Studiu individual (Raport)

PROIECT PRELIMINAR

Tema: Program de evidenţă al vânzărilor unui magazin online (carţi)

Autor: Grosu Daniel

1. Introducere

Context și justificare
Digitalizarea sectorului de vânzări a condus la o creștere semnificativă a magazinelor online, inclusiv a celor care comercializează cărți. Totuși, gestionarea eficientă a comenzilor, stocurilor și rapoartelor necesită un software dedicat care să minimizeze erorile și să automatizeze procesele complexe.

Scopul proiectului
Dezvoltarea unui software modular, personalizabil și ușor de utilizat pentru gestionarea completă a procesului de vânzare într-un magazin online de cărți, optimizat pentru scalabilitate și extindere.

Obiective

1. Automatizarea procesului de comandă și stoc.
2. Crearea unui sistem de generare automată a rapoartelor.
3. Integrarea cu sisteme de plăți și notificări automate.
4. Oferirea unei interfețe prietenoase pentru utilizator.

Beneficii anticipate

Reducerea timpului de gestionare a comenzilor.
Creșterea preciziei în procesarea datelor.
Îmbunătățirea experienței utilizatorului.

---

2. Specificațiile

2.1 Posibilități multi-dimensionale
Programul va include următoarele caracteristici:

Gestionarea comenzilor: Înregistrarea automată, urmărirea stării comenzilor.
Monitorizarea stocurilor: Actualizări automate, alerte de stoc.
Rapoarte de vânzări: Personalizate pentru diferite perioade (zilnice, lunare).
Gestionarea utilizatorilor: Roluri definite (administrator, angajat).
Notificări: E-mailuri automate pentru clienți.
Securitate: Protecție avansată a datelor clienților și comenzilor.

2.2 Motoare și module

Backend: Sistem server-side care rulează logica afacerii.
Frontend: Interfața utilizatorului bazată pe tehnologii moderne (React/Angular).
Baza de date: MySQL/PostgreSQL pentru stocarea datelor.
Module principale:

- Gestionare produse: Adăugare, editare, ștergere cărți.
- Comenzi: Evidența comenzilor plasate, procesarea acestora.
- Notificări: Sistem de alertare automată pentru clienți și echipa internă.

2.3 Schema interacțiunii

Diagrama va include următoarele elemente:

1. **Clientul** interacționează cu frontend-ul prin browser sau aplicație mobilă.
2. Frontend-ul comunică cu **backend-ul** pentru procesare.
3. Backend-ul accesează **baza de date** pentru actualizarea stocurilor sau validarea comenzilor.
4. Modulul de notificări trimite mesaje automate utilizatorilor.

---

3. Roadmap-ul

3.1 Faze de evoluție

1. Analiză și planificare: Identificarea cerințelor.
2. Design-ul sistemului: Crearea arhitecturii și prototipului.
3. Dezvoltare: Implementarea modulelor și funcționalităților.
4. Testare: Validarea funcționalității și securității sistemului.
5. Lansare: Implementarea soluției și livrarea către client.

3.2 Etape de implementare

1. Crearea bazei de date și configurarea structurii de stocare.
2. Dezvoltarea backend-ului pentru logica afacerii.
3. Construirea interfeței utilizator (frontend).
4. Integrarea modulelor: Comenzi, notificări, rapoarte.

3.3 Schema de implementare

Vom include o diagramă care descrie interacțiunile între module.

3.4 Graficul de implementare

Graficul Gantt poate fi utilizat pentru a reprezenta durata fiecărei etape.

4. Unitatea de implementare

Această secțiune detaliază echipa, resursele și suportul tehnic necesar pentru implementarea proiectului.

4.1 Echipa de proiect

Echipa este alcătuită din specialiști cu competențe specifice pentru asigurarea succesului implementării:

Manager de proiect: Supraveghează progresul proiectului și alocarea resurselor.
Analist de business: Identifică cerințele funcționale și non-funcționale.
Dezvoltatori backend: Se ocupă de logica aplicației și conectarea cu baza de date.
Dezvoltatori frontend: Creează interfața utilizatorului.
Testeri QA: Realizează teste pentru identificarea erorilor.
Administrator de sistem: Configurează și menține infrastructura tehnică.

4.2 Instituții pilot de testare

Pentru validarea sistemului, o entitate pilot (magazin online fictiv) va fi utilizată:

Scop: Simularea proceselor reale de vânzare, cu utilizatori fictivi.
Rezultat: Identificarea punctelor slabe și îmbunătățirea aplicației înainte de lansarea oficială.

4.3 Tehnologiile aferente

Pentru implementare, utilizăm:

Limbaje de programare:

- Backend: Python (framework Django/Flask).
- Frontend: JavaScript (framework React sau Angular).

Baze de date: PostgreSQL/MySQL pentru stocarea informațiilor despre cărți, comenzi și clienți.
Sisteme de integrare: Stripe/PayPal pentru plăți, Twilio/SendGrid pentru notificări.
Infrastructură: Hosting pe Amazon Web Services (AWS) sau Google Cloud Platform (GCP).

4.4 Mentenanța sistemului software

Suport tehnic:

- Disponibil 24/7 pentru incidente critice.
- Actualizări lunare pentru îmbunătățiri de securitate.

Procesul de actualizare:

- Versiuni noi lansate în funcție de cerințele clienților.

Monitorizare:

- Instrumente precum New Relic sau DataDog pentru urmărirea performanței aplicației.

---

5. Spațiul aplicației

Această secțiune descrie conceptele și arhitectura tehnică a aplicației.

5.1 Concepte specifice

1. **Scalabilitate**:
   - Sistemul trebuie să suporte creșteri bruște în volumul de comenzi.
   - Soluții cloud pentru extinderea resurselor.
2. **Ușurința în utilizare**:
   - Interfață intuitivă, accesibilă și bine organizată.
3. **Securitate**:
   - Protecția datelor sensibile prin criptare și autentificare.

5.2 Arhitectura proceselor

Aplicația va fi organizată pe 3 niveluri principale:

Nivelul prezentare: Interfața utilizatorului.
Nivelul logicii de afaceri: Gestionarea funcționalităților (plasarea comenzilor, calcularea stocurilor).
Nivelul bazei de date: Stocarea permanentă a informațiilor despre produse, clienți și vânzări.

Fluxul proceselor:

1. Clientul adaugă produse în coșul de cumpărături.
2. Backend-ul verifică stocul disponibil.
3. Comanda este plasată și datele sunt stocate în baza de date.
4. Sistemul generează un e-mail de confirmare pentru client.

5.3 Domeniul Informatic utilizat

Tehnologii cloud pentru scalabilitate.
API-uri RESTful pentru comunicarea între componente.
Design modular pentru adăugarea ușoară a funcționalităților viitoare.

---

6. Obiective și activități

6.1 Obiective specifice

1. Crearea unei aplicații funcționale și stabile.
2. Automatizarea proceselor cheie (stocuri, comenzi, notificări).
3. Reducerea timpului necesar pentru procesarea unei comenzi.
4. Îmbunătățirea raportării prin analize detaliate și vizualizări grafice.

6.2 Activități specifice

Activitate	Descriere	Durată estimată
Analiza cerințelor	Colectarea informațiilor de la client și identificarea cerințelor.	2 săptămâni
Proiectarea arhitecturii	Realizarea schemelor sistemului și a prototipurilor.	3 săptămâni
Dezvoltarea modulului stocuri	Implementarea funcției de gestionare a stocurilor și notificare pentru stoc redus.	2 săptămâni
Dezvoltarea modulului comenzi	Crearea funcționalităților de plasare, procesare și urmărire a comenzilor.	2 săptămâni
Integrarea notificărilor	Implementarea trimiterii automate de e-mailuri și notificări interne.	1 săptămână
Testare și optimizare	Validarea aplicației, optimizarea	2 săptămâni

	performanței și corectarea erorilor.	

6.3 Riscuri specifice

Risc	Impact potențial	Soluție
Lipsa de resurse financiare	Întârzierea livrării sistemului.	Planificarea bugetului și prioritizarea funcțiilor.
Timp de răspuns lent al aplicației	Nemulțumirea utilizatorilor și pierderea clienților.	Optimizarea bazei de date și utilizarea caching-ului.

Link la git:
https://github.com/GrosuDaniel/ManagamentProject