**CAPITOLUL 1. INTRODUCERE**

**1.1 Situatia actuala si motivatia temei**

Fenomenul cel mai reprezentativ pentru societatea actuala este dezvoltarea accelerata a tehnologiei care a inceput acuma cateva zeci de ani, deoarece aceasta a devenit un element foarte important in viata fiecarui om. Astfel in ziua de astazi, dispozitivele electronice ne fac viata mai usoara si mai comoda avand posibilitatea de a accesa si trimite informatii cu ajutorul a catorva clickuri sau atingeri de ecran.

Cu timpul, companiile s-au adaptat treptat la acest stil de viata al oamenilor, devenind in cele din urma un standard ca acestea sa aiba o prezenta online intr-un fel sau altul, fie prin existenta unor website-uri pentru clienti prin care acestia pot lua contact cu firma sau se pot informa, fie prin exista unor platforme destinate angajatilor care le permite acestora sa gestioneze si sa automatizeze anumite elemente cheie din cardul business-ului. In general companiile sunt dispuse sa investeasca tot mai mult in dezvoltarea unor astfel de platforme pe masura, iar cele care aleg sa nu investeasca in domeniul online pierd o multime de beneficii fiind depasite de catre ceilalti competitori.

Un exemplu extrem de intalnit este prezenta unei aplicatii de e-commerce pentru magazine.

Cu ajutorul unei astfel de aplicatii, se pot gestiona automat stocuri si vanzari, eliminandu-se erorile umane daca aplicatia este proiectata corect si respecta cerintele impuse. De altfel este important ca navigarea aplicatiei sa fie intuitiva si usoara. De asemenea un site cu un aspect neplacut sau neingrijit le poate inspira utilizatorilor nesiguranta si neincredere pierzandu-se astfel posibili clienti in ciuda serviciilor oferite mai ales in cadrul unor firme mai putin cunoscute.

In concluzie o platforma online este o nevoie pentru aproape orice business, deoarece se economiseste timp si bani, se elimina erorile umane din managementul contabilitatii si logisticii si le ofera posibililor clienti oportunitatea de a apela la serviciile firmei rapid si usor.

**1.2 Scopul lucrarii**

Se urmareste dezvoltarea unei platforme online special concepute pentru management-ului unui magazin care comericalizeaza articole sportive pentru ski si snowboarding.

Clientii, la randul lor, au la dispozitie o interfata intuitiva si prietenoasa prin care pot vizualiza si filtra produsele dupa mai multe criterii, pe care le pot salva ulterior intr-o lista de produse favorite sau intr-un cos de cumparaturi. Aplicatia le pe permite utilizatorilor sa plaseze comenzi si sa cumpere produsele dorite, deci implicit este prezent si un sistem de autentificare.

Anumiti utilizatori autentificati si autorizati cu rol de administrator au posibilitatea de a monitoriza si modifica stocurile produselor si de a gestiona comenzi prin intermediul unei interfete diferite.

Aceasta aplicatie le ofera o solutie eficienta si convenabila atat administratorilor cat si clientilor. Prin intermediul acestei aplicatii software, se pot aduce imbunatatiri semnificative in gestionarea elementelor de logistica si in relatia cu clientii, crescanda satisfactia acestora si vanzarile afacerii.

**1.3 Structura lucrarii**

Lucrarea este impartita in X capitole principale .

Capitolul 1 – se prezinta motivatia temei, scopul lucrarii si cateva concepte generale

Capitolul 2 –se vor analiza cerintele sistemului informatic folosind diagrame

Capitolul 3 – se va proiecta aplicatia si vor fi prezentate cateva concepte importante din diferite perspective folosind diagrame

Capitolul 4 – se vor prezenta detalii tehnice legate de implementarea propriuzisa a aplicatiei

**1.4 Baze de date – concepte generale**

**1.4.1 – Definitia bazei de date**

O baza de date este o colectie organizata de informatie, de obicei localizata pe un dispozitiv de stocare. In general o baza de date este controlata de un sistem de gestiune a bazelor de date (DBMS – database management system). Impreuna, baza de date, DBMS-ul si aplicatiile asociate cu acestea formeaza un sistem informational cu baze de date.

In functie de cerintele care trebuie satisfacute, pot exista o multime de tipuri de baze de date, iar cateva dintre acestea sunt:

1. B.D. relationale – cel mai intalnit tip de baze de date si cel folosit in aplicatie
2. B.D. non-relationale – alt tip comun de baza de date
3. B.D. JSON
4. B.D. orientate pe obiecte

**1.4.2 – Baze de date centralizate si distribuite**

In etapa de proiectare a unui sistem informatic cu baze de date, trebuie analizate cerintele sistemului si caracteristicile fiecarui tip de baza de date.

Principala diferenta intre cele doua categorii de baze de date este ca in baza de date centralizata, informatiile sunt manipulate si stocate intr-o singura locatie, pe cand intr-o baza de date distribuita(decentralizata) informatiile se gasesc in mai multe locatii.

**1.4.2.a – Avantajele unei baze de date centralizate**

Datele se afla pe un singur dispozitiv. Orice informatie poate fi obtinuta, adaugata sau modificata rapid deoarece totul este intr-un singur loc.

Securitatea datelor este usor de intretinut deoarece toate datele sunt stocate intr-o singura locatie iar masurile de securitate pot fi concentrate pe singurul punct de acces. In cazul pierderii de date acestea pot fi recuperate daca exista o copie de rezerva actualizata periodic si stocata pe un alt dispozitiv.

Modelul datelor este simplu de proiectat in comparatie cu o baza de date distribuita. Datele sunt consistente datorita relatiilor stabilite intre tabele. In cazul unei baze de date dristribuite, datele sunt implicit mult mai complexe si mai greu de manipulat.

Redundanta minima- acest avantaj este o urmare a modelului simplu al datelor. Informatia redundanta este cea care este intalinita in doua sau mai multe locuri. Sansele de aparitie a acestora sunt proportionale cu complexitatea modelului datelor.

**1.4.2.b Dezavantajele unei baze de date centralizate**

Limitari din cauza vitezei- daca sunt multi utilizatori care acceseaza acelasi set de informatii intr-un timp scurt, pot aparea probleme de performanta semnificative, fenomen care nu apare atat de des intr-o baza de date decentralizata.

Daca exista probleme legate de sistemul care detine baza de date, aceasta poate fi inaccesibila pana la remedierea problemelor, lucru rezolvat in cadrul bazei de date distribuite: daca o locatie “cade”, celelalte locatii ii vor indeplinii functia, iar serviciul corespunzator functioneaza in continuare.

Riscul distrugerii informatiei- in cazul in care se intampla ceva cu sistemul fizic, exista posibilitatea ca toate datele stocate sa fie pierdute pentru totdeauna daca nu a fost facuta o copie de rezerva a datelor inaintea incidentului.

**CAPITOLUL 2. CERINTELE SISTEMULUI**

Aplicatia conceputa va trebui sa indeplineasca cateva cerinte minime pentru a-si atinge scopul. Programatorul trebuie sa urmeze aceste cerinte pentru ajunge la produsul final dorit.

La final aplicatia trebuie sa indeplineasca functiile de baza specifice oricarei platfome de eCommerce si anume:

* Autentificare si autorizare pentru fiecare utilizator in parte ( atat pentru clienti cat si pentru angajati).
* Efectuarea de operatii CRUD (Create Retrieve Update Delete) asupra datelor prin prezenta unor interfete specializate.
* Stocarea articolelor sportive in baza de date trebuie sa fie cat mai detaliata. Echipamentele destinate sporturilor de iarna sunt foarte diversificate iar stocarea acestora trebuie sa reflecte realitatea cat mai bine.
* Avand in vedere modelul de date, trebuie implementata o functie de filtrare pe masura, cat mai amanuntita.
* Interfata clientilor trebuie sa fie simpla si intuitiva. Trebuie sa se poata ajunge in orice punct al aplicatiei in cel mult 3 sau 4 clickuri.
* Clientii trebuie sa aiba posibilitatea de a salva produse intr-o lista de favorite.
* Clientii sa poata sa initializeze comenzi pe baza unui cos de cumparaturi.

Cateva dintre aceste cerinte pot fi descrise cu ajutorul unor diagrame.

In continuare se va prezenta diagrama cazurilor de utilizare pentru un site de ecommece.

**Diagrama cazurilor de utilizare**

Orice program software trebuie sa interactioneze cu exteriorul fie cu alti oameni, fie cu alte sisteme.

Diagrama cazurilor de utilizare (use case diagram) este o diagrama UML, utilizata in faza de proiectare si analiza a unui sistem software pentru a defini functionalitatile sistemului si pentru a clarifica cerintele si nevoile impuse atunci cand acestea trebuiesc transmise altor persoane intr-un mod usor de inteles, independent de tehnologia utilizata in implementarea aplicatiei.

Programatorii, pot urmarii aceste diagrame pe parcursul intregului proces de dezvoltare pentru a se asigura ca sistemul este construit conform specificatiilor si cerintelor initiale.

O diagrama use case are cateva componente specifice: actori, use case-uri si relatiile dintre ele.

**Actorii**

Acestia reprezinta niste entitati externe care pot avea roluri diferite si care interactioneaza cu sistemul descris. Actorii pot fii niste simplii utilizatori, sisteme software sau dispozitive hardware si se reprezinta grafic printr-o persoana.

Shape

Description automatically generated with low confidence

Figura 2.1. Reprezentarea grafica a unui actor

**Use case-uri**

Acestea sunt actiunile pe care sistemul le ofera utilizatorilor. Un use case descrie o functionalitate a sistemului, relevanta pentru actor insa nu se specifica alte detalii precum modul in care se face aceasta actiune si nici modul in care aceasta a fost implementata.

Grafic, un use case se descrie printr-o elipsa in interiorul careia apare numele acestuia.

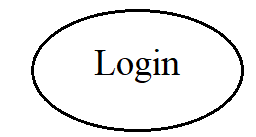


Figura 2.2. Reprezentarea grafica a unui use-case care descrie actiunea de logare

Numele cazului de utilizare trebuie sa fie unic, simplu si sa descrie o actiune sau o secventa de actiuni usor de inteles pentru utilizatori.

**Relatii**

Intre use case-uri si actori se stabilesc relatii care sunt importante pentru descrierea modului in care sitemul functioneaza si cum interactioneaza actorii cu acesta.

Exista cateva tipuri de relatii: asociere, dependenta si generalizare.

**Relatia de asociere** se poate defini intre use case-uri sau intre actori si use case-uri. Aceasta sugereaza comunicarea intre componentele pe care le uneste si se poate reprezenta grafic printr-o linie.

Shape, rectangle

Description automatically generatedFigura 2.3. Reprezentarea grafica a unei relatii de asociere intre doua use case-uri

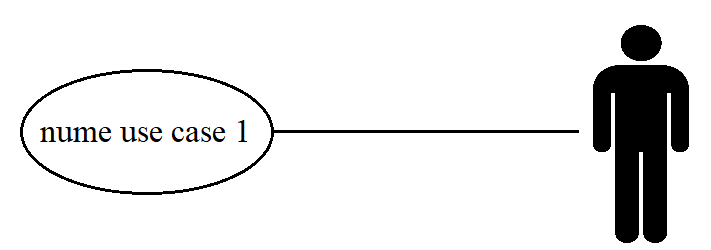


Figura 2.4. Reprezentarea grafica a unei relatii de asociere intre un use case si un actor

**Relatia de dependenta** se poate stabili intre doua cazuri de utilizare si este de doua tipuri: include si extend.

Relatia de tip **include** este folosita atunci cand un caz de utilizare depinde de alt caz de utilizare pentru a functiona.

In urmatorul exemplu, use case-ul B este inclus de catre use case-ul A. Cu alte cuvinte, atunci cand se executa cazul A si cazul B se executa inclusiv.

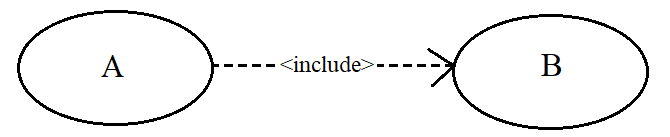


Figura 2.5. Depeneta de tip **include**

Relatia de tip **extend** se foloseste atunci cand cele doua cazuri de utilizare sunt de sine statatoare adica nu depind unul de calalalt. In exemplul urmator, executia use case-ului A determina daca B se executa sau nu.

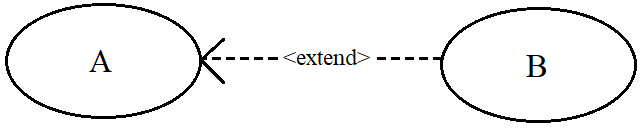


Figura 2.6. Dependenta de tip **extend**

**Relatia de generalizare** este folosita pentru a arata ca un caz de utilizare este o varianta sau specializare a altui caz de utilizare iar spre deosebire de celelalte tipuri de relatii, poate avea loc numai intre doi actori sau intre doua cazuri de utilizare. Un actor va mosteni rolul altui actor, impreuna cu cazurile de utilizare atribuite acestuia.

A picture containing screenshot, circle, black and white

Description automatically generated

Figura 2.7 – Diagrama Use Case pentru un magazin online

In cele ce urmeaza, se vor explica componentele diagramei si semnificatia cazurilor de utilizare:

-Avem 4 actori: Customer, Existing User,Guest User si Admin

-Existing User este orice utilizator care este prezent in baza de date. Acesta

poatevizualiza catalogul de produse, se poate autentifica cu contul propriu, poate adauga produse in cosul de cumparaturi si poate plasa o comanda in baza cosului de cumparaturi. De asemena isi poate verifica si modifica detaliile personale si poate vedea un istoric al propriilor comenzi.

- Guest User este orice utilizator care nu este inregistrat in baza de date. Acesta se poate inregistra si poate vizualiza catalogul de produse. Se evidentiaza faptul ca un simplu user vizitator are privilegii limitate in comparatie cu un user prezent in baza de date

- Orice Customer este considerat un Guest User pana in momentul in care

-Customer este orice utilizator al site-ului. Acesta mostenste atributiile Guest User-ului si Existing User-ului.

-Admin – un admin este un actor care are alte privilegii fata de orice Customer. Acesta poate edita catalogul de produse si poate vizualiza datele fiecarui Customer (inclusiv istoriucul de comenzi).

**CAPITOLUL 4 – Implementarea aplicatiei**

Aplicatia este constituita din 3 parti componente principale:

1. Frontend-ul : reprezinta interfata prin care utilizatorii (atat administratorii cat si clientii) iau contact cu aplicatia. S-a folosit React, o biblioteca JavaScript folosita pentru dezvoltarea de interfete de utilizator si Typescript , care exinde sintaxa limbajului JavaScript, permitandu-se definirea tipurilor de date in mod static si nu dinamic precum in JavaScript-ul simplu. Framework-ul React a fost folosit pentru creearea unor aplicatii precum Facebook, Netflix, Instagram, Discord si multe altele.

2. Backend-ul: reprezinta “creierul” aplicatiei. In functie de datele primite de la partea de frontend, acesta introduce informatii in baza de date sau le extrage si le trimite sub forma de raspuns inapoi catre frontend. Am folosit Java Spring Boot. Acestea este un framework al limbajului Java care ofera suport pentru dezvoltarea de aplicatii web. Aici am definit modelele entitatiilor bazei de date si API-ul prin care se face comunicarea cu frontend-ul. De asemenea am implementat si o parte de autentificare si autorizare a utilizatorilor prin Spring Security care este tot un framework al limbajului Java.

3. Baza de date: S-a folosit o baza de date relationala MySQL, unde am salvat toate informatiile aplicatiei, pornind de la tipurile de produse la datele utilizatorilor.