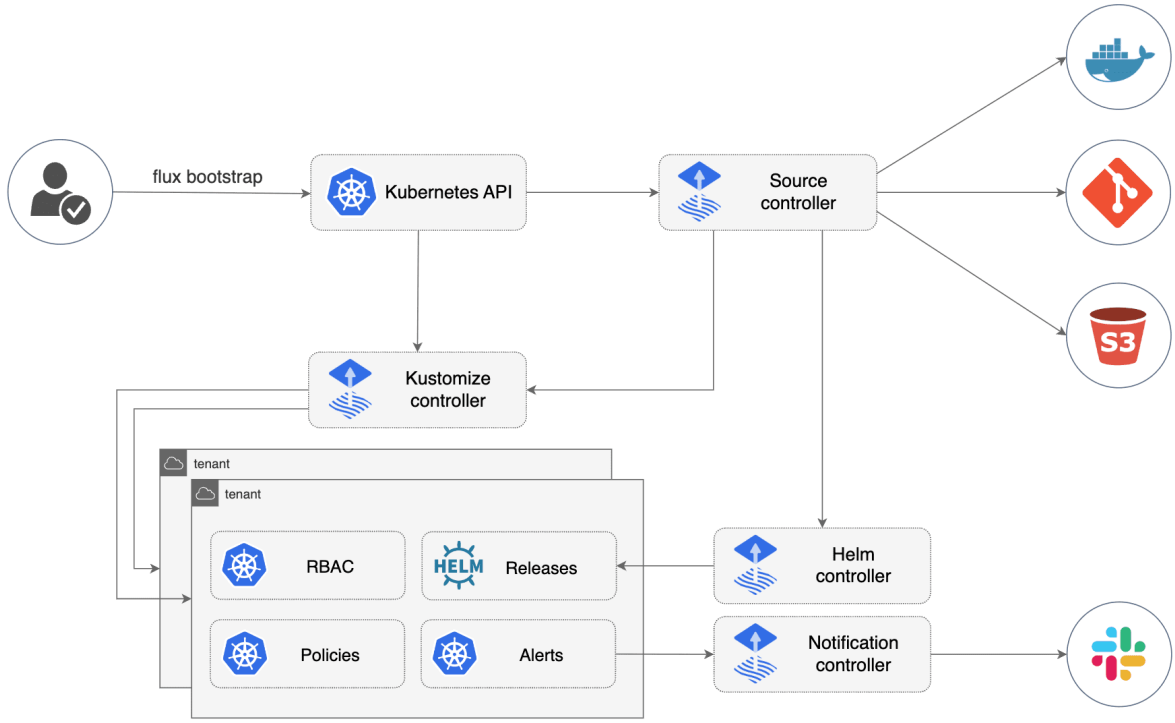

Fig. 2.12: Diagramă Controlere FluxCD[51]

În diagrama2.12 este prezentată arhitectura serviciului FluxCD, fiind ilustrate controalele necesare pentru a se realiza sincronizarea cu repozitoarele monitorizate:

- Controler de sursă: folosit pentru sincronizarea infrastructurii declarate în repositoriul de Git.
- Controler de Helm: configurat pentru a reconcilia cu repozitoriul de Helm pe care îl monitorizează serviciul FluxCD.
- Controler de notificări: utilizat pentru primirea/trimiterea notificărilor în situația în care s-a întâmplat un eveniment. Spre exemplu, în cazul în care s-a întâmplat o sincronizare cu un repozitoriu de Git, controlerul responsabil de notificări poate trimite un mesaj echipei de dezvoltare pe platforme cum ar fi Microsoft Teams sau Discord.

[51] https://fluxcd.io/flux/components/

- Controller de ”Kustomize”: colaborează cu controlerul de sursă pentru a actualiza infrastructura, folosind serviciul Kustomize furnizat de platforma Kubernetes. La nivelul acestui serviciu se realizează integrarea aplicației în mediul de producție prin încheierea execuției pod-urilor și crearea unora noi care vor rula noua versiune.

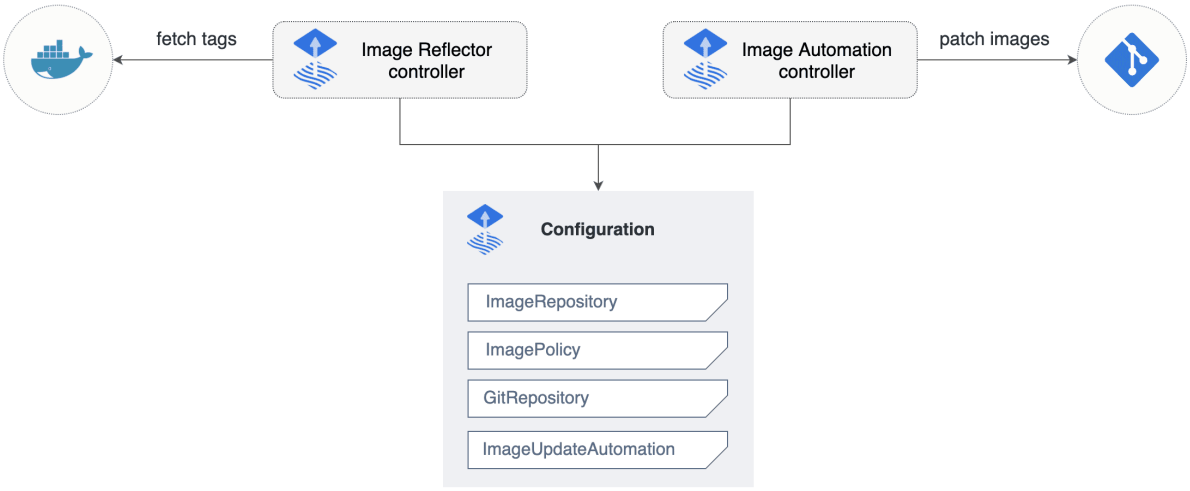

Fig. 2.13: Controler de reflectare/automatizare a imaginilor[52]

În diagrama2.13 sunt prezentate două elemente esențiale configurate de FluxCD:

- Controler de reflectare
- Controler de automatizare

Controlerul de reflectare/automatizare a imaginilor reprezintă o componentă principală prin intermediul căreia este actualizată infrastructura astfel încât să fie utilizată în permanență ultima versiune a aplicației.

Controlerul de reflectare a imaginilor este configurat de către dezvoltator să monitorizeze în permanență containerele prezente în registrul de imagini Docker. Pentru aceasta este configurată o politică a imaginilor[53] prin care controlerul de reflectare să poată declanșa procesul de automatizare, alegând cea mai actuală variantă a aplicației, în funcție de modalitatea de versionare aleasă de echipa de dezvoltare. FluxCD permite versionare numerică, alfabetică și semantică, iar în proiectarea platformei de coaching a fost aleasă ultima metodă.

Controlerul de automatizare este responsabil pentrua actualizarea fișierelor de configurare a infrastructurii cu cea mai recentă etichetă a imaginii Docker, astfel utilizând ultima versiune a aplicației. Această editare a fișierelor de

[52] https://fluxcd.io/flux/components/image/
[53] https://fluxcd.io/flux/components/image/imagepolicies/

configurare va declanșa o sincronizare gestionată de controlerul de sursă, care va detecta schimbarea și va încheia execuția pod-urilor ce rulează versiunea veche a aplicației, permițând crearea altor pod-uri cu versiunea actualizată.

2.3.4 Platforma Azure

Microsoft Azure este o platformă de *cloud computing*, ce oferă o gamă largă de servicii cum ar fi: metode diverse de stocare a datelor, modele de inteligență artificială și servicii de calcul ca mașini virtuale și clustere. Azure oferă posibilitatea dezvoltatorilor să proiecteze infrastructuri într-un mod eficient, garantând prin contract (Service Level Agreement[54]) că serviciile furnizate de ei vor fi disponibile cel puțin 99.9% din timp. Datorită notorietății sale, serviciilor furnizate și integrării, aplicația de coaching online este disponibilă pe platforma Azure.

2.3.4.1 Azure Kubernetes Service

Azure Kubernetes Service (AKS) reprezintă un serviciu de Kubernetes gestionat de platforma Azure, care poate fi folosit pentru a gestiona aplicații containerizate, având nevoie de o expertiză minimă în folosirea serviciului de Kubernetes pentru orchestrare. AKS este o platformă ideală pentru gestionarea aplicațiilor containerizate, care necesită disponibilitate ridicată, scalabilitate și portabilitate, făcând posibilă integrarea cu instrumentele de DevOps existente pe piață[20].

AKS permite integrarea cu multe servicii disponibile în Azure cum ar fi:

- Managementul identității și securității: AKS permite aplicarea politicilor de securitate, permitând doar utilizatorilor avizați accesul la resursele din cluster. Pentru asta se pot folosi multe metode cum ar fi Managed Identity[55] sau Kubernetes RBAC[56].
- Metode de logging și monitorizare: AKS se integrează cu majoritatea instrumentelor de monitorizare disponibile cum ar fi Prometheus și Grafana[21].
- Scalare automată: AKS permite scalarea automată și adăugarea nodurilor în cluster în cazul în care se atinge o anumită limită de funcționare a procesorului, a memoriei RAM sau a discului.
- Integrare cu instrumente de dezvoltare: AKS permite integrarea cu Visual Studio Code pentru managementul infrastructurii.

[54] https://www.azure.cn/en-us/support/sla/summary/

[55] https://learn.microsoft.com/en-us/entra/identity/managed-identities-azure-resources/

[56] https://learn.microsoft.com/en-us/azure/aks/azure-ad-rbac?tabs=portal

2.3.4.2 DNS Zone

Platforma FitStack își are numele de domeniu furnizat de registratorul GoDaddy[57], fiind recunoscut ca unul dintre cei mai populari. Aceasta poate fi accesată la adresa www.fit-stack.online. Azure permite crearea unei zone DNS[58] care este folosită pentru gestionarea înregistrărilor DNS pentru un domeniu. În cadrul zonei de DNS din Azure este adăugat numele de domeniu fit-stack.online, la care s-a adăugat înregistrarea de tip ”A” pentru numele ”www”, la care se face rezoluție către adresa IP a serviciului care gestionează interogările din cadrul clusterului de Kubernetes.

Delegarea zonei de DNS si Azure DNS

Azure DNS permite gestionarea unei zone DNS și a înregistrărilor DNS pentru un domeniu. Domeniul părinte trebuie delegat către Azure DNS pentru ca interogările DNS ale domeniului să ajungă la Azure DNS[23].

Pentru a ilustra mai bine cum functioneaza rezolutia numelui de domeniu catre adresa IP a inregistrarii din zona de DNS gestionata de Azure, voi parcurge un exemplu, din documentatia acestora.[23].

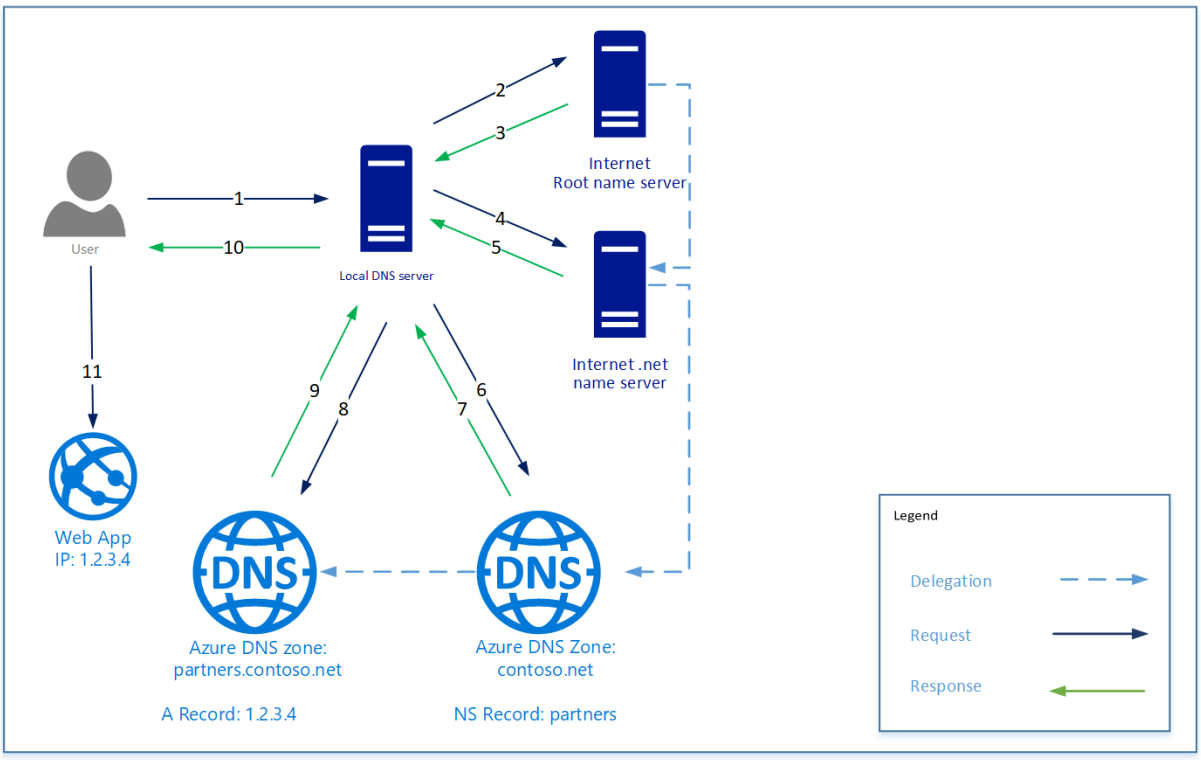

Fig. 2.14: Rezoluție DNS Azure[23]

1. Clientul dorește să acceseze pagina web www.partners.contoso.net, facilitând interogarea serverului local de DNS pentru obținerea adresei IP

[57] https://www.godaddy.com/

[58] https://learn.microsoft.com/en-us/azure/dns/dns-zones-records

aferente.

2. Serverul local nu conține această intrare, urmând interogarea serverului de DNS rădăcină.

3. Serverul DNS rădăcină nu conține această intrare, returnând răspunsul de tip ”CNAME” către serverul de nume pentru domeniul ”.net”.

4. Serverul local trimite interogarea către serverul DNS pentru domeniul ”.net”.

5. Serverul de nume ”.net” nu conține această intrare, returnând răspunsul DNS de tipul ”CNAME” către serverul ”contoso.net”. În acest caz va răspunde cu adresa serverului de nume din cadrul zonei de DNS furnizată de platforma Azure.

6. Serverul local trimite interogarea către serverul de nume ”contoso.net” din cadrul zonei de DNS.

7. Serverul de nume din cadrul zonei de DNS nu conține intrarea dorită, dar cunoaște serverul de DNS pentru numele de domeniu ”partners.contoso.net”.

8. Serverul local trimite interogarea către serverul de nume partners.contoso.net din cadrul zonei de DNS.

9. Serverul de DNS ”partners.contoso.net” din cadrul zonei de DNS conține înregistrarea de tip ”A” pentru ”www.partners.contoso.net” și va returna adresa IP.

10. Serverul returnează răspunsul clientului cu adresa IP dorită.

11. Clientul accesează site-ul web ”www.partners.contoso.net”.

2.3.4.3 Broker de mesaje publisher/subscriber pentru librăria Socket.io

Socket.IO [36] este o bibliotecă care are suport nativ pe platforma Azure prin intermediul resursei Web PubSub for Socket.IO. Aceasta reprezintă un dispecer de mesaje între serverul de Socket.IO și clienți, furnizând astfel comunicare în timp real a datelor. Acest serviciu ușurează munca dezvoltatorilor, întrucât, pentru a transfera aplicația în producție, aceștia trebuie doar să folosească apelul ”useAzureSocketIO()” și toată infrastructura din spate este gestionată de platforma Azure.

Această funcționalitate este complet gestionată de Azure, ceea ce aduce numeroase beneficii:

- Scalabilitate garantată: dispecerul de Socket.IO poate suporta până la 100.000 de conexiuni concurente.
- Eliminarea erorii umane: fiind un serviciu gestionat complet de Azure, eroarea umană este minimizată.
- Rezilienţă crescută: Azure garantează, prin intermediul SLA[59], un timp de funcţionare de peste 99,5%.
- Replicare între regiuni: Web PubSub pentru Socket.IO permite replicarea între regiuni geografice diferite, făcând serviciul mult mai rezilient. De exemplu, în cazul unui dezastru natural sau al unei întreruperi de alimentare, serviciul va fi în continuare disponibil.

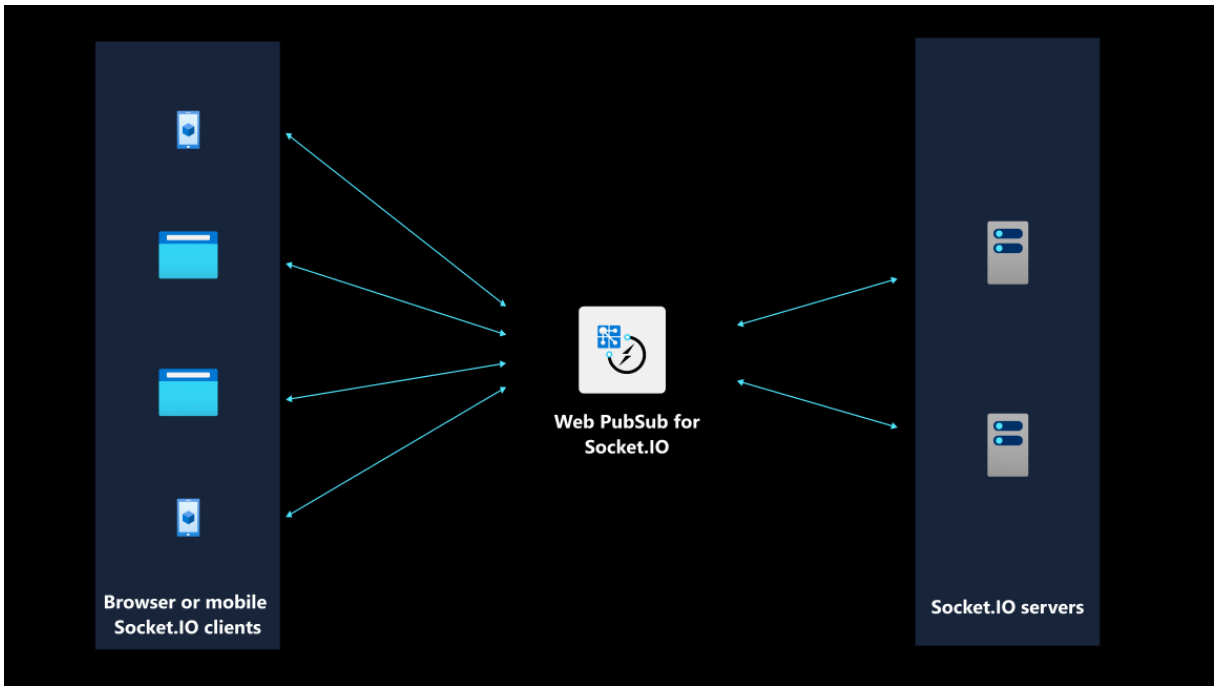

Fig. 2.15: Diagramă infrastructură Web PubSub for Socket.IO Azure[22]

Diagrama2.15 prezintă infrastructura serviciului Web PubSub pentru Socket.IO. Această implementare aduce cu sine următoarele beneficii:

- serverul de Socket.IO nu mai gestionează direct conexiunile.
- conexiuni persistente între client şi dispecer de Socket.IO, disponibile pentru o gamă variată de dispozitive.
- conexiuni persistente între server şi dispecer de Socket.IO.

[59] https://www.azure.cn/en-us/support/sla/summary/

- compatibilitate retroactivă cu o multitudine de versiuni disponibile de Socket.IO.

2.3.5 Automatizarea certificatelor

Un certificat digital este un fișier sau o parolă electronică care, prin utilizarea infrastructurii de chei publice (PKI), demonstrează că un dispozitiv, server sau utilizator este cine pretinde că este. În mediul online, certificatele digitale asigură comunicarea criptată. Ele sunt necesare pentru a autentifica serverele web și a stabili o conexiune securizată cu browserele web prin protocolul SSL/TLS. Un certificat conține o serie de informații cum ar fi numărul de versiune, numărul de serie, algoritmul de criptare, numele autorității de certificare, dar cea mai importantă informație o reprezintă cheia publică a organizației. Certificatul conține o copie a cheii publice care trebuie să fie asociată cu cheia privată a organizației pentru a se stabili conexiunea criptată între client și server.

2.3.5.1 Ciclul de viata al unui certificat

Fig. 2.16: Ciclul de viață al certificatelor[7]

Gestionarea certificatelor este o sarcină dificilă pentru dezvoltatori, întrucât aceștia trebuie să fie conștienți de toate etapele din ciclul de viață al acestora[7]:

1. Descoperirea: acest proces presupune descoperirea certificatelor compromise, revocate, neutilizate, expirate sau care trebuie înlocuite. Deși nu pare un aspect foarte important, are foarte multe implicații în eliminarea breșelor de securitate din cadrul organizației.

2. Crearea/Achiziționarea: etapa care presupune crearea unui Certificat Digital. Dezvoltatorul va trebui să completeze un CSR(Certificate Signing Request)[60] care va trebui înaintat către o autoritate de certificare. Dacă

[60] https://en.wikipedia.org/wiki/Certificate_signing_request

datele furnizate în CSR sunt legitime, autoritatea de certificare va elibera un certificat pentru cheia publică a organizației.

3. Instalarea și Stocarea: presupune instalarea certificatului la nivelul serverului și stocarea sa într-o locație sigură, dar totuși să fie disponibilă clienților care încearcă să stabilească o conexiune.

4. Monitorizare: dezvoltatorii sau sistemele automate trebuie să monitorizeze certificatele în permanență, întrucât evenimente precum expirarea sau compromiterea ar putea duce la nefuncționarea întregii organizații.

5. Reînnoire: orice certificat are precizată o dată de expirare. Administratorii trebuie să fie conștienți de această dată și să creeze un nou CSR pentru un alt certificat valid. Nu este recomandat ca un certificat să aibă o perioadă de valabilitate mai mare de 5 ani.

6. Revocare: dacă un certificat este compromis, acesta va fi revocat pentru a omite periclitarea securității organizației.

7. Înlocuire: Unele organizații preferă înlocuirea folosind o parte terță, și folosesc propriile lor PKI-uri și autorități de certificare. Acesta reprezintă un caz rar.

Prin automatizarea proceselor menționate anterior, dezvoltatorii nu ar mai fi nevoiți să parcurgă aceste etape. În schimb, totul ar putea fi lăsat în baza unor sisteme specializate, care favorizează un mediu mult mai rezilient și lipsit de eroare umană.

2.3.5.2 Cert-manager

Cert-manager este un instrument folosit pentru managementul certificatelor X.509[61], creat pentru a se integra nativ cu platforma Kubernetes.

[61] https://en.wikipedia.org/wiki/X.509

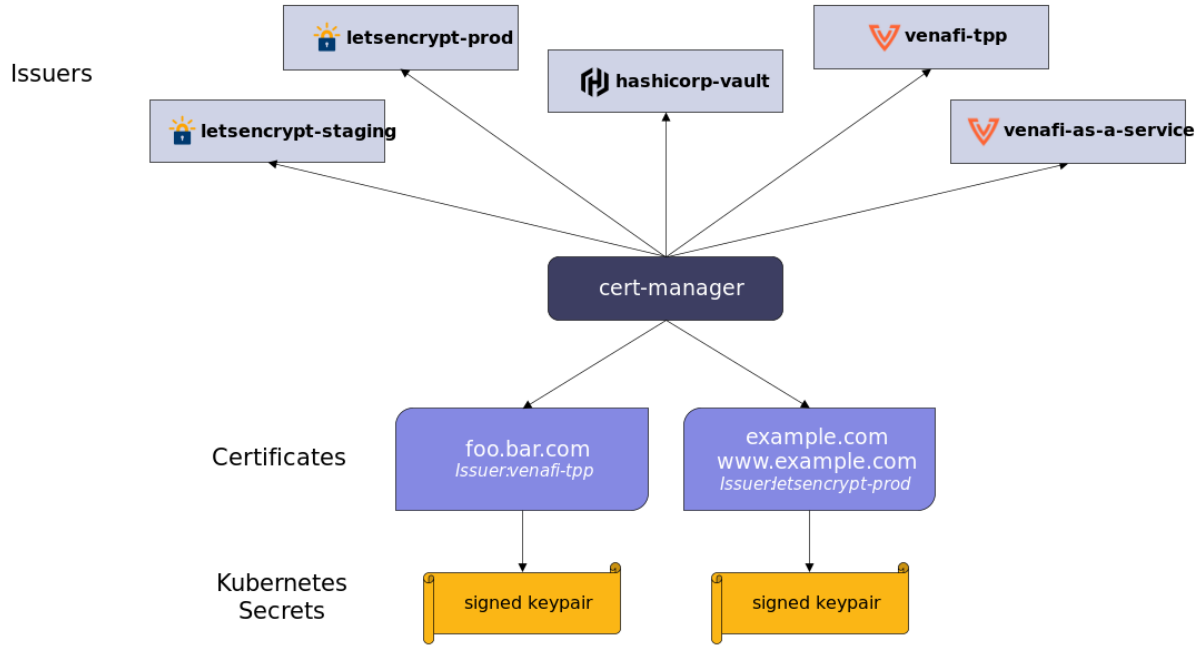

Fig. 2.17: Arhitectură Cert Manager[4]

Acest instrument are ca scop principal conexiunea cu autorități de certificare pentru reînnoirea certificatelor înainte de a expira. Cert-manager contribuie cu o gamă largă de CA-uri care emit certificate de încredere pentru browserele web, cum ar fi: Let's Encrypt[62], Venafi[63] sau Hashicorp[64]. Certificatele obținute de la CA sunt apoi stocate la nivelul clusterului de Kubernetes ca secrete[65] la care are acces doar serviciul cert-manager. Administratorul de sistem nu mai e nevoit să facă niciun fel de procedeu, reînnoirea certificatelor fiind complet automatizată.

Obținerea certificatelor

Certificatul pentru platforma de coaching online este eliberat de CA-ul Let's Encrypt[17], deoarece este o organizație non-profit care furnizează certificate gratuite. De asemenea, reînnoirea certificatelor se face instantaneu, gratuit și este compatibilă cu majoritatea browserelor și dispozitivelor de pe piață.

Pentru ca cert-manager sa poata contribui cu autoritatea de certificare, la nivelul clusterului de Kubernetes trebuie sa declaram intr-un fisier de tip YAML, o resursa de tip ClusterIssuer[66] de tip ACME[3]. La crearea acestei resurse de tip ClusterIssuer, se creează în mod automat un cont unic de tip ACME la nivelul clusterului, prin care ClusterIssuer se conectează la serverul ACME gestionat de

[62] https://letsencrypt.org/
[63] https://venafi.com/
[64] https://www.hashicorp.com/
[65] https://kubernetes.io/docs/concepts/configuration/secret/
[66] https://cert-manager.io/docs/concepts/issuer/

Let's Encrypt. Certificate autorizate de servere ACME publice sunt de incredere pentru dispozitivele clinetilor.

Pentru ca serverul ACME sa verifice ca un client detine domeniul, acesta trebuie sa rezolve una din cele doua provocari furnizate de catre server, si anume:

- HTTP01[67]
- DNS01[68]

Pentru a dovedi autenticitatea domeniului fit-stack.online s-a optat pentru rezolvarea provocării de tip HTTP01. Această provocare constă în următorii pași:

- serverul ACME al organizației Let's Encrypt va trimite un token clientului ACME.
- clientul va stoca pe serverul web token-ul într-un fișier care va fi disponibil prin URL-ul următor: http://<YOUR_DOMAIN>/.well-known/acme-challenge/<TOKEN>[69]. Acest fișier va conține tokenul furnizat de server și o amprentă dată de cheia privată configurată la crearea contului.
- clientul ACME va comunica serverului că este pregătit.
- serverul ACME va încerca să obțină tokenul de la adresa respectivă. Dacă serverul va găsi tokenul corect la nivelul clientului, atunci provocarea se consideră încheiată, altfel va trebui reluată provocarea cu un certificat nou.

După ce parcurgem pașii prezentati anteriori, vom avea o resursă ClusterIssuer care va putea autoriza certificate în numele platformei Let's Encrypt. Următorul pas este configurarea unui fișier YAML prin intermediul căruia va fi creată o resursă de tip Certificate[70], care va avea drept autoritate de certificare resursa de tip ClusterIssuer creată anterior. Imediat după configurarea fișierului YAML pentru Certificate, se va crea automat o resursă de tip CertificateRequest[71], care va solicita un certificat X.509 de la autoritatea de certificare, în cazul nostru, ClusterIssuer.

[67] https://cert-manager.io/docs/configuration/acme/http01/
[68] https://cert-manager.io/docs/configuration/acme/dns01/
[69] https://letsencrypt.org/docs/challenge-types/
[70] https://cert-manager.io/docs/usage/certificate/
[71] https://cert-manager.io/docs/usage/certificaterequest/

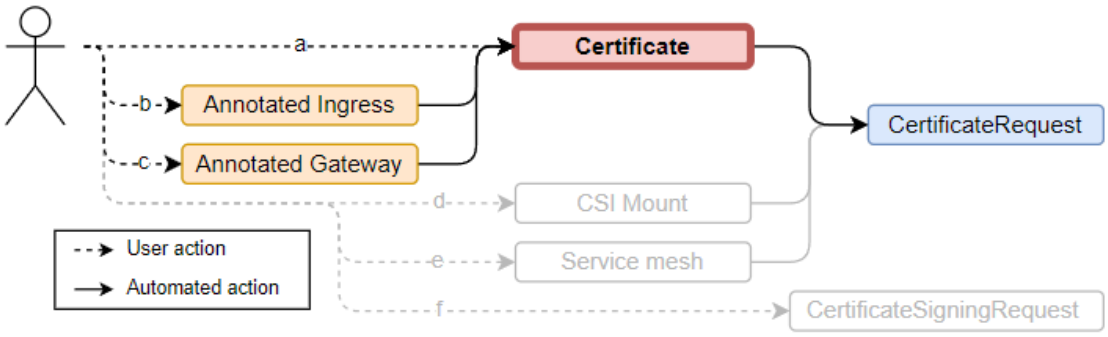

Fig. 2.18: Flux automatizarea certificatelor[72]

Certificatul va fi emis imediat, deoarece ClusterIssuer este o resursă gestionată local. Acesta va fi salvat într-un secret Kubernetes și va fi utilizat de resursa de tip Ingress[73] pentru a facilita traficul HTTPS către platforma de coaching online.

[73] https://kubernetes.io/docs/concepts/services-networking/ingress/

3 Soluții existente

3.1 Soluții existente

Pe piață există o multitudine de platforme dedicate activităților sportive, fiecare având abordări diferite în ceea ce privește fitnessul. Unele se concentrează pe creșterea masei musculare, altele pe exerciții cardio, precum alergatul sau ciclismul, iar unele sunt destinate utilizatorilor care doresc să își îmbunătățească flexibilitatea sau stilul de viață în general printr-o nutriție adecvată. Aceste aplicații sunt unice prin abordarea lor, încercând să acopere în mod exhaustiv o categorie specifică de utilizatori care aleg să adopte acest stil de viață.

În continuare vor fi enumerate o serie de aplicații populare, care s-au remarcat prin caracteristici inovative.

1. **Caliber**[34]: Această aplicație a fost votată ca fiind cea mai bună per ansamblu deoarece integrează toate categoriile de sportivi într-un mod extensiv. Aplicația de bază este gratuită și oferă două planuri premium: primul este un abonament de 19 dolari pe lună care include sesiuni de coaching de grup, iar cel de-al doilea începe de la 200 de dolari pe lună și furnizează programe de fitness personalizate, create de antrenori avizați pentru nevoile clienților. Caliber oferă utilizatorilor peste 500 de exerciții, fiecare având un video demonstrativ și explicații clare. De asemenea, utilizatorii își pot configura propriile antrenamente. Această aplicație primește un scor de 4,68 din 5.

2. **Future**[34]: Această aplicație este foarte apreciată din perspectiva antrenamentelor personalizate, oferind o gamă largă de antrenori disponibili pentru a ghida călătoria în domeniul fitness a utilizatorilor. Future oferă cea mai bună opțiune imediat după a lucra față în față cu un antrenor. Deoarece un antrenor real este responsabil pentru modul de antrenare al clienților, utilizatorii vor avea o încredere mai mare în aplicație, știind că este o persoană avizată care furnizează conținut sportiv și nu un algoritm. Aplicația permite sortarea antrenorilor după stilul și expertiza acestora sau chiar după sporturi specifice precum fotbal, hochei, etc. Aplicația permite

integrarea cu dispozitivele wearable[1], fiind o funcționalitate importantă în timpul antrenamentului. Future prevede o taxă de 199 de dolari pe luna, fiind o opțiune costisitoare, dar care oferă conținut de calitate și antrenori avizați. Această aplicație primește un scor de 4,5 din 5.

3. **iFIT**[34]: Acesta aplicatie este conceputa pentru a se integra cu echipamnetul de tip cardio combatibile cu platforma iFIT. Aplicatia este recunoscuta pentru varietate, intrucat ofera peste 16,000 de antrenamente din care clientii pot alege. aplicatia are antrenamente care pot fi realizate atat acasa cat si sau afara, ghidate de catre un antrenor avizat, primind un scor ridicat de 5 din 5 pentru varietate. Aplicatia prevede doua planuri de plata, primul fiind de 15 dolari pentru un cont individual, iar cel de-al doilea este 39 de dolari pentru un plan de familie. Aplicatia iFIT primeste scorul per total de 4.2 din 5.

4. **Nike Training Club**[34]: Nike Training Club este recunoscută drept cea mai bună aplicație gratuită dedicată antrenamentelor cu greutăți, având ca scop creșterea în forță a utilizatorilor. Aceasta oferă o multitudine de funcționalități valoroase pentru o aplicație gratuită, cum ar fi clase de fitness în timp real, programe de antrenament pentru diverse scopuri și antrenori avizați. Aplicația include antrenamente în care utilizatorii își pot folosi doar greutatea corporală, în cazul în care nu dispun de echipamentul necesar. Nike Training Club primește o evaluare de 4 din 5, apropiată de scorurile acordate platformelor care solicită taxe, ceea ce o face ca o alternativă foarte buna pentru o gamă mai largă de utilizatori.

5. **CENTR**[34]: Această aplicație nu oferă doar unele dintre cele mai bune opțiuni pentru antrenamente destinate forței și creșterii masei musculare, ci și furnizează planuri nutriționale cât mai eficiente pentru a atinge aportul caloric din surse sănătoase și bogate în proteine. Această aplicație abordează o viziune holistică asupra întregului proces de a deveni mai puternic, bazată pe principii sănătoase, atât din punct de vedere al antrenamentelor, cât și al nutriției. Aplicația permite jurnalizarea exercițiilor, pe baza căruia va maximiza progresul sugerând greutăți din ce în ce mai mari. CENTR va genera liste de cumpărături pe baza felurilor de mâncare selectate de utilizator. Aplicația oferă varietate din perspectiva exercițiilor care pot fi făcute acasă, cu greutatea corporală sau în sălile de fitness. Aplicația prevede o taxă de 29.99 dolari pe lună și la început o perioadă gratuită de 7 zile.

6. **JuggernautAI**[34]: Această aplicație oferă programe de antrenament de

[1] https://en.wikipedia.org/wiki/Wearable_technology

tip powerlifting[2] generate de un model de inteligență artificială. JuggernautAI nu dispune de antrenori, ci lasă munca în mâinile modelului de inteligență artificială care va sugera noi antrenamente de powerlifting pe baza unui formular completat de utilizator la înregistrarea în aplicație. Scopul aplicației este creșterea în forță și maximizarea greutății pentru cele trei exerciții esențiale din powerlifting: genoflexiuni, împins din culcat și îndreptări. Modelul de inteligență artificială va crea programe de powerlifting pe baza statisticilor utilizatorilor. Aplicația presupune că utilizatorii dispun de toate echipamentele necesare pentru a completa antrenamentele de powerlifting, fiind o soluție axată pe o categorie restrânsă de utilizatori. Aplicația are o taxă de 35 de dolari pe lună și oferă 14 zile gratuite la începutul abonamentului. Aplicația JuggernautAI este evaluată la 4 din 5.

7. **Aaptiv**[34]: Această aplicație oferă o gamă largă de tipuri de exerciții precum cardio, forță, yoga, mobilitate și pilates, cu foarte puțin echipament necesar. Aplicația este potrivită pentru începători deoarece oferă o gamă largă de exerciții, care nu sunt potrivite pentru o persoană de nivel intermediar sau avansat. Aaptiv, pe lângă antrenamentele de forță care nu necesită echipament sofisticat, are și moduri adaptate pentru antrenamente de alergare ce pot fi executate în aer liber. Aplicația prevede o taxă de 14.99 de dolari pe lună și o perioadă gratuită de 7 zile. Aplicația este evaluată la scorul 4.14 din 5.

8. **Joggo**[34]: Alergatul poate fi un sport intimidant pentru o persoană, astfel Joggo își propune maximizarea progresului utilizatorilor, fie că sunt începători sau avansați. Pentru a putea începe este nevoie doar de o pereche de adidași adecvați și Joggo va fi companion de încredere în maximizarea anduranței. Joggo va crea un plan personalizat, în urma completării unui formular ce conține întrebări create pentru a cunoaște scopurile utilizatorilor. De asemenea, Joggo va oferi ajutor și pe plan nutrițional, deoarece oferă rețete pentru feluri de mâncare sănătoase, permite jurnalizarea caloriilor consumate, ținând cont și de macro-nutrienți. Aplicația prevede o taxă de 33 de dolari pe lună și este evaluată la nota 3.3 din 5.

9. **Beyond The Whiteboard**[34]: Aplicatia este orinetata catre categoria de exercitii de tip CrossFit[3]. Beyond The Whiteboard se remarca prin mecanismele exhaustive de jurnalizare a datelor rezultate in urma antrenamentelor, furnizand statistici, metrici si date despre recordurile person-

[2] https://en.wikipedia.org/wiki/Powerlifting

[3] https://en.wikipedia.org/wiki/CrossFit

ale care au ca scop o analiza profunda a progresului utilizatorilor. De asemenea aplicatia ofera un set mare de exercitii pentru mai multe nivele de dificultate. APlicatia prevede o taxa de 7.99 de dolari pe luna si o perioada de gratuitat de 30 de zile. Beyond The Whiteboard este evaluata la 4.43 din 5.

10. **Down Dog**[34]: Down Dog este una dintre cele mai apreciate aplicații pentru antrenamentele de yoga, oferind peste 60.000 de exerciții personalizate. Aplicația este foarte personalizabilă, permițând utilizatorilor să aleagă dintr-o gamă variată de categorii de muzică, instructori, durată a sesiunii și părți ale corpului asupra cărora doresc să se concentreze. De asemenea, permite vizualizarea unui istoric pentru a putea vedea exercițiile completate în trecut. Aplicația are o taxă de 7.99 dolari pe lună și este evaluată la scorul 3.6 din 5.

3.2 Detalii financiare

După cum se poate observa, aplicațiile enumerate în subcapitolul anterior variază destul de mult din punct de vedere financiar, oferind atât planuri gratuite, freemium[4], dar și unele care prevăd taxe ce pot ajunge la sume considerabile, de până la 200 de dolari. Prețul vine, de asemenea, cu o garanție a calității și a încrederii, deoarece acele aplicații dispun de funcționalități în care este implicată și resursa umană, și anume antrenori care garantează antrenamente avizate și testate de-a lungul timpului.

Desigur, deși prețurile implicate de unele aplicații pot părea considerabile, aceste servicii oferă mult mai multe beneficii decât o simplă sesiune în persoană cu un antrenor. Aceste platforme simplifică munca pentru antrenori și oferă un serviciu apropiat de cel față în față pentru clienți, furnizând și numeroase alte date adiționale precum statistici și recomandări pentru noi exerciții.

Mecanismul principal de creștere a profitului pentru organizațiile ce dezvoltă aceste aplicații este o taxă de tip abonament, în urma căreia clienții vor avea parte de conținut special personalizat pentru nevoile lor. O altă soluție este abordarea de tip freemium, în care clienții au acces la anumite funcționalități de bază ale aplicației, dar există opțiunea de cont premium care adaugă noi caracteristici.

Platforma de coaching online Fitstack se ghidează după abordarea de tip freemium. Prin aderarea la un abonament, utilizatorul va avea acces la un cont premium și va putea vizualiza conținut sportiv adăugat de antrenor, cu care va putea comunica prin serviciul de mesagerie în cazul în care există nelămuriri.

[4] https://en.wikipedia.org/wiki/Freemium

3.3 Studiu comparativ al soluțiilor populare existente

Când vine vorba despre un studiu comparativ al soluțiilor populare existente, trebuie luate în considerare toate punctele cheie pe care o aplicație ideală ar trebui să le aibă. Un aspect important este faptul că aceste aplicații au caracteristici foarte diferite, fiecare având ca scop satisfacția publicului țintă pe care și-l propune. Deși o aplicație care satisface cât mai multe categorii de oameni ar putea fi considerată mai profitabilă, trebuie să aducem în discuție și calitatea serviciilor pe care le furnizează. O aplicație dedicată unei categorii specifice de sportivi ar putea aduce rezultate mult mai bune decât o platformă care încearcă să acopere cât mai multe categorii de atleți.

În următorul tabel3.1, voi evidenția principalele caracteristici esențiale pe care ar trebui să le aibă o platformă de fitness, împreună cu avantajele și dezavantajele acestora, din următoarele puncte de vedere:

- Preț: prețul pentru abonament
- Tendință spre progres: o aplicație de fitness ar trebui să aibă ca scop principal progresul utilizatorilor
- User Experience: interfața utilizatorilor și experiența acestora în cadrul aplicației
- Echipament necesar: o aplicație de fitness ar trebui să furnizeze antrenamente care nu necesită echipament sofisticat
- Implicare: un antrenor și un sistem de notificări potrivit ar putea mări nivelul de implicare al utilizatorilor
- Monitorizarea progresului: o aplicație de fitness trebuie să monitorizeze progresul utilizatorului
- Variația antrenamentelor: un antrenament poate să facă parte din mai multe categorii precum: cardio, forță, powerlifting, yoga, etc.
- Nutriție: nutriția este foarte importantă în ceea ce privește stilul de viață al sportivilor
- Scor total

Pentru a aprecia nivelul fiecărei caracteristici pentru aplicațiile analizate voi folosi cuvintele: ”Minim”, ”Scăzut”, ”Mediu”, ”Ridicat”, ”Maxim”, mai puțin pentru categoria de scor.

	Preț	Tendință progres	User experience	Echip. necesar	Implicare	Monit. progres	Variație exerciții	Nutriție	Scor
Caliber	Maxim	Maxim	Maxim	Scăzut	Maxim	Ridicat	Maxim	Minim	4.68
Future	Maxim	Maxim	Maxim	Scăzut	Maxim	Mediu	Maxim	Minim	4.3
IFIT	Scăzut	Mediu	Maxim	Scăzut	Mediu	Mediu	Maxim	Minim	4.2
Nike Training Club	Minim	Mediu	Maxim	Scăzut	Mediu	Minim	Maxim	Minim	4.16
CENTR	Mediu	Maxim	Maxim	Scăzut	Ridicat	Ridicat	Mediu	Maxim	3.9
Juggernaut AI	Mediu	Maxim	Maxim	Ridicat	Maxim	Maxim	Minim	Minim	4
Aaptiv	Scăzut	Mediu	Maxim	Minim	Ridicat	Mediu	Ridicat	Minim	4.13
Joggo	Mediu	Maxim	Mediu	Minim	Mediu	Mediu	Minim	Mediu	3.3
Beyond The Whiteboard	Scăzut	Maxim	Ridicat	Mediu	Mediu	Maxim	Ridicat	Minim	4.43
Down Dog	Scăzut	Mediu	Maxim	Minim	Minim	Scăzut	Scăzut	Minim	3.8

Tabel 3.1: Studiu comparativ al soluțiilor populare existente

4 Descrierea implementării

Platforma de coaching online are ca scop nu doar îmbunătățirea experienței sportive a utilizatorilor, ci și crearea unei conexiuni puternice între antrenori și clienții lor. Acest lucru implică proiectarea unei aplicații care să poată satisface aceste cerințe la cel mai înalt nivel de calitate.

4.1 Cazuri de utilizare

Platforma de coaching online găzduiește patru tipuri de uitlizatori:

1. Utilizator cu cont standard
2. Utlizator cu cont premium
3. Antrenor
4. Administrator

În următoarele subcapitole voi evidenția cazurile de utilizare pentru fiecare tip de utilizator, prin intermediul unor diagrame UML.

4.1.1 Utilizator cu cont standard și cont premium

Figura 4.1 surprinde principalele scenarii de utilizare ale platformei atât în cazul utilizatorilor cu cont standard, cât și a celor cu cont premium.

Fig. 4.1: Cazuri de utilizare pentru client mobile

4.1.2 Antrenor și administrator

În diagrama UML4.2 sunt prezentate cazurile de utilizare ale aplicației pentru antrenori sau administratori.

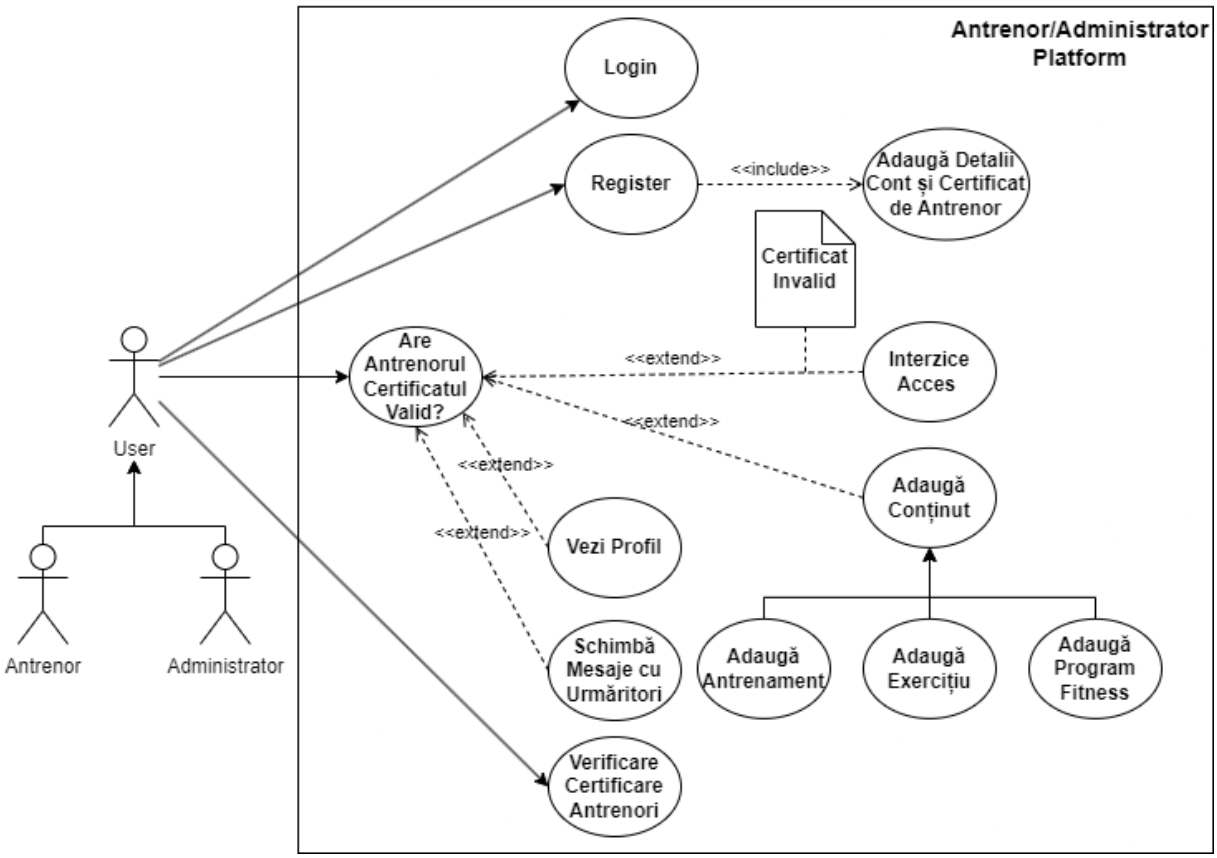

Fig. 4.2: Cazuri de utilizare pentru antrenor/administrator

4.2 Cerințe funcționale

4.2.1 Aplicatia mobile pentru clienti

În platforma de coaching online Fitstack, dedicată clienților mobile, sunt prezente următoarele cerințe funcționale:

- **Autentificare și înregistrare**: aplicația permite clienților înregistrarea prin mai multe modalități. O primă metodă este înregistrarea pe baza completării unui formular unde utilizatorul va trebui să furnizeze detalii precum: nume, prenume, email, parolă, parolă repetată. Cea de-a doua metodă este înregistrarea pe baza contului Google. În momentul când utilizatorul va alege această opțiune, Google va furniza un bilet de autentificare OAuth2.0 ceea ce va dovedi aplicației autenticitatea persoanei, urmând să fie redirecționat către aplicație. În ambele cazuri vor fi furnizate

tichete JWT, în care va fi stocată adresa de email, rolul utilizatorului și permisiunile acestuia la resursele aplicației.

- **Resetare Parolă**: Dacă utilizatorul și-a uitat parola, acesta va putea completa un formular de resetare în care va furniza adresa de email cu care s-a înregistrat în platformă. În cazul în care utilizatorul este înregistrat în platformă cu acea adresă de email, va primi un email care conține un link către o pagină web unde va trebui să introducă o parolă nouă. După completarea noului formular, parola va fi resetată, iar utilizatorul va putea accesa din nou contul.

- **Inițiere sesiune de antrenament**: Utilizatorii pot începe antrenamente fie create de administrator, fie create de antrenorii urmăriți, dacă au cont premium. Aceste antrenamente reprezintă o serie de exerciții pe care clientul va trebui să le îndeplinească într-o anumită ordine. Utilizatorul va putea alege greutatea, numărul de repetări și numărul de seturi, în funcție de tipul exercițiului. În cadrul exercițiului sunt menționate detalii cu privire la grupa de mușchi, categoria, echipamentul și nivelul de greutate al acestuia. Pentru fiecare exercițiu sunt furnizate poze în care este evidențiată poziția de început sau de final, cu scopul de a oferi clienților o explicație mai detaliată. Antrenorii pot adăuga și un filmuleț scurt în care vor explica modul corect de realizare a exercițiului. De asemenea, clienții pot folosi temporizatorul pentru a putea înregistra durata de parcurgere a exercițiului. În cazul exercițiilor de tip cardio, numărul de calorii va fi calculat în funcție de durata acestuia de către modelul de inteligență artificială bazat pe regresie neliniară. Seturile se execută rând pe rând până la ultimul, caz în care se va trece la următorul exercițiu. În cazul în care utilizatorul se află la ultimul set din cadrul ultimului exercițiu, după încheierea acestuia va avea opțiunea de terminare a antrenamentului. Pe parcursul antrenamentului, toate detaliile stabilite de client (număr de repetări, greutate, timp, etc.) vor fi salvate în istoric pentru calcularea statisticilor.

- **Continuare sesiune de antrenament**: În cazul unui factor extern care determină utilizatorul să întrerupă parcurgerea antrenamentului, acesta va putea salva starea curentă ieșind din ecranul exercițiului prin apăsarea butonului reprezentat printr-o săgeată ”←” în colțul din stânga sus al ecranului. Întrucât toate antrenamentele sunt salvate în istoric, un utilizator poate relua execuția unei sesiuni. Antrenamentul va fi reluat de la ultimul exercițiu făcut.

- **Inițiere program de fitness**: Clienții aplicației pot începe un program de fitness care constă într-o listă de antrenamente. Programele sunt constituite

fie din antrenamente, fie din zile de odihnă. Această ordine este stabilită de către antrenor sau de către administrator. Aplicația va ține cont de antrenamentul la care a rămas ultima dată utilizatorul, marcându-le cu culoarea verde pe cele completate până în momentul curent.

- **Calcularea Statisticilor**: În urma fiecărui antrenament vor fi calculate statistici după mai multe criterii, și anume: timpul total, caloriile arse, volumul total (volum = număr de repetări * greutate). De asemenea, vor fi incluse diagrame circulare cu categoriile, grupurile de mușchi lucrate și nivelurile de dificultate din cadrul exercițiilor pentru o vizualizare procentuală a acestor date. Totodată, pentru a pune în perspectivă cu antrenamentele anterioare, se vor genera grafice pentru vizualizarea progresului în ceea ce privește timpul, volumul și numărul de calorii arse în cadrul antrenamentelor.

- **Deservirea premiilor și calcularea punctajului**: Acesta este un aspect important în ceea ce privește caracteristica de gamification a aplicației. Utilizatorii vor primi premii în urma îndeplinirii antrenamentelor ce au rolul de a oferi clientului satisfacție. Aceste premii sunt oferite în cazul în care un client depășește un anumit număr de exerciții, antrenamente sau programe de fitness. De asemenea, la finalul antrenamentului se calculează un număr de puncte după următoarea formulă:

$$\text{număr\_puncte} = \sum_{n=1}^{n} (p \times \text{volum}(e_i) + \frac{p}{2} \times \text{timp}(e_i) + p \times \text{număr\_calorii}(e_i)) \tag{4.1}$$

  În formula anterioară, p reprezintă o pondere care este egală cu 0.1 și e_i reprezintă exercițiul cu indicele i din cadrul antrenamentului. La punctaj se adaugă câte 200 de puncte în cazul în care utilizatorul a obținut un premiu pe parcursul antrenamentului.

- **Vizualizare clasament**: O altă caracteristică importantă de gamification o reprezintă noțiunea de clasament. În cadrul aplicației există două clasamente. Primul este unul global în care sunt sortați descrescător toți utilizatorii platformei de coaching după numărul de puncte obținute în urma realizării exercițiilor, antrenamentelor și programelor de fitness. Cel de-al doilea clasament poate fi accesat doar dacă utilizatorul a optat pentru înregistrarea cu contul Google, deoarece va conține toate contactele Google, inclusiv cele din secțiunea ”Others”. Acest lucru este posibil prin integrarea People API, furnizat de platforma Google. Metodele de clasificare sunt identice cu cele menționate pentru primul clasament.

- **Filtrare antrenamente și programe**: Alegerea unui antrenament sau a unui program potrivit pentru utilizator poate fi o sarcină dificilă. O metodă prin care această căutare poate fi ușurată este prin intermediul unui filtru, prin care utilizatorul poate specifica anumite cuvinte cheie. De asemenea, utilizatorul poate specifica un interval de dificultate, de la 1 (nivel începător) până la 3 (nivel expert).
- **Creaza propriul antrenament**: Un utilizator al platformei poate adauga un antrenament creat de el prin completarea unui formular in care trebuie adaugat: numele exercitiului, descriere scurta si o lista de exercitii care vor constitui antrenamentul. Acestia vor putea alege fie exrcitii create de administrator, fie exercitii create de antrenorii pe care ii urmareste utilizatorul. Antrenamentul va aparea la sectiunea ”My workouts”.
- **Creare cont premium**: Clienții cu cont standard pot opta pentru un abonament premium, devenind astfel utilizatori plătitori. Aceștia vor avea acces la exercițiile, antrenamentele și programele de fitness ale antrenorilor, precum și la serviciul de trimitere a mesajelor. Plata este realizată prin integrarea cu Gateway-ul furnizat de PayPal. În cadrul platformei PayPal a fost creat un obiect de tip ”Subscription” care reprezintă o plată recurentă, fiind configurată perioada de o lună și suma de zece dolari.
- **Urmărire antrenori**: Un utilizator care are cont premium va putea urmări antrenorii, având astfel acces la conținutul postat de aceștia: exerciții, antrenamente și programe de fitness. Clienții vor putea accesa pagina de profil a antrenorilor, unde va fi listat atât conținutul adăugat de aceștia, cât și anumite detalii legate de numărul de urmăritori, exerciții, antrenamente și programe fitness. De asemenea, pagina de profil va avea opțiunea de urmărire sau oprire din urmărire și opțiunea de a trimite un mesaj antrenorului.
- **Schimb mesaje**: Un utilizator ce dispune de un cont premium va avea posibilitatea de a comunica cu antrenorii pe care îi urmărește. Aceștia pot apela la antrenori în cazul în care au întrebări în legătură cu conținutul adăugat. Serviciul de mesagerie este unul în timp real, fiind proiectat să folosească biblioteca Socket.IO, specializată pentru astfel de cerințe. Mesajele vor fi salvate într-o bază de date pentru a putea reda istoricul conversației.

4.2.2 Aplicația pentru antrenori și administratori

În platforma de coaching pentru antrenori și administratori sunt implementate următoarele cerințe funcționale:

- **Autentificare și înregistrare**: În aplicația dedicată antrenorilor, aceștia se pot autentifica pe baza emailului și a parolei stabilite la înregistrare. Procesul de înregistrare în platformă este format din doi pași esențiali, și anume: completarea primului formular de înregistrare unde utilizatorul va trebui să furnizeze date precum nume, prenume, email, parolă (care trebuie să conțină cel puțin o literă mare, două caractere speciale și două cifre); al doilea pas constă în furnizarea detaliilor specifice antrenorilor: număr de ani de experiență, tipul certificării de antrenor și o poză cu certificatul de antrenor. Acesta va trebui să aștepte ca administratorul să valideze certificatul de antrenor. În urma înregistrării și a autentificării va fi emis un tichet JWT, care va conține emailul utilizatorului, rolul și permisiunile acestuia în cadrul sistemului.

- **Adăugare exercițiu**: Atât administratorii, cât și antrenorii au posibilitatea de a adăuga exerciții, prin completarea unui formular în care vor trebui să furnizeze detalii precum: numele, descrierea, categoria, grupa de mușchi, echipamentul necesar, nivelul de dificultate și două poze în care vor fi surprinse pozițiile de start și final ale exercițiului. Opțional, aceștia pot furniza și un videoclip explicativ în care vor demonstra modul corect de execuție. Salvarea videoclipului este optimizată astfel încât acesta să fie stocat pe server; videoclipul va fi trimis de la aplicația clientului în bucăți, urmând să fie reasamblat când va ajunge ultima parte a acestuia.

- **Adăugare antrenament**: Administratorul și antrenorii pot adăuga antrenamente prin completarea unui formular care va consta în următoarele câmpuri: nume, descriere, lista exercițiilor ce constituie antrenamentul și poza de copertă. Adăugarea exercițiilor potrivite în cadrul antrenamentului poate fi o sarcină dificilă pentru antrenor, astfel platforma de coaching online furnizează o serie de filtre speciale care vor ajuta utilizatorii să aleagă exercițiile în funcție de categorie, grupă de mușchi, echipamentul necesar și nivelul de dificultate. De asemenea, filtrul va putea fi folosit atât pentru a căuta după numele exercițiului, cât și pentru a alege dacă antrenorul dorește să vadă doar exercițiile lui sau doar exercițiile propuse de administrator. Dificultatea antrenamentului va fi calculată automat, în funcție de nivelul de dificultate al exercițiilor din cadrul acestuia, prin calcularea unei medii aritmetice a dificultăților. Astfel, exercițiile pentru începători vor avea valoarea 1, cele pentru avansați vor avea valoarea 2, iar cele pentru experți vor avea valoarea 3.

- **Adăugare program fitness**: În cadrul aplicației de coaching online, utilizatorii pot adăuga programe de fitness prin completarea unui formular în care vor fi furnizate următoarele detalii: nume, descriere, durata pro-

gramului în zile, lista de antrenamente ce constituie programul și o poză de copertă. Utilizatorul poate folosi anumite filtre pentru o căutare mai ușoară a antrenamentelor. Filtrele fac posibilă căutarea antrenamentelor în funcție de nume, nivel minim și maxim de dificultate admis. Totodată, antrenorul poate menționa dacă dorește să fie afișate doar antrenamentele personale sau și cele adăugate de administrator. Adăugarea se va realiza prin funcționalitatea de "drag and drop" în căsuțele furnizate pentru fiecare zi din cadrul programului. Dacă utilizatorul nu va adăuga un antrenament într-o căsuță, acea zi va fi considerată zi de odihnă pentru clienți.

- **Vizualizare profil**: Antrenorii vor avea posibilitatea de a naviga catre pagina de profil unde vor fi listate toate exercitiile, antrenamentele, programele de fitness si certificatele de antrenor ale acestora. De asemenea acestia vor putea actiona asupra exercitiilor, antrenamentelor si programelor, avand posibilitatea de a sterge sau a vizualiza in mod detaliat continutul lor.

- **Verificare certificare antrenori**: Administratorii sunt responsabili cu validarea certificatelor pentru a putea oferi antrenorilor posibilitatea de a adăuga conținut în platforma de coaching. Administratorul poate mări poza în fiecare regiune a ei, astfel încât să poată analiza în detaliu fiecare câmp. Acesta are opțiunea de a valida sau invalida o cerere.

- **Schimb de mesaje**: Antrenorii vor putea comunica cu utilizatorii cu cont premium din platforma mobilă, cu scopul de a răspunde la întrebările acestora în legătură cu conținutul adăugat. Modulul de mesagerie funcționează pe baza bibliotecii Socket.IO pentru transferul mesajelor în timp real în aplicație. Aceste mesaje vor fi salvate într-o bază de date pentru a persista în timp.

4.3 Cerințe nefuncționale

Cerințele nefuncționale sunt elemente esențiale ale unei aplicații software, care nu se referă direct la funcționalitatea pe care o oferă utilizatorului final. În schimb, se referă la caracteristici care afectează performanța, securitatea, utilizabilitatea și alte calități importante ale sistemului.

Platforma de coaching online include urmatoarele cerinte software nefunctionale:

- **Securitate**: Platforma FitStack oferă utilizatorilor un mediu securizat în care pot avea încredere, deoarece datele acestora sunt protejate. Un prim aspect legat de securitate este autentificarea și autorizarea, reprezentând elemente esențiale în cadrul unei aplicații. Autentificarea se realizează pe

baza unor tichete JWT, adăugate la antetul de autentificare al cererii HTTP. Aceste tichete conțin adresa de email, rolul și permisiunile utilizatorului, dictând accesibilitatea la resursele sistemului. Comunicarea între client și server este criptată prin intermediul HTTPS, prevenind interceptarea și manipularea datelor de către un atac de tip Man In The Middle. Parolele utilizatorilor sunt hash-uite folosind algoritmul bcrypt și un salt unic pentru fiecare parolă. Platforma Azure aduce, de asemenea, un plus de securitate prin introducerea unor mecanisme importante de protecție, cum ar fi Azure Distributed Denial of Service (DDoS) și Azure Web Application Firewall (WAF), asigurând astfel protecția aplicației împotriva atacurilor externe.

- **Performanța**: Platforma de coaching online deserveste utilizatori în mod eficient și rapid, asigurându-se că poate gestiona un număr considerabil de clienți fără a compromite drastic fiabilitatea sau viteza de răspuns. Pentru a asigura o performanță ridicată, aplicația FitStack este optimizată în acest sens. Arhitectura de tip microservicii aduce beneficii de performanță prin împărțirea volumului de lucru, fiind capabilă de gestionarea unui număr mai mare de utilizatori, într-un timp mai scurt.

- **Scalabilitate**: Platforma Fitstack trebuie să scaleze în funcție de numărul utilizatorilor care vor să acceseze aplicația. Arhitectura aplicației este bazată pe microservicii, împărțind astfel volumul de lucru în mod egal între serviciile sistemului. Folosirea clusterului de Kubernetes furnizat de platforma Azure are implicații majore în ceea ce privește scalabilitatea, întrucât un volum mare de lucru care determină suprasolicitarea atât a memoriei, cât și a procesorului va declanșa în mod automat adăugarea de noduri adiționale în cluster, fiind astfel capabilă de gestionarea unui număr mai mare de utilizatori.

- **Reziliența**: Această cerință nefuncțională se referă la capacitatea sistemului de a recupera starea de funcționare în urma unei erori sau condiții neprevăzute. Platforma Kubernetes joacă un rol esențial în reziliența sistemului, deoarece maximizează timpul de funcționare prin asigurarea faptului că toate podurile sunt în picioare. Kubernetes gestionează automat distribuția volumului de lucru și poate redirecționa traficul către podurile sănătoase în cazul apariției unor erori neprevăzute.

- **Compatibilitate**: Aplicația de coaching online trebuie să fie disponibilă pentru cele mai folosite platforme și sisteme de operare, precum Android și iOS. Aplicația destinată antrenorilor și administratorilor va fi disponibilă doar ca aplicație web. Aceasta este o cerință importantă, deoarece se dorește utilizarea aplicației de către un număr cât mai mare de utilizatori.

- **Utilizabilitate**: Aplicația FitStack trebuie să fie ușor de utilizat și intuitivă pentru utilizatori de toate nivelurile de experiență. Deoarece interfața aplicației a fost concepută folosind principiile de design UX (User Experience), utilizatorii pot naviga și interacționa cu ușurință cu diferitele funcționalități. Utilizatorii pot accesa funcționalitățile importante folosind doar câteva clicuri sau gesturi tactile, iar în timpul interacțiunii cu aplicația, ei primesc feedback prompt și clar pentru a-i ajuta să se orienteze și să fie informați în mod corespunzător.

- **Fiabilitate**: Platforma de coaching online oferă utilizatorilor un timp de funcționare ridicat, prin intermediul platformei Azure, care asigură un uptime de cel puțin 99.5% pentru toate resursele furnizate de ei. De asemenea, Platforma Kubernetes asigură fiabilitatea prin resetarea rapidă a podurilor în cazul unor erori neprevăzute, garantând maximizarea timpului de funcționare.

- **Monitorizare**: Platforma de coaching online folosește mecanisme de monitorizare avansate pentru asigurarea tratării evenimentelor neprevăzute cât mai rapid. Pipeline-ul de CI/CD este gestionat de platforma GitHub, prin intermediul GitHub Actions, oferind o interfață intuitivă, care semnalează erorile apărute pe parcursul procesului de integrare continuă. De asemenea, în cazul unei erori într-o etapă a pipeline-ului, GitHub Actions va trimite o notificare prin email către dezvoltatorul responsabil de repozitoriul de GitHub. Platforma Azure permite integrarea cu cele mai populare sisteme de monitorizare externă, precum Prometheus și Grafana, monitorizând în timp real performanța și sănătatea clusterului de Kubernetes. Aceste tehnologii asigură notificarea instantanee a dezvoltatorilor pentru a trata erorile apărute.

- **Mentenanță**: Această cerință nefuncțională se poate exprima ca probabilitatea ca o componentă să poată fi reparată într-o perioadă definită de timp. Platforma de coaching online își propune maximizarea acestei probabilități, prin utilizarea unor tehnologii moderne și robuste. Procesul de livrare continuă folosește unealta FluxCD, prin intermediul căreia se poate versiona infrastructura. Dezvoltatorii pot opta oricând pentru o versiune anterioară a aplicației în cazul unei erori cu infrastructura curentă. Această posibilitate oferă dezvoltatorilor siguranța și opțiunea de a dezvolta infrastructura având un plan de rezervă în cazul unui eveniment neașteptat.

4.4 Arhitectura soluției

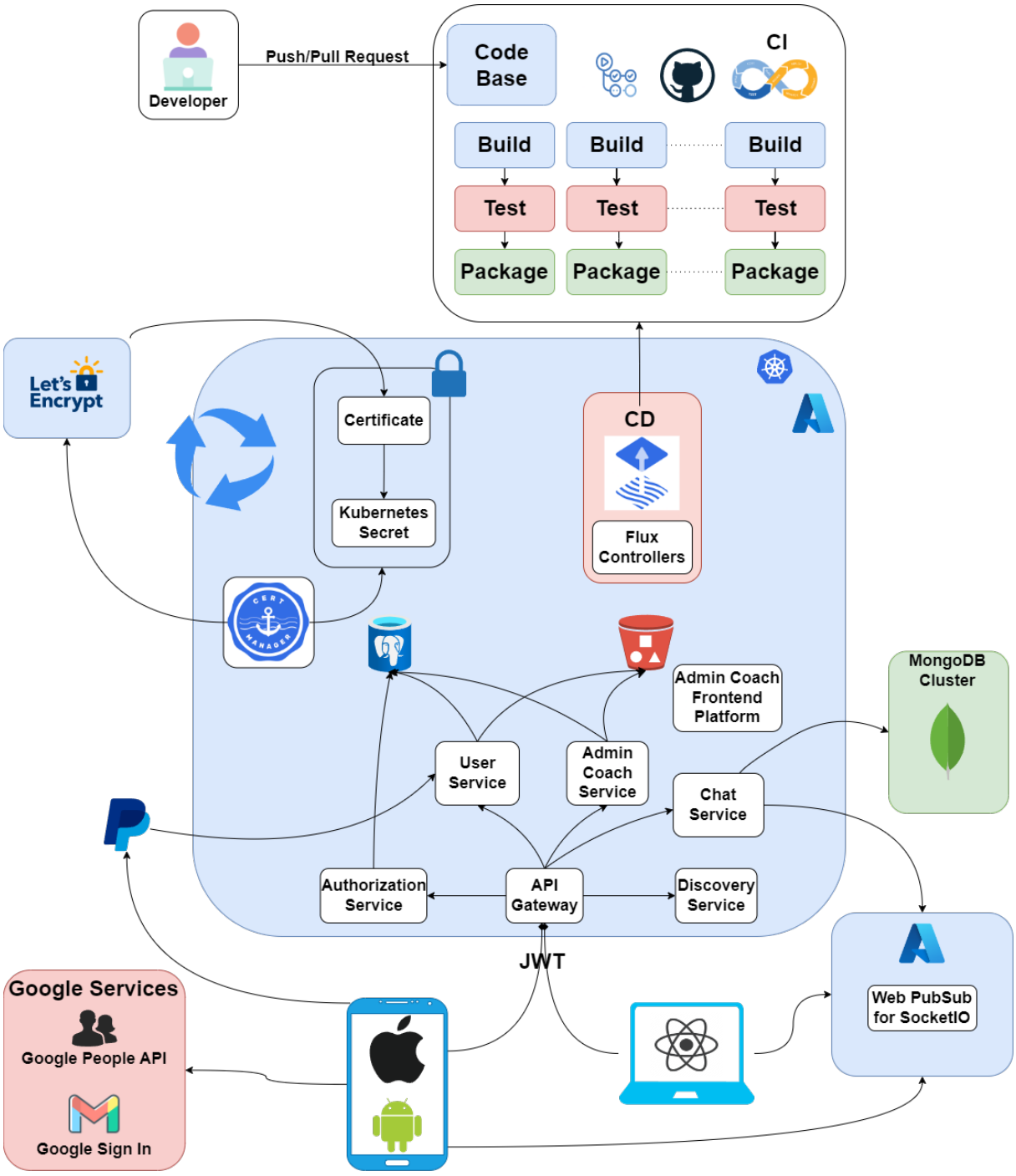

Fig. 4.3: Arhitectura Generală a Sistemului

Acest subcapitol prezintă caracteristicile arhitecturale principale ale aplicației de coaching online, FitStack. Platformă se concentrează pe furnizarea rapidă a datelor către clienți și pe toleranța erorilor prin utilizarea unor framework-uri moderne și robuste, creând un mediu real de producție.

Arhitectura aplicatiei are in vedere opt mari componente principale:

1. **Frontend**: aplicațiile clienților și ale antrenorilor

2. **Backend**: serviciile care facilitează datele către clienți
3. **Serviciul de trimitere a mesajelor**: modulul care faciliteaza comunicarea între antrenori și uilizatori
4. **Deploy**: mediul în cloud în care rulează serviciile
5. **Servicii Google**: serviciile externe aplicației
6. **Serviciul Paypal**: folosit pentru monetizare
7. **Pipeline CI/CD**: automatizarea procesului de deploy atunci când există schimbări în codul sursă
8. **Modulul de automatizare al certificatelor**: responsabil de gestionarea ciclului de viață al certificatelor

4.5 Descrierea modulelor componente

4.5.1 Aplicația mobile a clienților

Aplicația mobile este dezvoltată prin intermediul framework-ului Flutter și este disponibilă pentru sistemele de operare Android și iOS. Acest framework este foarte potrivit pentru o platformă dedicată sportului, deoarece clienții își doresc o aplicație rapidă, fluentă, ușor de folosit și cu un design simplist, dar totodată plăcut din punct de vedere vizual. Dart este un limbaj orientat pe obiect cu o structură simplă și ușor de înțeles, care servește ca fundament al frameworkului Flutter.

Utilizatorii beneficiază de o aplicație care le permite să execute exerciții, antrenamente și programe de fitness, având de asemenea acces la tracking-ul progresului prin intermediul unor statistici clare și bine definite. Conceptul de gamification constă în folosirea unor elemente care în mod obișnuit ar fi întâlnite în jocurile video, precum clasamente, recompense și puncte bonus. Aceste elemente, pe termen lung, vor contribui la maximizarea performanțelor și vor aduce o notă jovială aplicației. De asemenea, aplicația se integrează cu modul de plată, facilitând accesul la antrenorii platformei. Toate aceste elemente sunt prezentate utilizatorilor printr-o interfață interactivă, simplă de utilizat și care respectă convențiile UI/UX.

4.5.2 Aplicația antrenorilor și a administratorilor

Platforma pentru antrenori și administratori este implementată cu ajutorul bibliotecii React.js, utilizând limbajul de programare JavaScript. Aceasta este

o bibliotecă robustă, apreciată datorită arhitecturii sale bazate pe componente reutilizabile, care oferă modularizare și mentenanță crescută a codului, mărind productivitatea dezvoltatorilor.

Aplicația pentru antrenori și administratori permite adăugarea conținutului sportiv, precum exerciții, antrenamente și programe de fitness, într-o manieră optimizată și eficientizată. Platforma include și un serviciu de mesagerie pentru comunicarea cu clienții, facilitând răspunsurile rapide la întrebările legate de conținutul adăugat. De asemenea, această platformă permite administratorilor să aprobe sau să dezaprobe cererile antrenorilor de a se înscrie în platformă, asigurând astfel accesul doar personalului acreditat.

4.5.3 Serviciile de backend

Soluția creată pentru componenta de backend se bazează pe un pattern arhitectural de tip microservicii, facilitând astfel împărțirea încărcăturii de date în mod eficient. Backend-ul este proiectat pentru a gestiona un număr considerabil de utilizatori, menținând în același timp standardele de securitate online.

Serviciile dezvoltate pentru modulul de backend includ:

- **API Gateway**: responsabil pentru asigurarea securității și rutarea pachetelor către serviciile solicitate.
- **Eureka Discovery Service**: serviciu oferit de Spring Cloud pentru a păstra o listă actualizată în timp real a tuturor serviciilor disponibile.
- **Authorization Service**: utilizat împreună cu Spring Security pentru eliberarea token-urilor de autentificare pentru diferitele tipuri de utilizatori.
- **User Service**: reprezintă serviciul principal, care include logica implementării pentru aplicația mobilă a clienților.
- **Admin Coach Service**: proiectat pentru a deservi date utilizatorilor de tip administrator și antrenor.
- **Chat Service**: responsabil cu livrarea rapidă a mesajelor între clienți și antrenori, permițând un număr mare de conexiuni active.

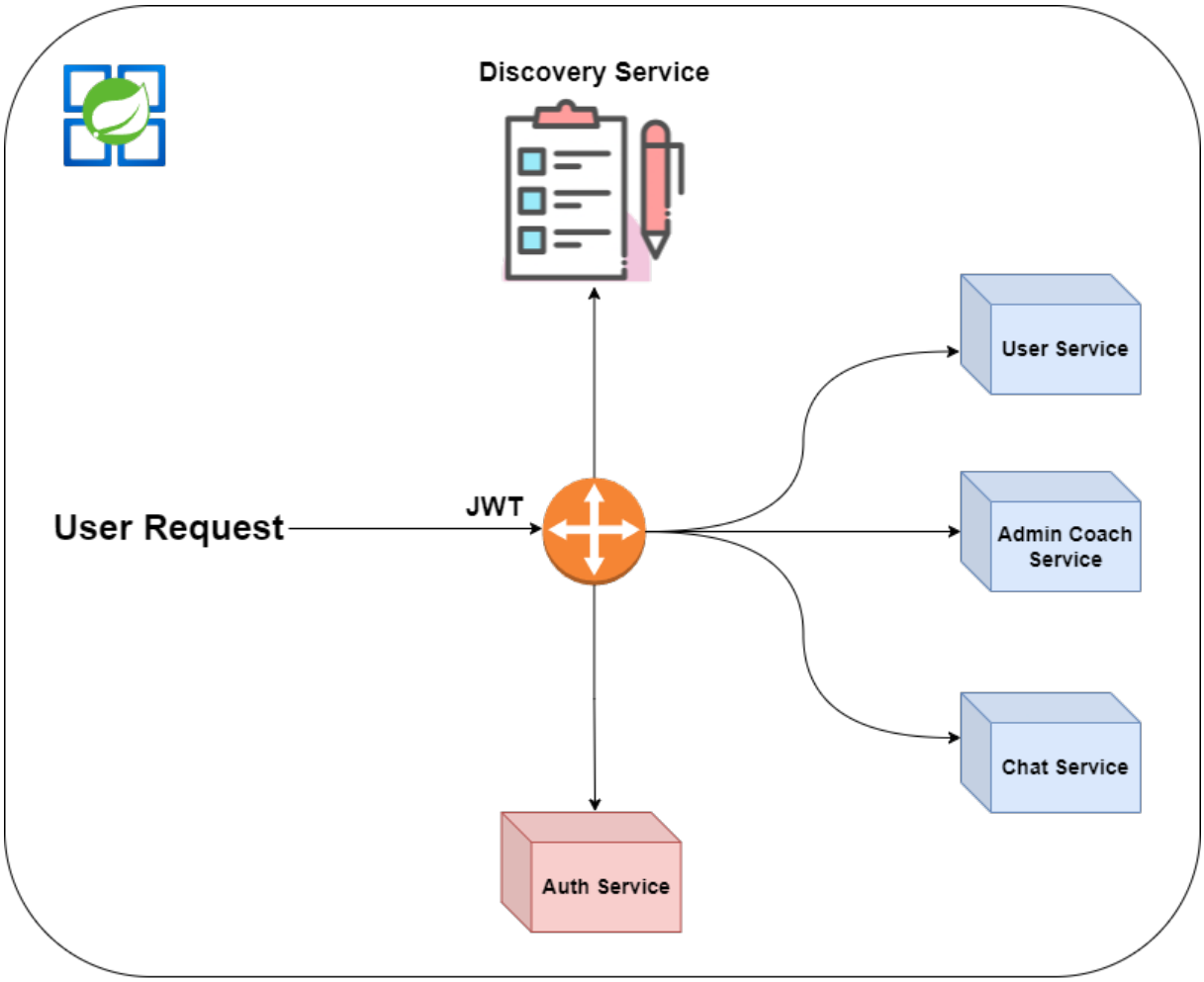

Fig. 4.4: Arhitectura Backend

4.5.3.1 Spring Boot

Framework-ul Spring Boot, si in special Spring Cloud, reprezinta pilonul principal al platformei, oferind solutii moderne si eficiente pentru nevoile unei aplicatii ce folosete un pattern orientat catre mictoservicii.

Spring Cloud permite modularizarea serviciilor platformei, astfel încât fiecare entitate are asociată o funcționalitate bine definită. În acest fel, serviciile de autorizare (Auth Service), de descoperire (Discovery Service) și de rutare (API Gateway) colaborează în procesul de autentificare și autorizare a utilizatorilor în platformă.

Serviciul de autorizare și autentificare acționează ca o unitate centrală de generare și validare a JWT-urilor, create pe baza unui secret stocat exclusiv la nivelul acestuia. Framework-ul Spring Security joacă un rol esențial în funcționarea acestui serviciu, reprezentând standardul de facto pentru aplicațiile web dezvoltate în limbajul de programare Java. De asemenea, acest modul oferă suport pentru serviciile REST destinate înregistrării și autentificării în platformă.

API Gateway este o componentă a bibliotecii Spring Cloud care își propune

să ofere metode simple, dar eficiente, de rutare a interogărilor de tip HTTP către API-uri REST. De asemenea, acesta oferă suport și pentru cerințe suplimentare, cum ar fi: securitatea, monitorizarea și toleranța la defecte. Acest modul colaborează cu serviciul de autorizare și autentificare pentru a implementa un filtru de tip HTTP, la nivelul căruia interogările sunt acceptate sau respinse în funcție de validitatea JWT-ului.

Serviciul de descoperire este o parte esențială a creării aplicațiilor bazate pe microservicii. O componentă furnizată de biblioteca Spring Cloud este Eureka Service Discovery, care permite păstrarea unei liste actualizate în permanență cu domeniul și portul serviciilor înregistrate în cadrul aplicației.

User și Admin Coach Service sunt de asemenea implementate cu ajutorul framework-ului Spring Boot, fiind responsabile de o parte consistentă a funcționalităților platformei de coaching.

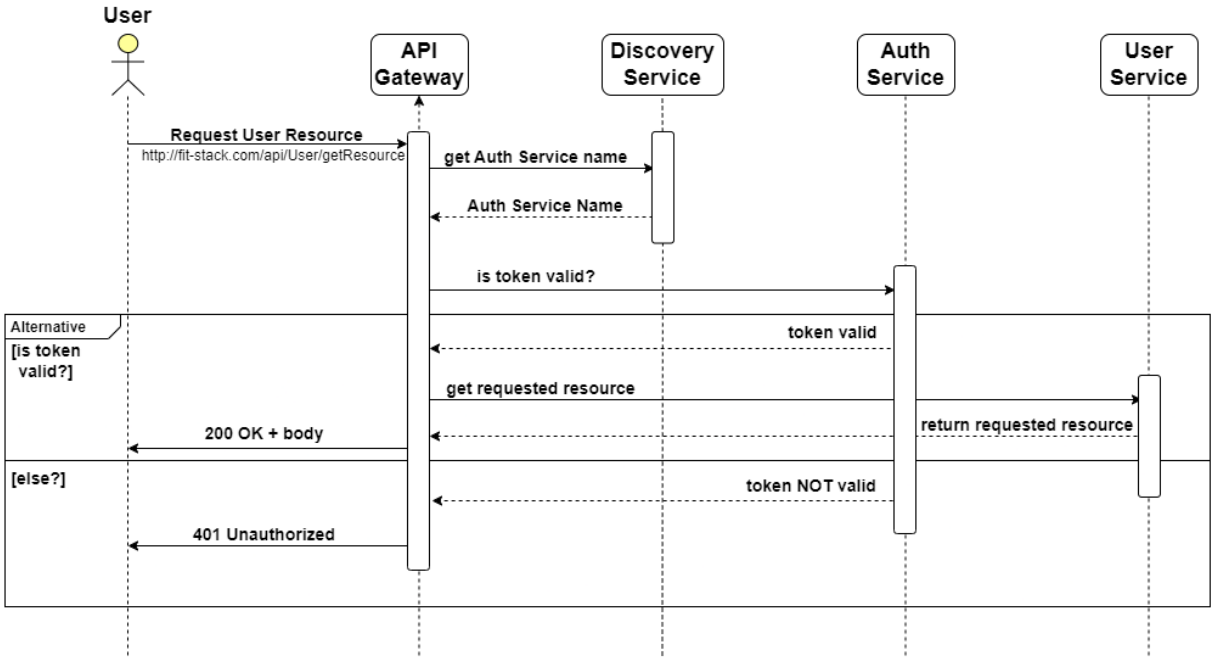

Fig. 4.5: Flux autorizare distribuită

În diagrama 4.5 este detaliat procesul prin care cele trei servicii colaborează pentru a valida interogările unui utilizator extern asupra sistemului. Utilizatorul va încerca să acceseze resursa dorită prin apelarea unui REST API, atașând un antet de autentificare interogării HTTP. API Gateway aplică tuturor cererilor un filtru HTTP, la nivelul căruia se analizează legitimitatea tichetului JWT aflat la nivelul antetului Authorization. Acest antet va conține un camp bearer unde se află tichetul JWT care va fi evaluat de serviciul de autentificare și autorizare prin colaborare cu filtrul HTTP. În funcție de răspunsul acestuia, sistemul va returna fie mesajul de eroare ”401 Unauthorized”, fie va continua fluxul de deservire a interogării, gateway-ul dirijând traficul către serviciul aferent care va furniza

datele clientului. Atunci când acest proces este finalizat, se va returna mesajul de succes HTTP ”200 OK”, care va avea atașat calupul de date dorit.

4.5.3.2 Serviciul de Chat

Pentru serviciul de chat, s-a optat pentru implementarea aplicației în limbajul de programare Node.js, deoarece serverul furnizat de acesta poate menține un număr mare de conexiuni deschise datorită infrastructurii sale de tip event-driven[1].

Serviciul de mesagerie permite clienților mobile să comunice cu antrenorii înscriși în platformă, crescând nivelul de coeziune între aceștia. Acest serviciu are ca scop facilitarea unei metode rapide de comunicare în timp real, în cazul în care clienții aplicației au nevoie de lămuriri în legătură cu conținutul adăugat de antrenori.

Interfața grafică implementată atât pentru clienți, cât și pentru antrenori, este una intuitivă și respectă principiile de design patentate de UI/UX, precum:

- mesajele trimise de cel care trimite mesajele sunt poziționate pe partea stângă a ecranului, în timp ce mesajele recepționate se află pe partea dreaptă;
- mesajele primite au asociată data și ora la care au fost primite;
- fiecare intrare din lista cu conversațiile are atașat avatarul persoanei, numele și ultimul mesaj primit/trimis;
- opțiunea de a căuta conversații după numele utilizatorului.

4.5.4 Servicii Google

Platforma de coaching online implementează o serie de funcționalități care necesită comunicarea cu servicii terțe parte furnizate de platforma Google.

4.5.4.1 Serviciul de autentificare cu contul Google

Autentificarea și înregistrarea cu o platformă socială sunt extrem de facile în cadrul acestei aplicații, reprezentând o metodă simplă de a obține un cont în aplicație într-o manieră sigură, folosind conceptul de single sign-on.[2]. Conceptul de social login[3] este unul foarte răspândit în rândul aplicațiilor moderne, deoarece este o opțiune mai rapidă și confortabilă pentru utilizatori.

[1] https://aws.amazon.com/event-driven-architecture/

[2] https://en.wikipedia.org/wiki/Single_sign-on

[3] https://en.wikipedia.org/wiki/Social_login

Platforma de coaching online folosește modulul de autentificare cu contul Google furnizat de platforma Firebase. Modulul Firebase Authentication[4] furnizează o gamă largă de platforme sociale ce pot fi folosite pentru a autentifica utilizatorii în cadrul aplicației prin metode robuste din punct de vedere al securității.

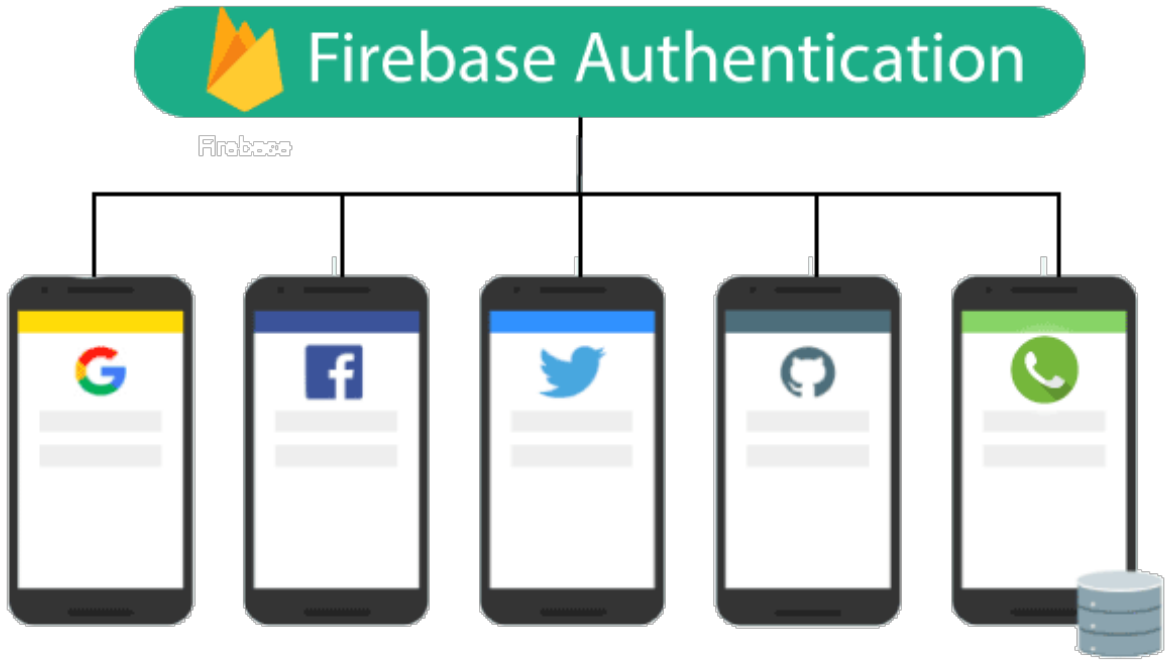

Fig. 4.6: Firebase social login

Modulul Firebase Authentication, împreună cu platforma Google, aduc o multitudine de beneficii în cadrul platformei de coaching online:

- **Securitate**: Autentificarea este gestionată de Google, ceea ce înseamnă standarde ridicate de securitate, fiind una dintre cele mai populare platforme ce oferă funcționalitatea de social login.
- **Experiență îmbunătățită pentru utilizator**: Utilizatorul trebuie doar să apese butonul dedicat autentificării/înregistrării cu platforma Google și să aleagă contul acestuia, urmat de introducerea parolei. Aceasta reprezintă o variantă mult simplificată față de completarea formularelor de înregistrare sau autentificare, deoarece utilizatorul este nevoit să țină minte o singură parolă.
- **Update automat**: Firebase gestionează actualizările serviciului de autentificare, însemnând că dezvoltatorii nu sunt nevoiți să rescrie codul pentru a menține funcțională autentificarea cu platforma Google.

[4] https://firebase.google.com/docs/auth

- **Simplu de implementat**: Configurarea serviciului de autentificare furnizat de Firebase este mult simplificată față de modul tradițional. Firebase oferă SDK-uri și biblioteci pentru o multitudine de platforme.

4.5.4.2 Google People API

People API[5] este un serviciu furnizat de platforma Google, ce oferă acces programatic la datele utilizatorilor, inclusiv persoane din zona de *contacts* sau chiar *other contacts*, care vor fi folosite pentru a crea lista prietenilor. Aceste informații vor fi utilizate în cadrul aplicației pentru crearea unui clasament cu prietenii utilizatorului. Această funcționalitate se integrează foarte bine cu aplicația de coaching online, deoarece aduce o notă de competitivitate pozitivă care va stimula progresul utilizatorului pe termen lung.

4.5.5 Serviciul paypal

Fig. 4.7: Serviciul PayPal

Mecanismul de plată este crucial în cadrul aplicației de coaching online FitStack, deoarece determină o metodă de trecere de la un cont de utilizator standard la un cont premium unde utilizatorul va avea acces atât la conținutul adăugat de antrenorii înscriși în platformă, cât și la serviciul de mesagerie. Pentru ca utilizatorii să actualizeze contul la unul premium, aceștia vor trebui să opteze pentru o plată recurentă de tip abonament, care va fi reînnoit în fiecare

[5] https://developers.google.com/people

lună pentru suma de zece dolari. Serviciul PayPal[6] oferă aplicației FitStack o metodă sigură și eficientă de efectuare a plății, integrându-se perfect cu cerințele platformei. PayPal oferă o multitudine de planuri de plată, care pot fi customizate pentru a oferi flexibilitate organizațiilor care optează pentru această alternativă.

Integrarea și configurarea serviciului PayPal cu platforma de coaching online s-a realizat prin următorii pași:

1. **Adăugarea unui produs în platforma PayPal**: pentru acest pas a fost folosit Catalog Products REST API[7] furnizat de PayPal. Prin intermediul acestui REST API de tip POST, un utilizator de tip developer al platformei PayPal poate declara produse, cărora ulterior li se va aplica o taxă. În cadrul REST API-ului vor fi furnizate detalii despre produs cum ar fi: nume, tip, descriere, categorie, etc. Ca răspuns la REST API, PayPal va returna date despre produs și implicit ID-ul acestuia ce va fi folosit ulterior pentru crearea taxei.

2. **Asocierea taxei pentru produs**: odată creat, produsului îi pot fi asociate taxe, prin intermediul REST API-ului de tip POST furnizat de platforma PayPal. Pentru aplicația FitStack este necesară configurarea unei plăți de tip abonament ce va fi reînnoit o dată pe lună. Subscriptions REST API[8] este folosit pentru a crea obiectul de tip abonament în cadrul PayPal. Pentru a crea abonamentul a fost apelat REST API-ul cu următoarele detalii: ID-ul produsului creat anterior, un nume, și au fost definite ciclurile de taxă în care a fost configurată perioada de reînnoire și prețul abonamentului. Ca răspuns la REST API, PayPal va returna date despre abonament și implicit ID-ul acestuia care va fi folosit ulterior.

3. **Integrarea butonului PayPal[9]**: serviciul PayPal implementează codul pentru buton care va include tot fluxul pentru un gateway de plată. Codul este scris în limbajul de programare JavaScript și poate fi inclus în cadrul aplicației pentru ca utilizatorii să poată actualiza contul la unul premium.

4. **PayPal WebHook[10]**: Pentru a putea actualiza conturile utilizatorilor trebuie configurat un webhook pentru a putea notifica aplicația de backend în cazul unor evenimente. Pentru asta sunt folosite REST API-urile furnizate de PayPal. În REST API-ul de tip POST de configurare a webhookului trebuie să adăugăm o serie de detalii: evenimentele care vor declanșa notificări și URL-ul care reprezintă webhook listener la nivelul căruia se

[6] https://developer.paypal.com/home
[7] https://developer.paypal.com/docs/api/catalog-products/v1/
[8] https://developer.paypal.com/docs/api/subscriptions/v1/
[9] https://developer.paypal.com/sdk/js/configuration/
[10] https://developer.paypal.com/docs/api/webhooks/v1/

vor lua măsurile necesare pentru a trata notificările menționate anterior. Acest webhook listener este de fapt un REST API din cadrul platformei de coaching online creat pentru a comunica cu serviciul PayPal.

4.5.6 Serviciul de predicție a numărului de calorii

Acest serviciu este proiectat cu scopul de a furniza informații cât mai ancorate în realitate despre numărul de calorii arse pe parcursul unui exercițiu în funcție de durata acestuia, înălțimea, greutatea, genul și vârsta persoanei care îl execută. Acest modul are la bază un model de învățare automată bazat pe regresie liniară multiplă prin intermediul căruia se găsește o corelație puternică între caracteristicile menționate anterior și numărul de calorii arse.

Serviciul de predicție a numărului de calorii este expus prin intermediul unui REST API de tip POST, care primește, prin intermediul conținutului cererii HTTP, în format JSON, caracteristicile necesare pentru a furniza un număr de calorii arse realist în urma apelării modelului de regresie. Serviciul este expus prin intermediul framework-ului Flask[30], fiind o alegere facilă din perspectiva implementării modelului cu ajutorul bibliotecii Scikit-learn[35] dezvoltate în limbajul de programare Python.

4.6 Tehnologii folosite pentru implementare

4.6.1 Tehnologii Backend

Tehnologiile de backend folosite în cadrul platformei de coaching online sunt robuste, performante, eficiente, au documentație exhaustivă și beneficiază de o comunitate puternică.

4.6.1.1 Spring

Framework-ul Spring și modulele ce îl alcătuiesc formează o bază puternică pentru orice aplicație web. Spring se bazează pe limbajul de programare Java și este unul dintre cele mai folosite frameworkuri în domeniul dezvoltării web, fiind cunoscut pentru eficiență și productivitate, deoarece elimină codul de tip boilerplate[11].

Modulele din cadrul frameworkului Spring sunt esentiale dezvoltarii platformei de coaching online, fiind suportul unei aplicatii performante, scalabile si eficiente.

Modulele folosite sunt:

[11] https://en.wikipedia.org/wiki/Boilerplate_code

- **Spring Boot**[42]: oferă un suport major în dezvoltarea aplicațiilor web, facilitând procesul de testare prin introducerea conceptului de injectare a dependințelor.
- **Spring Cloud**[38]: oferă o gamă variată de instrumente specifice dezvoltării aplicațiilor în cloud bazate pe microservicii, precum: Api Gateway, serviciu de descoperire microserviciilor, mecanism de încărcare în mod egal a serviciilor, întreruperi de circuit în cazul unor evenimente neprevăzute.
- **Spring Security**[40]: reprezintă frameworkul de facto folosit pentru autentificarea și autorizarea în sistem prin mai multe mecanisme, cum ar fi OAuth 2.0 și JWT
- **Spring Data JPA**[39]: acest modul este folosit pentru realizarea operațiilor CRUD (Create, Read, Update, Delete) asupra bazei de date. Acesta reprezintă nivelul de repository în cadrul arhitecturii bazate pe pattern-ul stratificat (layered pattern).

4.6.1.2 Node.js

Node.js[26] este un mediu de runtime bazat pe limbajul de programare JavaScript[12]. Node.js folosește V8 Engine, care este nucleul browserului Google Chrome, permițând acestei tehnologii să fie foarte performantă.

Serverul furnizat de Node.js este foarte performant întrucât se bazează pe o paradigmă non-blocantă, astfel acțiunile precum operații I/O sau citirea de la placa de rețea nu blochează firul principal, eliminând timpul de așteptare inutil. Serverul furnizat de Node.js permite, de asemenea, un număr foarte mare de conexiuni, tocmai prin prisma arhitecturii sale event-driven[13], fiind o opțiune potrivită pentru serviciul de mesagerie.

4.6.1.3 Pyhton

Python[32] este un limbaj de programare care oferă suport cuprinzător pentru o multitudine de domenii, inclusiv inteligența artificială, învățarea automată, analiza datelor etc. Framework-ul Flask[30], dezvoltat în Python, oferă un mecanism facil pentru implementarea aplicațiilor web, oferind posibilitatea creării serviciilor REST robuste. Acest framework este folosit în aplicația de coaching online pentru expunerea modelului de regresie liniară multiplă, creat pentru a oferi clienților un serviciu foarte util în domeniul aplicațiilor de fitness,

[12] https://developer.mozilla.org/en-US/docs/Web/JavaScript
[13] https://en.wikipedia.org/wiki/Event-driven_architecture

și anume prezicerea numărului de calorii. Pentru implementarea modelului de regresie a fost folosită biblioteca Scikit-learn[35], care oferă suport extins pentru toate operațiile legate de domeniul învățării automate, inclusiv clasificare, regresie, clusterizare, reducerea dimensionalității și regresie.

4.6.2 Metode de stocare

Alegerea unor metode de stocare potrivite pentru o aplicație este un lucru dificil, necesitând o analiză profundă a tipului de date pe care le manipulează aplicația.

Aplicația stochează o cantitate considerabilă de date cu o structură fixă, motiv pentru care majoritatea informațiilor sunt salvate într-o bază de date relațională orientată pe obiect, și anume PostgreSQL[31]. Această bază de date oferă o multitudine de funcționalități, cum ar fi: securitate, performanță crescută pentru operațiile de citire/scriere și extensii pentru limbajele și framework-urile populare de programare.

Baza de date MongoDB[25] este folosită pentru a stoca în cloud mesajele utilizatorilor. MongoDB este recunoscut pentru rapiditatea operațiilor de scriere și citire datorită gestionării eficiente a datelor. Aceasta este o bază de date orientată pe documente, non-relațională care structurează conținutul în format JSON și este optimizată pentru a deservi serviciile rapid.

Minio[24] este un serviciu *open-source* de stocare a obiectelor care gestionează fișiere de orice tip. Acesta este un serviciu găzduit intern, oferind modalități extinse de manipulare a datelor prin intermediul unei game largi de limbaje de programare precum Java, Python, Go, etc.

În cadrul platformei de coaching online, MinIO[24] este utilizată pentru manipularea conținutului media de tip fotografie sau videoclip. MinIO este proiectat pentru a funcționa în cadrul unui cluster de Kubernetes și este o soluție populară, cunoscută pentru scalabilitate, performanțe crescute, securitate și ușurința configurării.

Platforma FitStack beneficiaza de eficientizarea procesului de citire si scriere a videclipurilor.

Scrierea sau postarea videoclipurilor se realizeaza in mod eficient si performant prin paralelizarea incarcarii videoclipului in bucati egale ca dimensiune. Serverul web care deserveste cererea este exclus din acest proces care consuma foarte multe resurse atat de memorie cat si de procesare, fiind responsabil doar cu initierea si cu finalizarea acestui proces.

Se vor prezenta pașii prin care se execută cererea de postare a unui videoclip în clusterul de MinIO:

1. Clientul, în cazul nostru browserul, face o cerere către serverul web pentru inițierea încărcării videoclipului.

2. Serverul web deserveste către client un număr de URL-uri pre-semnate[14] de tip POST, egal cu dimensiunea videoclipului împărțită la dimensiunea maximă admisă care este de 20 de Megabytes. Astfel, se vor genera un număr de *n* URL-uri prin intermediul cărora browserul va comunica direct cu clusterul de MinIO pentru stocare.

3. Clientul notifică serverul când va trimite ultima parte a videoclipului.

4. Serverul deserveste cererea browserului și semnalează mai departe către clusterul de MinIO faptul că s-a primit ultima parte a videoclipului.

5. La nivelul clusterului, se deserveste cererea respectivă prin fuzionarea celor *n* părți ale videoclipului.

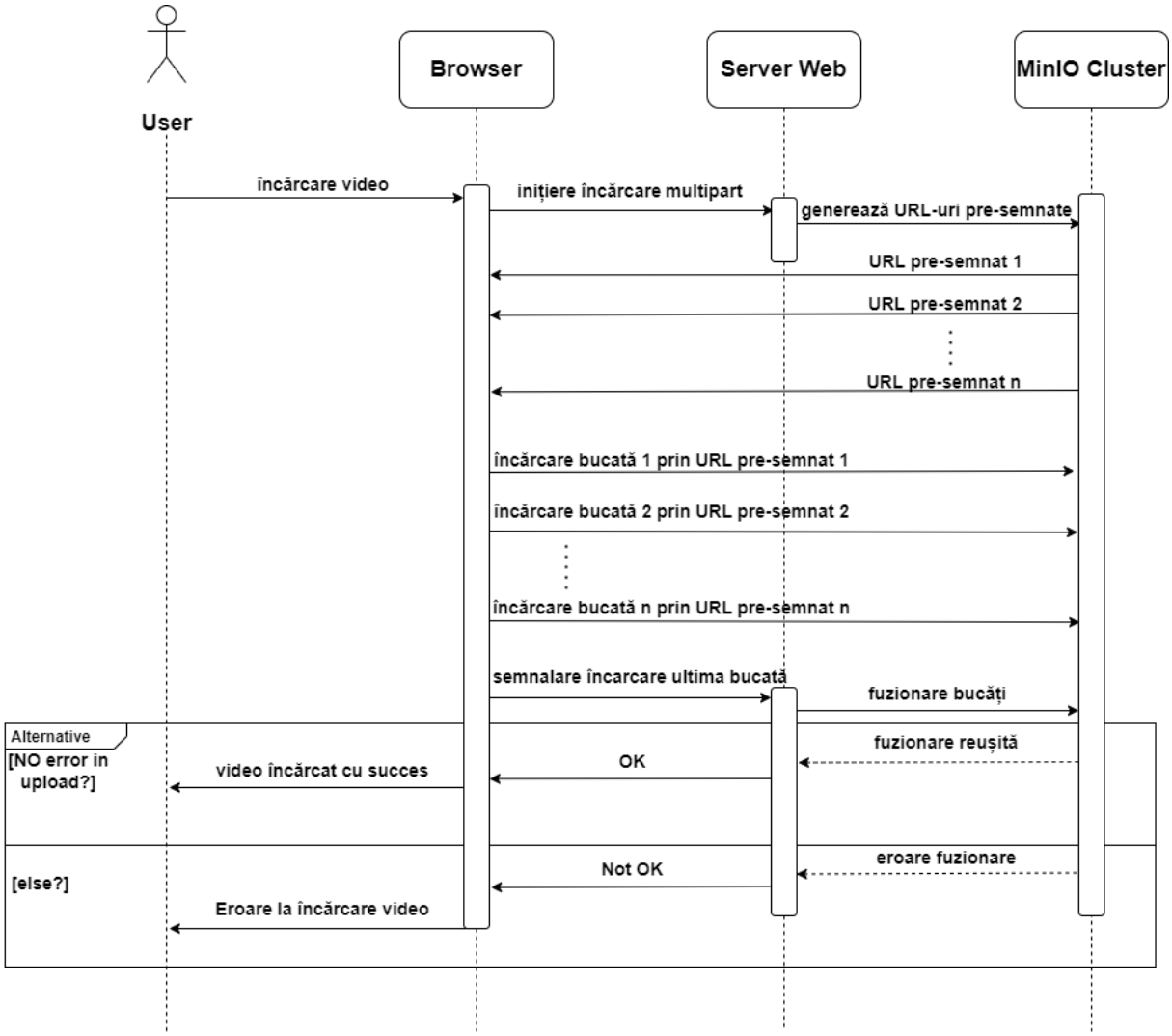

Fig. 4.8: Diagramă de secvență încărcare videoclip

[14] https://min.io/docs/minio/linux/integrations/presigned-put-upload-via-browser.html

De asemenea, citirea de pe server a videoclipurilor se realizează în mod eficient, fiind implementată o funcționalitate de streaming video. Un rol esențial îl are codul de succes 206 HTTP[15]. Antetul răspunsului HTTP cu codul de succes 206 conține o intrare *Content-Range* unde este menționat intervalul de bytes al conținutului primit, alături de câmpul *Body* care conține conținutul cerut. Astfel, se transmit doar bucăți din videoclip, fără a încărca degeaba memoria și banda de rețea a dispozitivului care dorește să vizioneze conținutul video.

4.6.3 Tehnologii Frontend

4.6.3.1 Flutter

Flutter[14] este un framework *open-source* care se bazează pe limbajul de programare Dart. Acest framework a fost creat pentru simplificarea implementării aplicațiilor multi-platformă, întrucât este nevoie de o singură sursă de cod pentru a genera interfețe pentru mai multe sisteme de operare. Codul scris în Flutter, deși nu rulează nativ pe dispozitivele unde este instalat, e recunoscut pentru rapiditatea și fluiditatea componentelor vizuale. Această performanță crescută se datorează nucleului Flutter care este scris în limbajul de programare C++. Dezvoltarea aplicației de coaching online prin intermediul frameworkului Flutter o face să fie disponibilă pe platformele Android și iOS.

4.6.3.2 React.js

React.js[8] este un framework, având la bază limbajul de programare JavaScript. React.js este foarte popular în domeniul dezvoltării interfețelor web, datorită posibilității de a crea componente reutilizabile, crescând modularitatea aplicației și productivitatea dezvoltatorilor. Interfața aplicației dedicată antrenorilor și administratorilor este implementată cu ajutorul acestui framework, deoarece este o soluție simplă și eficientă, care are documentație cuprinzătoare.

4.6.4 Tehnologii Deployment

4.6.4.1 Docker

Docker[6] este o platformă *open-source*, creată pentru împachetarea aplicațiilor în containere izolate care au toate dependențele necesare pentru a putea funcționa. Un container Docker va putea rula oriunde există un nucleu de Docker, oferind portabilitate aplicațiilor. Un container este cu mult mai ușor de gestionat față de o mașină virtuală care nu doar că este foarte înceată în procesul de pornire/oprire, dar ocupă mult mai multă memorie. Izolarea modulelor aplicației în

[15] https://developer.mozilla.org/en-US/docs/Web/HTTP/Status/206

containere de Docker oferă aplicației, scalabilitate, redundanță, performanță și disponibilitate crescută.

Containerele de Docker au la bază imagini de Docker care conțin toate fișierele aplicației alături de dependențele necesare. În cadrul aplicației FitStack, imaginile sunt stocate la nivelul platformei Docker Hub, care este cel mai mare registru de containere din lume.

4.6.4.2 Kubernetes

Kubernetes este o platformă *open-source* dezvoltată pentru orchestrarea containerelor, asigurând minimizarea timpului de nefuncționare a aplicațiilor. Platforma de coaching online se bazează pe această tehnologie pentru asigurarea scalabilității, redundanței, eficienței și toleranței la defecte. Kubernetes garantează disponibilitatea și performanța aplicației chiar și în cazul unui număr semnificativ de utilizatori, ceea ce o face o alegere potrivită pentru un mediu de producție complex.

4.6.4.3 Pipeline CI/CD

Pipeline-ul de CI/CD este o componentă esențială în cadrul arhitecturii aplicației de coaching online, deoarece ușurează munca dezvoltatorului prin integrarea unei noi versiuni a aplicației în mod automat în mediul de producție.

Acest proces se împarte în două mari componente:

- Integrare continuă: pentru acest proces un rol important îl are platforma GitHub Actions[13], care permite configurarea fluxurilor de lucru. GitHub Actions permite reciclarea acțiunilor precum instalarea automată a codului din depozitoriul de git, logarea în platforme precum DockerHub sau Azure, și multe altele, eliminând astfel codul repetitiv în fișierele de configurare. Fiecare microserviciu din cadrul platformei de coaching online are un fișier în care este declarat fluxul de lucru, asigurând automatizarea compilării, testării și împachetării în imagini de Docker pentru fiecare în parte. În urma procesului de integrare continuă, fiecare modul al aplicației va avea o imagine de Docker postată pe Docker Hub, având atașată o etichetă corespunzătoare ultimei versiuni a aplicației.

- Livrare continuă: pentru acest procedeu a fost ales un instrument bazat pe concepte de GitOps, și anume FluxCD[9]. Acest framework aduce cu sine conceptul de *Infrastructure as Code*[16], permițând configurarea infrastructurii prin intermediul codului. Versionarea codului pentru infrastructură este un beneficiu al acestei abordări, fiind posibilă revenirea la o versiune

[16] `https://aws.amazon.com/what-is/iac/`

anterioară în cazul unor evenimente neprevăzute. FluxCD își propune să sincronizeze infrastructura din mediul de producție cu infrastructura declarată în depozitoriul de Git, acest proces fiind cunoscut sub numele de reconciliere.

4.6.4.4 Azure

Azure[18] este o platformă de cloud computing oferită de organizația Microsoft, care include o multitudine de servicii și mecanisme ce oferă dezvoltatorilor posibilitatea de a gestiona aplicațiile în mediul cloud. În cadrul aplicației de coaching online este folosit clusterul de Kubernetes gestionat de platforma Azure[20], care oferă scalabilitate aplicației prin crearea unui grup de noduri ce pot fluctua în funcție de trafic și de încărcătura sistemului. Pentru serviciul de mesagerie se folosește resursa Pub/Sub for Socket.IO[22], care acționează ca un dispecer între server și client, fiind o soluție complet gestionată de Azure. Pentru gestionarea numelui de domeniu al platformei este folosită o zonă DNS gestionată tot de Azure.

4.7 Mecanisme de securitate

Odată cu creșterea exponențială a utilizării tehnologiei și aplicațiilor pentru a stoca date importante, este din ce în ce mai mare nevoia de utilizarea unor metode clare și bine definite de securitate. Platforma de coaching se folosește de o multitudine de mecanisme și instrumente pentru a oferi un nivel ridicat de securitate clienților și datelor acestora.

4.7.1 Metode de autentificare si autorizare

Autentificarea și autorizarea în cadrul platformei de coaching online se realizează prin intermediul modulului Spring Security din cadrul frameworkului Spring, oferind mecanisme de securitate ușor de integrat precum: HTTP Basic, JWT și OAuth2.0. În cadrul platformei se folosește tehnologia JWT pentru a trata autentificarea și autorizarea.

4.7.1.1 JWT

JWT[2] reprezintă un mecanism standard care definește o metodă sigură și compactă de a transmite informații sub forma unui obiect JSON într-un mod securizat între două entități. Informația din cadrul tichetului JWT este

de încredere deoarece este semnată digital, cu un secret prin intermediul algoritmului HMAC[17] sau cu o cheie privată creată cu RSA[18] sau ECDSA[19].

Un JWT este format din trei părți encodate base64[20] și despărțite prin caracterul ”.”:

- Antet: este format din două părți, și anume tipul tichetului, care este ”JWT”, și algoritmul folosit pentru semnare.
- Conținut util: în această parte se adaugă detaliile care se doresc a fi transmise. Câmpurile care se adaugă în această parte a JWT-ului pot fi capuri publice sau private. Câmpurile publice sunt standardizate, fiind disponibile pentru toate părțile implicate într-un schimb de tichet. Câmpurile private sunt personalizate și sunt definite la nivelul unei aplicații sau organizații pentru a reprezenta structura conținutului util.
- Semnătura: pentru a crea această parte este nevoie de cele două părți anterioare, și anume antet și conținut util, encodate în base64 și despărțite prin punct, împreună cu secretul, aplicându-se peste toate acestea algoritmul HMAC.

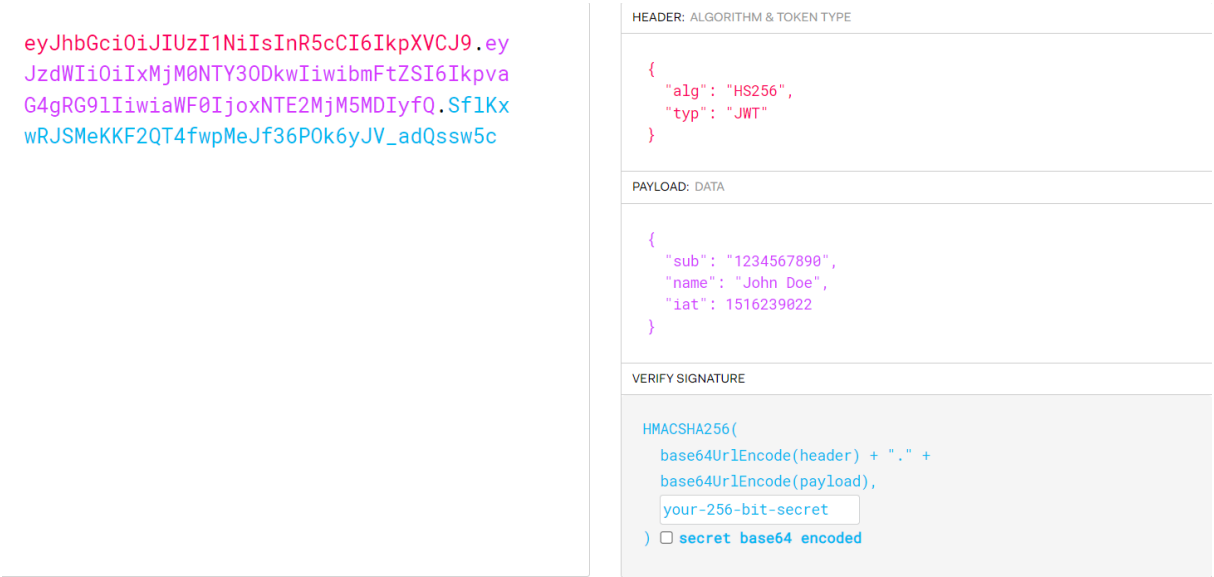

Fig. 4.9: Structura unui JWT[2]

În cadrul platformei de coaching online, conținutul util al JWT conține următoarele câmpuri:

- Publice:

[17] https://www.okta.com/identity-101/hmac/
[18] https://ro.wikipedia.org/wiki/RSA
[19] https://en.wikipedia.org/wiki/Elliptic_Curve_Digital_Signature_Algorithm
[20] https://en.wikipedia.org/wiki/Base64

  - sub: reprezintă atributul subject și este folosit pentru a putea transmite o informație unică la nivelul unui utilizator. În cadrul aplicației FitStack este folosit email-ul utilizatorului.
  - iat: reprezintă data și ora la care a fost creat JWT-ul.
  - exp: reprezintă data și ora la care va expira JWT-ul.

- Private:

  - authorities: conține atât rolul utilizatorului, care poate fi "ROLE_USER", "ROLE_PAYING_USER", "ROLE_COACH", "ROLE_ADMIN", cât și permisiunile acestuia în cadrul sistemului (read, write, update, delete). Pe baza acestor elemente se realizează autorizarea utilizatorilor în sistem.

4.7.1.2 Autentificare cu Google

Google reprezintă o sursă de încredere și este o metodă populară folosită în multe aplicații pentru a oferi o experiență de autentificare rapidă și sigură utilizatorilor. Dacă autentificarea utilizatorului în contul Google este îndeplinită cu succes, acesta va genera un tichet OAuth2.0, care va fi folosit mai departe pentru a avea acces la mai multe API-uri expuse de Google. Autentificarea cu platforma Google gestionează ciclul de viață al tichetului OAuth2.0, fiind reînnoit la fiecare login.

4.7.2 HTTPS

HTTPS[21] este versiunea securizată a lui HTTP, care este protocolul principal utilizat pentru transmiterea datelor între aplicațiile web și browser. HTTPS este folosit pentru criptarea datelor pe parcursul comunicării, ceea ce asigură integritatea și confidențialitatea informațiilor. HTTPS este foarte important deoarece previne pachetele de date să fie monitorizate de către un potențial atacator, fiind mai mult decât necesar în cazul unor aplicații web în care sunt folosite parole sau date ale cardului bancar. Criptarea traficului web este realizată prin intermediul TLS/SSL. HTTPS este realizat în urma transmiterii certificatelor TLS/SSL pentru a dovedi browserului că organizația este cine pretinde că este.

La nivelul platformei de coaching online Fitstack, ciclul de viață al certificatelor este gestionat în mod automat de către unealta cert-manager[4], care colaborează cu platforma Let's Encrypt[17] pentru furnizarea unor certificate valide și recunoscute de majoritatea browserelor. Cert-manager este responsabil de crearea certificatului pe baza căruia va rezulta un secret Kubernetes. Acest

[21] https://www.cloudflare.com/learning/ssl/what-is-https/

secret va dovedi browserului că certificatul obținut este validat de un CA[22] și că este cu adevărat al platformei de coaching online.

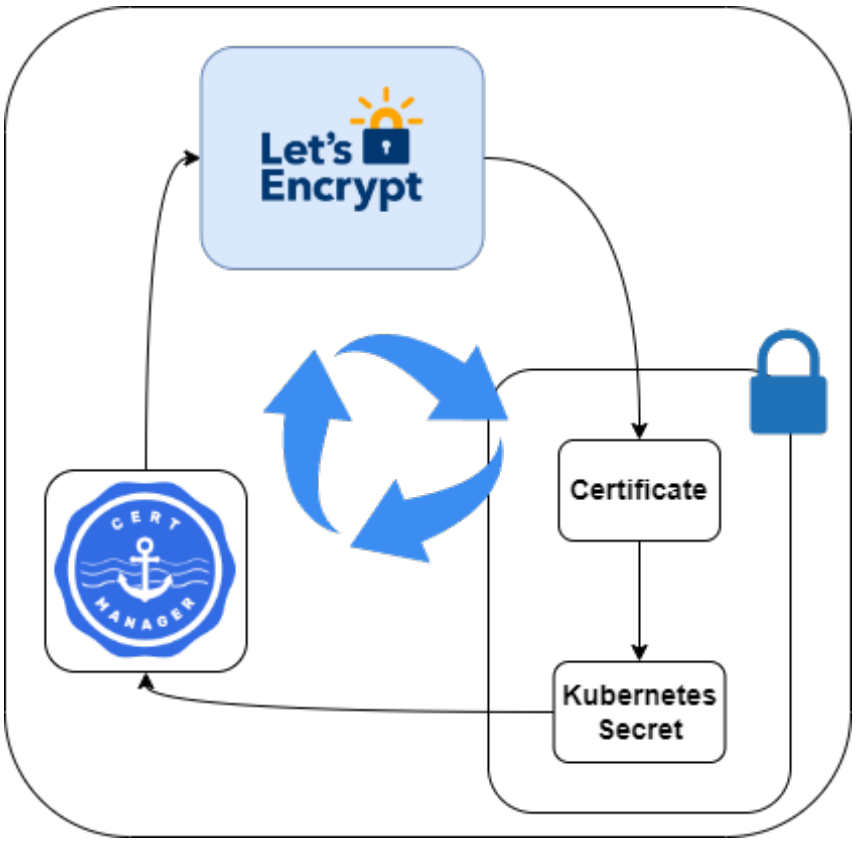

Fig. 4.10: Automatizarea certificatelor

4.8 Rezultatele testelor efectuate și interpretarea acestora

4.8.1 Importanța testării

Testarea funcționalităților software reprezintă un element principal în ciclul de viață al unei aplicații, întrucât se dorește găsirea erorilor cât mai devreme posibil pentru a asigura că toate elementele funcționează corect, atât individual cât și împreună. Scopul dezvoltatorilor este furnizarea unui produs software de calitate pentru clienții care beneficiază de funcționalitățile acestuia. În funcție de mediul în care este folosită aplicația, funcționarea necorespunzătoare a funcționalităților software ar putea avea efecte catastrofale, aducând un impact negativ în încasările organizației.

Proiectarea unui plan de testare cuprinzător este necesară în cazul oricărei aplicații, dezvoltatorii garantând pentru calitatea și funcționalitatea corectă a

[22] https://en.wikipedia.org/wiki/Certificate_authority

software-ului pe care îl creează.

Aplicația de coaching online este o aplicație complexă, ceea ce aduce în discuție necesitatea unui plan de testare cuprinzător pentru a asigura că majoritatea funcționalităților se comportă corect în toate situațiile. Aplicația FitStack a fost testată atât manual, prin verificarea funcționalităților principale pentru a descoperi orice anomalie, cât și automat, prin implementarea unor scripturi care testează produsul software înainte de a se integra în mediul de producție.

Testarea automată în cadrul platformei de coaching online, se împarte în mai multe tipuri de verificare a software-ului, cum ar fi:

- **Testarea unitară**: se verifică funcțiile și componentele în mod individual.
- **Testarea de integrare**: se testează modul în care colaborează componentele principale ale aplicației pentru realizarea cerințelor.
- **Testarea modelului de învățare automată**: vor fi folosite mecanisme pentru testarea acurateții sistemului.
- **Testarea orientată pe sarcini**: se va verifica cum se comportă aplicația în scenarii de utilizare reale.

4.8.2 Teste unitare

Testarea unitară[1] este procesul prin care se testează cele mai mici unități de cod din cadrul unui sistem software. Principalul scop al testării unitare este asigurarea că fiecare unitate izolată, precum funcție, metodă sau componentă modulară a sistemului, se execută și are rezultatele pe care le așteptăm. Acesta este cel mai de jos nivel în cadrul procesului de testare, având și cel mai mare impact în detectarea erorilor cât mai devreme în procesul dezvoltării aplicației. Testarea unitară pune temeliile conceptului de *Test-driven Development*[23], care are la bază configurarea scripturilor de testare înaintea implementării funcționalităților propriu-zise ale aplicației, asigurând astfel funcționarea corectă a sistemului.

[23] https://en.wikipedia.org/wiki/Test-driven_development

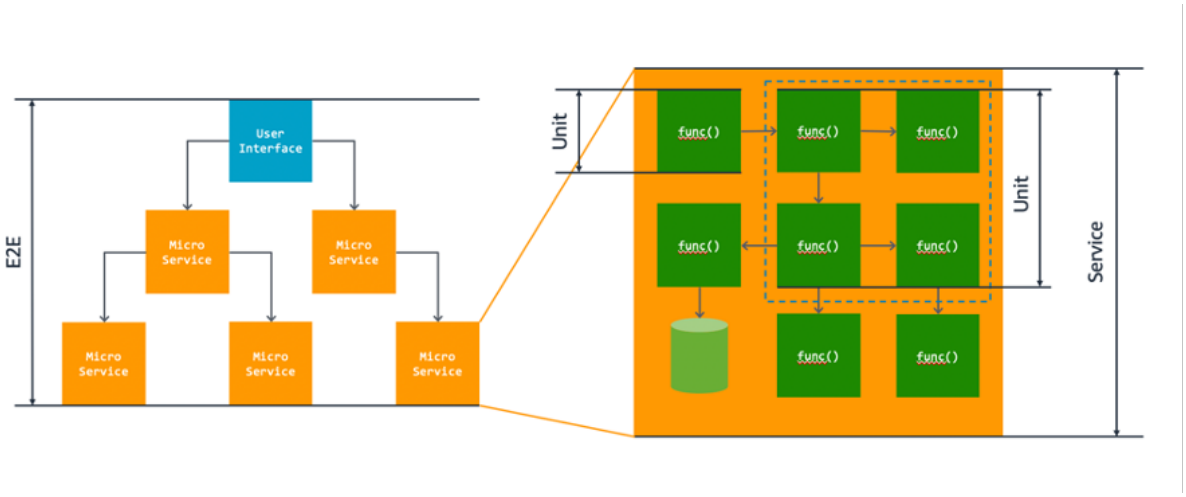

Fig. 4.11: Testarea unitară[1]

În cadrul platformei de coaching online testarea unitară se realizează cu ajutorul frameworkului JUnit5[24] care reprezintă un standard pentru testarea unitară în cadrul aplicațiilor dezvoltate în limbajul de programare Java.

Acest framework este recunoscut ca fiind unul dintre cele mai folosite frameworkuri pentru testare, din următoarele considerente:

- framework *open-source*
- furnizează adnotări pentru identificarea metodelor de test
- o gamă largă de aserțiuni folosite pentru a testa rezultatele așteptate
- executarea automată a testelor

Un alt framework utilizat în testarea unitară este Mockito[25], folosit pentru crearea obiectelor de tip *mock*[26], configurate pentru a simula funcționalitatea unei componente a aplicației. Cu ajutorul acestui framework se poate testa în mod izolat comportamentul unei componente prin simularea modulelor cu care aceasta comunică. Mockito permite setarea așteptărilor pentru metodele obiectelor mock, cum ar fi ce valori ar trebui să returneze sau cum ar trebui să răspundă la anumite apeluri. În plus, este posibilă verificarea interacțiunilor cu aceste obiecte mock pentru a se asigura că metodele sunt apelate în ordinea corectă și cu parametrii așteptați.

Spring este o alegere bună în acest context, deoarece framework-ul se bazează pe concepte de injectare a dependențelor, ceea ce ușurează procesul de creare a mockurilor și, în general, procesul de testare unitară.

[24] https://junit.org/junit5/docs/current/user-guide/index.html
[25] https://javadoc.io/doc/org.mockito/mockito-core/latest/org/mockito/Mockito.html
[26] https://en.wikipedia.org/wiki/Mock_object

4.8.2.1 Definirea testelor unitare

Vor fi definite o serie de teste unitare care au fost efectuate asupra aplicatiei. Acestea vor avea un cod special, pentru a putea fi adaugate in planul de testare:

1. TU01: Testarea REST API-ului pentru înregistrarea pe platformă are loc atunci când utilizatorul a furnizat corect toate datele necesare, iar adresa de email, care trebuie să fie unică, nu este deja prezentă în baza de date a platformei. Codul returnat de controller este HTTP 200 *OK*, împreună cu tichetul JWT.

2. TU02: Testarea REST API-ului pentru înregistrarea pe platformă în cazul în care adresa de email este deja prezentă în baza de date presupune generarea unei excepții și returnarea codului de eroare 400 *Bad Request*.

3. TU03: Testarea REST API-ului pentru autentificarea în platformă cu credențialele corecte presupune returnarea codului HTTP 200 *OK*, împreună cu tichetul JWT.

4. TU04: Testarea REST API-ului pentru autentificarea în platformă cu credențiale greșite va presupune returnarea codului de eroare HTTP 403 *Forbidden*.

5. TU05: Testarea REST API-ului pentru validarea tichetelor JWT presupune că în cazul în care tichetul este conform (adică nu a expirat sau nu a fost alterat), va fi returnat mesajul HTTP 200 *OK*.

6. TU06: Testarea REST API-ului pentru validarea tichetelor care nu sunt conforme (au expirat sau au fost alterate) presupune aruncarea unei excepții și returnarea unui cod de eroare HTTP 403 *Forbidden*.

7. TU07: Testarea REST API-ului pentru resetarea parolei. În cazul în care adresa de email pentru care se face cererea este existentă în baza de date, se așteaptă generarea unui token unic pe baza căruia se va crea un URL trimis pe adresa de email furnizată în cererea de schimbare a parolei. Se așteaptă codul HTTP 200 *OK*, alături de link-ul ce trebuie accesat de utilizator pentru resetare.

8. TU08: Testarea REST API-ului pentru resetarea parolei în cazul în care adresa de email nu este prezentă în baza de date. În acest caz, se va returna codul de eroare HTTP 400 *Bad Request*.

9. TU09: Testarea REST API-ului pentru adăugarea unui exercițiu. În cazul în care toate datele sunt corecte, se va returna codul 200 HTTP *OK*, alături de id-ul exercițiului care tocmai a fost creat.

10. TU10: Testarea REST API-ului pentru adăugarea unui exercițiu în cazul în care datele nu sunt corecte. Se va returna codul de eroare HTTP 400 *Bad Request*.

11. TU11: Testarea REST API-ului pentru adăugarea unui antrenament. În cazul în care toate datele sunt corecte(exercițiile din cadrul antrenamentului sunt create de antrenor), se va returna codul 200 HTTP *OK*, alături de id-ul antrenamentului care tocmai a fost creat.

12. TU12: Testarea REST API-ului pentru adăugarea unui antrenament. În cazul în care datele nu sunt corecte, se va returna codul de eroare HTTP 400 *Bad Request*.

13. TU13: Testarea REST API-ului pentru adăugarea unui program de fitness. În cazul în care toate datele sunt corecte(exitstă toate antrenamentele din cadrul programului și sunt create de antrenor), se va returna codul 200 HTTP *OK*, alături de id-ul programului de fitness care tocmai a fost creat.

14. TU14: Testarea REST API-ului pentru adăugarea unui program de fitness. În cazul în care datele nu sunt corecte, se va returna codul de eroare HTTP 400 *Bad Request*.

15. TU15: Testarea serviciului de înregistrare presupune verificarea logicii din spatele REST API-ului. În cazul în care toate datele furnizate sunt corecte (adresa de email nu există în baza de date), funcția va returna tichetul JWT.

16. TU16: Testarea serviciului de înregistrare în cazul în care adresa de email este deja prezentă în baza de date. În această situație, va fi aruncată o excepție care va fi tratată în *controller*.

17. TU17: Testarea serviciului de autentificare în cazul în care credențialele sunt corecte. Funcția va returna tichetul JWT.

18. TU18: Testarea serviciului de autentificare în cazul în care credențialele sunt incorecte. În această situație, funcția va arunca o excepție care va fi tratată în controller.

19. TU19: Testarea serviciului de verificare a tichetului în cazul în care JWT-ul este conform (adică nu a expirat sau nu a fost alterat). Se așteaptă returnarea rezultatului *true*.

20. TU20: Testarea serviciului de verificare în cazul în care JWT-ul nu este conform (adică a expirat sau a fost alterat). În această situație se așteaptă returnarea valorii *false*.

21. TU21: Testarea serviciului de resetare a parolei în cazul în care adresa de email a cererii este prezentă în baza de date. Se așteaptă generarea token-ului unic pe baza căruia se construiește URL-ul pe care îl va accesa utilizatorul. Se așteaptă returnarea URL-ului.

22. TU22: Testarea serviciului de adăugare a unui exercițiu în care toate datele furnizate sunt corecte. În acest caz, se așteaptă adăugarea corectă a exercițiului în baza de date și returnarea id-ului său.

23. TU23: Testarea serviciului de adăugare a unui exercițiu în cazul în care datele furnizate nu sunt corecte. În acest caz, funcția va arunca o excepție care va fi tratată în controller.

24. TU24: Testarea serviciului de adăugare a unui antrenament în care toate datele furnizate sunt corecte(exercițiile aparțin antrenorului). În acest caz, se așteaptă adăugarea corectă a antrenamentului în baza de date și returnarea id-ului său.

25. TU25: Testarea serviciului de adăugare a unui antrenament în cazul în care datele furnizate nu sunt corecte(cel puțin un exercițiu nu aparține antrenorului). În acest caz, funcția va arunca o excepție care va fi tratată în controller.

26. TU26: Testarea serviciului de adăugare a unui program de fitness în care toate datele furnizate sunt corecte(toate antrenamentele aparțin antrenorului). În acest caz, se așteaptă adăugarea corectă a programului în baza de date și returnarea id-ului său.

27. TU27: Testarea serviciului de adăugare a unui program de fitness în cazul în care datele furnizate nu sunt corecte(cel puțin un antrenament nu aparține antrenorului). În acest caz, funcția va arunca o excepție care va fi tratată în controller.

4.8.3 Teste de integrare

Testarea de integrare[11] este o etapă esențială a procesului de testare, fiind verificat nu doar modul în care colaborează modulele componente, ci și schimbul de date dintre acestea.

Scopul acestei etape este de a descoperi erori cât mai devreme în procesul de dezvoltare al aplicației în ceea ce privește comunicarea între modulele componente. Această etapă se realizează după testarea unitară, asigurând verificarea erorilor de integrare. Testarea de integrare verifică funcționalitățile aplicației de la un capăt la altul, asigurând coeziunea între nivelurile diferite ale aplicației.

Platforma de coaching online se bazează pe un șablon arhitectural stratificat, fiecare nivel având rolul lui bine definit. Astfel se dorește testarea interoperabilității între stratul *Presentation*, care expune REST API-urile aplicației, stratul *Business Logic*, care conține majoritatea implementării funcționalităților, stratul *Persistence*, care acționează ca o interfață pentru baza de date și oferă metode CRUD (Create, Read, Update, Delete) asupra acesteia, și stratul *Database*, unde se stochează propriu-zis datele.

Dezvoltatorii întâlnesc adesea dificultăți în configurarea unei baze de date pentru această etapă de testare. Testarea de integrare presupune utilizarea unei baze de date reale, nu a unor mock-uri. În ajutor vine framework-ul de testare Testcontainers, care permite crearea unor containere Docker efemere ce vor fi șterse după finalizarea procesului de testare. Ciclul de viață al containerelor poate fi declarat programatic, astfel încât crearea unui nou container pentru o bază de date Redis[27] poate fi realizată printr-o singură linie de cod:

```
GenericContainer redis = new GenericContainer("redis
    :5.0.3-alpine").withExposedPorts(6379);
```

Pentru a funcționa acest framework, este nevoie de un nucleu Docker în mediul de lucru unde se execută testarea.

Pentru această etapă există două abordări principale de testare, și anume *top down* și *bottom up*.

4.8.3.1 Top down

Testarea de integrare cu o abordare de tip *top down* [10] are atât beneficii cât și defecte. Acest model este cunoscut și ca testarea de integrare incrementală, deoarece nivelele cele mai înalte din ierarhia sistemului sunt testate înaintea submodulelor. Pentru a simula funcționalitatea modulelor neimplementate care se situează mai jos în ierarhie, se folosesc module false numite *stubs*[28], care înlocuiesc comportamentul unor componente până vor fi implementate. Această abordare se axează pe întreg sistemul încă de la începutul implementării, mitigând din start orice problemă critică cu modulele principale ale aplicației.

4.8.3.2 Bottom up

Testarea de tip *bottom up* [10] prioritizează testarea modulelor care se situează mai jos în ierarhie, urmând să fie integrate pe parcurs în arhitectura sistemului. Această abordare se concentrează pe funcționalitatea componentelor și nu pe funcționalitatea întregului sistem. Testarea de tip *bottom up* este

[27] https://redis.io/

[28] https://en.wikipedia.org/wiki/Method_stub

foarte populară, deoarece permite dezvoltarea aplicației în paralel de către echipe diferite de dezvoltatori.

4.8.3.3 Definirea testelor de integrare

Vor fi definite o serie de teste de integrare care au fost efectuate asupra aplicatiei. Acestea vor avea un cod special, pentru a putea fi adaugate in planul de testare:

1. TI01: Testarea integrării între nivelul *presentation*, *business*, *persistence* și *database* pentru serviciul de autorizare și autentificare. Se va verifica comunicarea corectă între acestea.
2. TI02: Testarea integrării între nivelul *presentation*, *business*, *persistence* și *database* pentru serviciul utilizatorilor standard și premium. Se va verifica comunicarea corectă între acestea.
3. TI03: Testarea integrării între nivelul *presentation*, *business*, *persistence* și *database* pentru serviciul antrenorilor și administratorilor. Se va verifica comunicarea corectă între acestea.
4. TI04: Testarea comunicării între API Gateway și microserviciile către care este redirecționat traficul.
5. TI05: Testarea colaborării între serviciile de autorizare și autentificare, API Gateway și Eureka pentru eliberarea tichetelor JWT.
6. TI06: Testarea comunicării între aplicația mobilă și serverul Tomcat.
7. TI07: Testarea comunicării între aplicația web și serverul Tomcat.
8. TI08: Testarea schimbului de mesaje prin intermediul brokerului *Web PubSub for Socket.IO*.
9. TI09: Testarea conexiunii între serverul Tomcat și clusterul Minio.
10. TI10: Testarea comunicării între serverul Tomcat și serviciul de mail.

4.8.4 Testarea modelului de invațare automată

Tesatrea oricarui model de invatare automata este esentiala deoarece ofera informatii despre cum se va comporta acesta in mediul de productie. Pentru antrenarea modelului a fost ales un procent de 70% totalitatea intrarilor din setul de date, iar pentru etapa de testare sunt folosite 30% din date. Pentru evaluarea modelului de regresie exista anumite metrici care reflecta performantele sistemului, precum:

- **R-squared score**: Scorul R^2 reprezintă un mecanism prin care se calculează proporția variabilității surprinse de model în comparație cu variabilitatea totală a datelor. Astfel, se calculează distanța punctelor care reprezintă valorile reale față de graficul asociat modelului de regresie.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{4.2}$$

  Modelul de regresie a obținut un scor de acuratețe de 93.3 în urma calculării scorului R^2.

- **Root Mean Square Error**: sau RMSE măsoară rădăcina pătrată a mediei pătratelor diferențelor între valorile prezise de model și valorile reale. Cu cât valoarea RMSE este mai mică, cu atât modelul are o precizie mai mare în predicțiile sale

$$\sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{4.3}$$

  Modelul de regresie a obținut un scorul 16.20 în calcularea valorii RMSE, ceea ce înseamnă că modelul va prezice numărul de calorii cu o eroare medie de aproximativ 16 unități de măsură a variabilei țintă.

Aceste rezultate obținute în urma testării sunt acceptabile pentru un model de predicție al caloriilor, sugerând faptul că există o corelație puternică între numărul de calorii și caracteristicile alese: vârsta, greutate, înălțime, gen și durata exercițiului.

4.8.5 Teste orientate pe sarcini

Din setul de cerințe funcționale definite anterior se vor extrage câteva pentru a le descrie și a observa cum se realizează testarea acestora.

4.8.5.1 Autentificarea utilizatorilor în aplicația mobile

Un utilizator al platformei de coaching online se va putea autentifica atât pe baza credențialelor stabilite în momentul înregistrării, cât și prin intermediul platformei Google, dacă au optat pentru înregistrare cu contul Google.

În cazul în care adresa de email sau parola utilizatorului sunt greșite, se va afișa un mesaj de eroare sugestiv: ”email/parolă incorecte”, semnalând utilizatorului că va trebui să furnizeze un set nou de credențiale, întrucat cele introduse anterior nu sunt conforme cu cele existente în baza de date.

Dacă utilizatorul optează pentru autentificarea cu contul Google, pe ecran va apărea formularul de sign in furnizat de Google, clientul fiind nevoit să aleagă un cont și să ofere credențialele sale. Dacă acesta introduce credențiale greșite, se va afișa un mesaj de eroare.

În cazul în care un utilizator s-a înregistrat în platformă cu contul Google și încearcă să se autentifice prin formularul clasic de login, se va afișa un mesaj de eroare adecvat. De asemenea, cazul invers, în care utilizatorul încearcă să se autentifice cu contul Google folosind formularul clasic de login, va fi tratat tot printr-un mesaj de eroare sugestiv.

4.8.5.2 Începerea unui antrenament

Alegerea unui antrenament popular poate fi realizată în ecranul principal al aplicației. Un client poate vizualiza exercițiile din cadrul unui antrenament prin apăsarea pe elementul de interfață atașat acestuia. Începerea unui antrenament se realizează prin apăsarea butonului ”Start workout”. Exercițiile vor fi derulate după executarea acestora, prin apăsarea butonului ”Next Exercise”. Dacă utilizatorul nu setează greutatea, numărul de repetări nu va putea trece la următorul, acest aspect fiind semnalat printr-un mesaj sugestiv.

În cazul în care clientul nu poate continua exercițiul din cauza unor evenimente externe, acesta poate apăsa pe butonul ”Înapoi” din colțul stânga sus al ecranului, salvând starea actuală a antrenamentului pentru a-l continua mai târziu. Finalizarea antrenamentului este marcată de apăsarea butonului ”Finish workout”, fiind redirecționat către pagina unde vor fi afișate premiile și punctajul acumulat.

4.8.5.3 Adăugarea unui antrenament de către un antrenor

Adăugarea unui antrenament de către un antrenor se poate realiza din pagina principală a aplicației, prin intermediul evenimentului de click pe componenta vizuală asociată. În continuare, antrenorul trebuie să introducă informații prin completarea unui formular în care va furniza: numele, descrierea, o poză de copertă și un set de exerciții. Aplicația va oferi o serie de filtre care vor ajuta antrenorii în alegerea exercițiilor potrivite. Dacă nu există niciun rezultat în urma aplicării filtrelor adăugate de antrenor, se va afișa un mesaj sugestiv. După ce au fost furnizate toate aceste detalii, antrenorul poate apăsa pe butonul ”Submit workout”, afișându-se un mesaj sugestiv în caz de succes sau eroare.

4.8.6 Raport de testare

În tabel4.1 vor fi prezentate rezultatele testelor descrise. Fiecare test are un id unic, identic cu cel folosit in definirea planului de testare.

ID	Rezultat	Observații
TU01	Passed	API-ul se comportă corect și returnează codul HTTP 200
TU02	Passed	API-ul se comportă corect și returnează codul HTTP 400
TU03	Passed	API-ul se comportă corect și returnează codul HTTP 200
TU04	Passed	API-ul se comportă corect și returnează codul HTTP 403
TU05	Passed	API-ul se comportă corect și returnează codul HTTP 200
TU06	Passed	API-ul se comportă corect și returnează codul HTTP 403
TU07	Passed	API-ul se comportă corect și returnează codul HTTP 200
TU08	Passed	API-ul se comportă corect și returnează codul HTTP 400
TU09	Passed	API-ul se comportă corect și returnează codul HTTP 200
TU10	Passed	API-ul se comportă corect și returnează codul HTTP 400
TU11	Passed	API-ul se comportă corect și returnează codul HTTP 200
TU12	Passed	API-ul se comportă corect și returnează codul HTTP 400
TU13	Passed	API-ul se comportă corect și returnează codul HTTP 200
TU14	Passed	API-ul se comportă corect și returnează codul HTTP 400
TU15	Passed	Funcția returnează tichetul JWT
TU16	Passed	Funcția aruncă o excepție
TU17	Passed	Funcția returnează tichetul JWT
TU18	Passed	Funcția aruncă o excepție
TU19	Passed	Funcția returnează rezultatul *true*
TU20	Passed	Funcția returnează rezultatul *false*
TU21	Passed	Funcția returnează URL-ul pentru resetare
TU22	Passed	Funcția returnează id-ul exercițiului
TU23	Passed	Funcția aruncă o excepție
TU24	Passed	Funcția returnează id-ul antrenamentului
TU25	Passed	Funcția aruncă o excepție
TU26	Passed	Funcția returnează id-ul programului
TU27	Passed	Funcția aruncă o excepție
TI01	Passed	nivelurile comunică corect
TI02	Passed	nivelurile comunică corect
TI03	Passed	nivelurile comunică corect
TI04	Passed	API Gateway redirectează traficul corect către servicii
TI05	Passed	Modulele comunică corect. Se generează JWT
TI06	Passed	Comunicarea a fost stabilită
TI07	Passed	Comunicarea a fost stabilită
TI08	Passed	Schimbul de mesaje se realizează corect
TI09	Passed	Conexiune realizată cu success
TI10	Passed	Comunicarea a fost stabilită

Tabel 4.1: Raport de testare

5 Concluzii

5.1 Sinteza principalelor idei din lucrare

Prin această lucrare se realizează o analiză detaliata a caracteristicilor necesare pentru realizarea unei aplicații dedicate sportului, care să îndeplinească nevoile unei game cât mai largi de utilizatori, fiecare având nevoi și cerințe diferite. O aplicație dedicată sportului reprezintă pentru mulți oameni o sursă de încredere, fiind responsabilă pentru deservirea unui conținut sportiv de calitate. Metrica principală folosită pentru evaluarea unei aplicații de fitness este progresul utilizatorilor, fiind nevoie de mecanisme care atestă acest progres, precum istoricul și statisticile despre conținutul sportiv consumat.

Un punct forte al platformei de coaching online Fitstack este capacitatea de gestionare a unui număr mare de utilizatori, datorită arhitecturii sale bazate pe microservicii. Toleranța la defecte și scalabilitatea sunt puncte cheie pentru o aplicație de fitness, deoarece se dorește disponibilitatea pentru un număr cât mai mare de utilizatori, fiind un factor esențial în încasările organizației.

Platforma Fitstack include o aplicație cross-platform, fiind disponibilă pentru sistemele de operare Android și iOS. Popularitatea lor este factorul decisiv pentru care s-a optat pentru dezvoltarea unei aplicații multi-platformă. De asemenea, Fitstack integrează antrenorii în contextul aplicației, oferind o platformă prin intermediul căreia să introducă conținut sportiv în mod intuitiv și eficient.

Utilizatorii beneficiază de o experiență plăcută în aplicație, aceasta incluzând concepte de gamification care oferă o notă de competitivitate ce va avea efecte pozitive asupra progresului utilizatorilor.

Astfel, platforma de coaching online Fitstack beneficiază de o gamă largă de funcționalități, menite să ofere utilizatorilor o aplicație de fitness robustă în care pot avea încredere.

5.2 Direcţii pentru continuarea cercetării

Platforma de coaching online poate fi comparată cu aplicațiile de pe piața actuală în ceea ce privește funcționalitățile disponibile utilizatorilor, fiind necesare mecanisme de publicitate pentru a atrage potențiali utilizatori.

Aplicația beneficiază de posibilitatea realizării antrenamentelor, oferind ca suport pentru utilizatori poze și videoclipuri adiționale pentru a corecta modul de efectuare a exercițiilor. O alternativă suplimentară ar putea fi opțiunea de a intra în camere de fitness unde un instructor va dirija în timp real modul de execuție pe parcursul antrenamentului.

Deși există posibilitatea de comunicare între antrenor și client, aceștia pot transmite doar mesaje text. O îmbunătățire adusă platformei ar fi posibilitatea utilizatorilor de a trimite conținut media, precum poze, videoclipuri, înregistrări și chiar opțiunea de apel video. Toate aceste funcționalități ar putea crește calitatea comunicării între antrenori și clienți, adăugând aplicației un serviciu de mesagerie complet și cuprinzător.

Pentru sugerarea celor mai bune antrenamente și programe de fitness, implementarea unui model de învățare automată care să ofere recomandări utilizatorilor ar fi foarte utilă în cadrul aplicației. Utilizatorii vor primi recomandări în funcție de conținutul sportiv pe care l-au urmărit până în prezent. Această funcționalitate ar crește productivitatea utilizatorilor, deoarece cele mai potrivite opțiuni vor fi afișate acestora pe pagina principală.

Pentru un sportiv, un factor mai important decât planul de antrenament este nutriția. Adăugarea sugestiilor și sfaturilor nutriționale ar putea fi un mare plus pentru platforma de fitness. De asemenea, ar putea fi incluși nutriționiști care să furnizeze planuri de dietă pentru diferite scopuri, precum slăbire, creștere în masă musculară, menținere, etc.

Bibliografie

[1] Amazon Web Services. *Unit Testing Overview*. https://aws.amazon.com/what-is/unit-testing/. Accessed: 2024-06-17.

[2] Auth0. *Introduction to JSON Web Tokens*. Accessed: 2024-06-12. 2024. URL: https://jwt.io/introduction.

[3] cert-manager. *ACME Issuer Configuration*. Cert-manager Documentation. URL: https://cert-manager.io/docs/configuration/acme/ (visited on 06/03/2024).

[4] *Cert-Manager Documentation*. https://cert-manager.io/docs/. Accessed: 2024-06-03.

[5] Docker. *Docker Hub Documentation*. https://docs.docker.com/docker-hub/. Accessed: 2024-05-31. 2024.

[6] Docker Documentation. *Docker Guides*. Accessed: 2024-06-04. 2024. URL: https://docs.docker.com/guides/.

[7] Encryption Consulting LLC. *Stages in a Certificate's Lifecycle*. https://www.encryptionconsulting.com/education-center/stages-in-a-certificates-lifecycle/. Accessed: 2024-06-03.

[8] Facebook. *React Documentation*. https://legacy.reactjs.org/docs/getting-started.html. Accessed: 2024-06-12. 2024.

[9] Flux CD Project. *Flux: The GitOps family of continuous delivery tools for Kubernetes*. [Online; accessed 31-May-2024]. 2024. URL: https://fluxcd.io/flux/.

[10] GeeksforGeeks. *Difference between Top-down and Bottom-up Integration Testing*. https://www.geeksforgeeks.org/difference-between-top-down-and-bottom-up-integration-testing/. [Accessed: June 18, 2024].

[11] GeeksforGeeks. *Integration Testing - Software Engineering*. https://www.geeksforgeeks.org/software-engineering-integration-testing/. Accessed: 2024-06-17.

[12] GeeksforGeeks. *Monolithic vs Microservices Architecture*. https://www.geeksforgeeks.org/monolithic-vs-microservices-architecture/. Accessed: 2024-06-16.

[13] GitHub. *GitHub Actions Documentation*. https://docs.github.com/en/actions. Accessed: 2024-06-17. n.d.

[14] Google. *Flutter Documentation*. https://docs.flutter.dev/. Accessed: 2024-06-10. 2024.

[15] Red Hat. *14 Software Architecture Patterns*. https://www.redhat.com/architect/14-software-architecture-patterns. Accessed: 2024-06-05.

[16] Kubernetes. *Kubernetes*. Accessed: June 4, 2024. URL: https://kubernetes.io/docs/home/.

[17] Let's Encrypt. *Documentation*. Accessed: 2024-06-03. 2024. URL: https://letsencrypt.org/docs/.

[18] Microsoft. *Microsoft Azure Documentation*. https://learn.microsoft.com/en-us/azure/?product=popular. Accessed: 2024-06-17.

[19] Microsoft. *Azure Monitor Data Platform Metrics*. Accessed: 2024-06-15. 2024. URL: https://learn.microsoft.com/en-us/azure/azure-monitor/essentials/data-platform-metrics.

[20] Microsoft Azure. *Azure Kubernetes Service (AKS) Documentation*. Accessed: 2024-06-02. 2024. URL: https://learn.microsoft.com/en-us/azure/aks/.

[21] Microsoft Azure. *Network observability in Azure Kubernetes Service (AKS)*. Accessed: 2024-06-02. 2024. URL: https://learn.microsoft.com/en-us/azure/aks/network-observability-byo-cli?tabs=non-cilium.

[22] Microsoft Azure. *Socket.IO overview in Azure Web PubSub*. Accessed: 2024-06-03. 2024. URL: https://learn.microsoft.com/en-us/azure/azure-web-pubsub/socketio-overview.

[23] Microsoft Azure. *Domain delegation to Azure DNS*. Accessed: Data accesării. Anul accesării. URL: https://learn.microsoft.com/en-us/azure/dns/dns-domain-delegation.

[24] MinIO Documentation. *MinIO Documentation for Kubernetes*. https://min.io/docs/minio/kubernetes/upstream/index.html. Accessed: 2024-06-16.

[25] MongoDB Documentation. *MongoDB Documentation*. https://www.mongodb.com/docs/. Accessed: 2024-06-16.

[26] Node.js. *Node.js v20.3.0 Documentation*. https://nodejs.org/docs/latest/api/. Accessed: 2024-06-05. 2024.

[30] Pallets Projects. *Flask Documentation*. https://flask.palletsprojects.com/en/3.0.x/. Accessed: 2024-06-19.

[31] *PostgreSQL Documentation*. Accessed: 2024-06-16. 2023. URL: https://www.postgresql.org/docs/current/.

[32] Python Software Foundation. *Python Documentation*. https://docs.python.org/3/. Accessed: 2024-06-19.

[33] Red Hat. *What is GitOps?* https://www.redhat.com/en/topics/devops/what-is-gitops. Accessed on 2024-06-01. Accessed 2024.

[34] Garage Gym Reviews. *Best Workout Apps*. https://www.garagegymreviews.com/best-workout-apps. Accessed: 2024-06-13. 2024.

[35] scikit-learn contributors. *scikit-learn: Machine Learning in Python*. https://scikit-learn.org/stable/. Accessed: 2024.

[36] Socket.IO. *Socket.IO Documentation (Version 4.x)*. https://socket.io/docs/v4/. Anul accesării.

[37] Sparkbox Foundry. *Compare Common Continuous Integration and Continuous Deployment Tools: Jenkins, CircleCI, GitHub Actions, AWS Pipeline*. https://sparkbox.com/foundry/compare_common_continuous_integration_and_continuous_deployment_tools_jenkins_circleci_github_actions_aws_pipeline. Accessed: June 1, 2024. 2024.

[38] Spring. *Spring Cloud*. URL: https://spring.io/projects/spring-cloud.

[39] Spring. *Spring Data JPA*. URL: https://spring.io/projects/spring-data-jpa.

[40] Spring. *Spring Security*. URL: https://spring.io/projects/spring-security.

[42] Spring Boot Team. *Spring Boot Documentation*. Accessed: June 4, 2024. VMware, Inc., 2024. URL: https://docs.spring.io/spring-boot/documentation.html.