vUNIVERSITATEA DIN BUCUREȘTI

FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ

**<Restaurant Management Application>**

**RAPORT DE ANALIZĂ ARHITECTURALĂ**

**<Cuciureanu Dragoș-Adrian>**

2024

Cuprins

[1. Scopul aplicației 3](#_Toc170426210)

[2. Aria de acoperire a aplicației 3](#_Toc170426211)

[3. Grupurile de interese 4](#_Toc170426212)

[4. Colectarea și analiza cerințelor 4](#_Toc170426213)

[5. Prioritizarea cerințelor 6](#_Toc170426214)

[6. Specificații de analiză arhitecturală (selecție) 6](#_Toc170426215)

[7. Tactici arhitecturale 9](#_Toc170426216)

[8. Specificațiile de proiectare 9](#_Toc170426217)

[9. Perspective arhitecturale 11](#_Toc170426218)

[a. Perspectiva de utilizare 11](#_Toc170426219)

[b. Perspectiva asupra datelor 11](#_Toc170426220)

[c. Perspectiva structurală 12](#_Toc170426221)

[d. Perspectiva comportamentală 13](#_Toc170426222)

[10. Alegerea tehnologiilor 15](#_Toc170426223)

# Scopul aplicației

Descrierea tipului, misiunii și utilizatorilor (rolurilor) aplicației (max. ½ de pagină)

Aplicația “Restaurant Management Application” este de tipul organizării (management) și al serviciilor.

Misiunea este punerea la dispoziție a unei platforme atractive care să mărească numărul de comenzi cât și valoarea acestora.

Utilizatorii aplicației:

1. Clienții: Utilizatori ce doresc să facă o rezervare și vadă produsele de la un restaurant.
2. Personalul restaurantului (stafful): Utilizatorii ce se ocupă de gestionarea meniurilor, rezervărilor și a comenzilor.
3. Administratorii sistemului: Utilizatorii care asigură buna funcționare și climatul de încredere al sistemului.
4. Sistemul: Furnizează în baza datelor caracteristici și funcționalități avansate.

# Aria de acoperire a aplicației

Ce este și ce nu este aplicația. Include obiectivele principale ale aplicației scrise sub formă de user stories (max. ½ de pagină)

Aplicația software facilitează gestionarea meniurilor, locurilor pentru clienți, comenzilor și rezervărilor în restaurante și baruri. Această aplicație va permite personalului să gestioneze eficient întregul flux de lucru dintr-un restaurant sau bar, inclusiv gestionarea meniurilor, atribuirea locurilor pentru clienți, preluarea comenzilor și administrarea rezervărilor. O dată ce o comandă a fost efectuată, clienții pot plasa review-uri asupra produselor cumpărate.

Aplicația nu este o aplicație de management al comenzilor online, acestea fiind înafara scopului propus. Aplicația se ocupă de gestionarea unui singur restaurant, nu al unui conglomerat.

Obiectivele principale:

1. Clienții doresc să facă o rezervare și vadă produsele din meniu de la restaurant pentru realizarea misiunii.
2. Personalul restaurantului se ocupă de gestionarea meniurilor, rezervărilor și al comenzilor pentru realizarea misiunii.
3. Administratorii sistemului asigură buna funcționare și climatul de încredere al sistemului pentru realizarea misiunii.
4. Sistemul furnizează în baza datelor caracteristici și funcționalități avansate și la curent cu standardele industriei pentru realizarea misiunii.

# Grupurile de interese

Care sunt persoanele sau grupurile de persoane interesate de aplicație (inclusiv rolurile în aplicație)

Grupuri de interese:

1. Utilizatorii:
   1. Clienții: Aceștia doresc posibilitatea de a face rezervări și de a avea acces la meniurile restaurantelor și la informații despre locurile disponibile și funcționalitatea de a lăsa review-uri pentru produsele achiziționate.
   2. Personalul restaurantului: Sunt preocupați cu gestionarea eficientă a meniurilor și actualizarea acestora, alocarea și administrarea locurilor pentru clienți, gestionarea rezervărilor și preluarea și gestionarea comenzilor în timp real.
   3. Administratorii sistemului: Aceștia se ocupă de monitorizarea și menținerea funcționalității sistemului, asigurarea securității și protecției datelor și gestionarea drepturilor de acces și a rolurilor utilizatorilor.
2. Proprietarii și managerii restaurantelor: Grup ce dețin restaurantul ce are ca scop principal creșterea profitului prin creșterea numărului și valorii comenzilor prin creșterea eficienței operaționale, cât și prin reducerea costurilor în sectoarele unde este posibil.
3. Dezvoltatorii aplicației: Persoanele/echipele care proiectează, dezvoltă și întrețin aplicația prin implementarea cerințelor, asigurarea unei experiențe de utilizare cât mai plăcută și menținerea și actualizarea aplicației pentru a răspunde noilor cerințe și tehnologii.
4. Investitorii: Persoane sau entități care investesc în dezvoltarea și extinderea aplicației ce doresc maximizarea rentabilității investițiilor.

# Colectarea și analiza cerințelor

Se vor prezenta cerințele colectare, indicându-se la fiecare persoanele sau grupurile interesate și sursa de unde au fost colectate. Cerințele vor fi consolidate și grupate pe categorii: cerințe funcționale, cerințe de calitate (pe subcategorii relevante), constrângeri (pe subcategorii relevante) => Lista generală a cerințelor

Cerințele au fost colectate din următoarele surse:

* + - * 1. Interviuri și discuții simulate cu persoanele/grupurile interesate, printre care se numără clienții, personalul restaurantului, administratorii sistemului, proprietarii și managerii restaurantelor, dezvoltatorii aplicației și investitorii.
        2. Kimes, S. E. (2009, March 2). *How restaurant customers view online reservations*. <https://ecommons.cornell.edu/items/dbe97f30-aaf8-433e-a309-cc5693bb7c87>
        3. *The importance of appointment confirmation emails | TIMIFY*. (n.d.). <https://www.timify.com/en/blog/importance-of-sending-appointment-confirmation-emails-in-2023/>
        4. Bandhiya, I. (2024, January 3). *Restaurant table reservation apps to streamlining dining experience*. Eatance App. <https://eatanceapp.com/blog/restaurant-table-reservation-app-development-cost-features/>

Cerințe funcționale:

Sistemul trebuie sa ofere clienților posibilitatea de a descoperi vedea mesele nerezervate, cât și produse din meniul unui restaurant.

1. Utilizatorii au nevoie de o interfață de unde sa descopere mesele nerezervate și produsele pe care le au restaurantul pentru a putea face o alegere informata asupra realizării unei comenzi. [sursa 2]
2. Pentru a nu insera datele de fiecare data când clienții doresc să realizeze o rezervare, acestora li se oferă opțiunea de a-și crea un cont. În același timp, oferim utilizatorilor si opțiunea de a realiza o rezervare fără cont, pentru a nu îi obliga sa își creeze un cont si a le îmbunătăți experiența de utilizare a aplicației. [sursa 2]
3. Utilizatorilor le este oferita funcționalitatea de a lăsa un review asupra produselor de la un restaurant pe care le-au achiziționat. [sursa 4]
4. Clienții primesc o notificare bazată pe tipul de interacțiune realizată (rezervarea unei mese, plasarea unei comenzi sau lăsarea unui review), pentru o interacțiune bună. [sursa 3]
5. Clienții pot realiza plata comenzii prin intermediul unui secure payment din cadrul aplicației. [sursele 1, 4]

Sistemul trebuie sa ofere personalului restaurantului de a gestiona aplicația:

1. Proprietarul restaurantului trebuie sa poată crea noi mese și să poată modifica layoutul restaurantului. [sursa 4]
2. Personalul are nevoie de o funcționalitate prin care să creeze și să gestioneze produsele. [sursa 1]
3. Personalul are nevoie de o funcționalitate prin care să creeze și să gestioneze comenzile realizate de către clienți. [sursa 4]
4. Personalul are nevoie de o funcționalitate prin care să gestioneze rezervările asupra meselor din restaurant. [sursa 2]
5. Sistemul trebuie sa ofere posibilitatea managerilor de a face promoții anumitor produse pentru a mări șansele ca acestea să fie cumpărate. Promoțiile pot fi de natura: un procent reducere (de exemplu 20/30% reducere) sau mai multe la preț de mai puține (4 produse la preț de 3) pentru a încuraja venirea în grupuri mai mari și astfel comenzi mai mari. [sursele 1, 4]

Cerințe de calitate:

* + 1. Utilizabilitate: Sistemul trebuie să fie ușor de folosit pentru orice grup de clienți, oferind fluxuri intuitive care să simplifice procesul de rezervare a unei mese, ajutând la mărirea numărului de comenzi cât și valoarea acestora. [sursa 1]
    2. Disponibilitate: Utilizatorii trebuie să poată accesa aplicația în orice moment, pentru a putea realiza o rezervare asupra mesei dorite oricând. O aplicație disponibilă nonstop permite utilizatorilor realizeze o rezervare atunci când au timp, indiferent de programul personalului restaurantului, facilitând o interacțiune optimă cu aplicația și crescând numărul de comenzi. [sursa 4]
    3. Securitate: Datorită integrării serviciilor de plată în cadrul aplicației, este esențial să asigurăm un nivel ridicat de securitate pentru a proteja confidențialitatea datelor clienților. [sursa 4]
    4. Modificabilitate: Un sistem modificabil este proiectat având în vedere posibilitatea dezvoltării ulterioare cu un minim de efort. Nivelul de modificabilitate al aplicației este evaluat prin costurile asociate implementării de funcționalități noi sau extinderii celor existente. [sursa 1]
    5. Testabilitate: În contextul aplicației pentru managementul restaurantelor, un specialist în testare sau un dezvoltator ar trebui să poată verifica acuratețea rezultatelor, consistența răspunsurilor și performanța tuturor funcționalităților sistemului. [sursele 1, 4]

Constrângeri:

1. Dezvoltarea aplicației software este constrânsa de numărul mic de dezvoltatori, cât și de posibila experiența limitată a acestora. [sursa 1]

2. O alta constrângere relevanta o reprezintă bugetul inexistent, fapt ce limitează dezvoltarea soluției software spre tehnologii open-source si servicii gratuite. [sursa 1]

# Prioritizarea cerințelor

Din lista generală a cerințelor vor fi selectate minim 3 cerințe de calitate, cele mai relevante pentru realizarea misiunii aplicației => Lista redusă a cerințelor

Cele mai relevante cerințe de calitate pentru realizarea misiunii aplicației sunt utilizabilitatea, disponibilitatea, securitatea și modificabilitatea, deoarece acestea contribuie în mod direct și predominant la creșterea numărului și valorii comenzilor. Utilizabilitatea asigură o experiență intuitivă, facilitând interacțiunea utilizatorilor cu aplicația, în timp ce disponibilitatea permite accesul permanent la funcționalități acesteia, indiferent de programul restaurantului. Securitatea protejează datele sensibile ale utilizatorilor, menținând încrederea acestora, iar modificabilitatea permite adaptarea rapidă la noile cerințe și tehnologii, asigurând relevanța și competitivitatea aplicației pe termen lung în ecosistemul restaurantelor.

Astfel, cele mai relevante 4 cerințe de calitate pentru realizarea misiunii aplicației "Restaurant Management Application" ce vor fi dezvoltate pe parcursul raportului sunt:

* + - 1. Utilizabilitate
      2. Disponibilitate
      3. Securitate
      4. Modificabilitate

# Specificații de analiză arhitecturală (selecție)

Se vor specifica cerințele de calitate din lista redusă:

1. Prima dată sub formă de user stories, prin descompunere graduală, plecând de la misiunea aplicației și obiectivele principale.

Misiunea este punerea la dispoziție a unei platforme atractive care să mărească numărul de comenzi cât și valoarea acestora.

Ob1. Clienții doresc **să facă o rezervare** și vadă produsele din meniu de la restaurant pentru realizarea misiunii.

Ob1.1. Ca și client, vreau să pot accesa aplicația 24/7, astfel încât să pot face o rezervare chiar și în afara orelor de funcționare ale restaurantului pentru a mări numărul de comenzi. => cerința de calitate 2 (disponibilitate) sub formă de user story

Ob1.2. Ca și client, vreau să pot face o rezervare rapid la o anumită masă și la o anumită oră, astfel încât să îmi asigur un loc la restaurant în mai puțin de câteva minute pentru a mări numărul de comenzi. => cerința de calitate 1 (utilizabilitate) sub formă de user story

Ob1.3. Ca și client, vreau să primesc o notificare (de preferat pe email), atunci când realizez o rezervare, pentru o experiență de utilizare bună.

Ob2. Personalul restaurantului **se ocupă de** **gestionarea** meniurilor, rezervărilor și al **comenzilor** pentru realizarea misiunii.

Ob2.1. Ca și client, vreau să pot vedea produsele din meniu în aplicație în mod rapid și intuitiv pentru a mări valoarea comenzii.

Ob2.2. Ca si personal al restaurantului, vreau să pot adăuga produsul ales de client rapid și ușor la comanda sa, pentru a crește eficiența procesului și astfel și interacțiunea clientului.

Ob2.3. Ca și client, vreau să pot plăti nota într-un mod facil din aplicație într-un mod securizat ce-mi va păstra confidențialitatea datelor, pentru a reveni în viitor și sa cresc numărul de comenzi. => cerința de calitate 3 (securitate) sub formă de user story

Ob3. Administratorii sistemului asigură buna funcționare și climatul de încredere al sistemului pentru realizarea misiunii.

Ob4. Sistemul furnizează în baza datelor caracteristici și **funcționalități avansate și la curent cu standardele industriei** pentru realizarea misiunii.

Ob4.1. Ca și dezvoltator, vreau ca aplicația să utilizeze tehnologii și practici de dezvoltare moderne, astfel încât să pot implementa eficient noi funcționalități pentru a adapta aplicația la posibilele schimbări din industrie. => cerința de calitate 4 (modificabilitate) sub formă de user story

Ob4.2. Ca și dezvoltator, vreau să pot testa rapid și eficient noile funcționalități, astfel încât să asigur calitatea și stabilitatea aplicației.

1. A doua oară sub forma unor scenarii de calitate (se va determina cel puțin un scenariu relevant pentru fiecare cerință din lista redusă).

**QS1 - Utilizabilitate**

|  |  |
| --- | --- |
| Artefact | Interfață |
| Starea | Normală |
| Evenimentul declanșator | Utilizatorul selectează o masă pentru a face o rezervare la o anumită oră |
| Strategia de răspuns | Simplitate/Intuitivitate |
| Măsura răspunsului | Timpul de realizare al fluxului să fie mai mic de 1 min |

**QS2 - Disponibilitate**

|  |  |
| --- | --- |
| Artefact | Sistemul |
| Starea | Afectata, o solicitare a returnat o eroare |
| Evenimentul declanșator | Sistemul a primit o solicitare care a dus la un caz de eroare |
| Strategia de răspuns | Detecția si notificarea administratorului |
| Măsura răspunsului | Procentul erorilor detectate și remediate sa fie >= 99.5% |

**QS3 - Securitate**

|  |  |
| --- | --- |
| Artefact | Sistemul |
| Starea | Normală |
| Evenimentul declanșator | Utilizatorul efectuează plata pentru comanda sa |
| Strategia de răspuns | Prevenția atacurilor prin folosirea serviciilor securizate |
| Măsura răspunsului | Atacurile sa fie prevenite în 100% din cazuri |

**QS4 - Modificabilitate**

|  |  |
| --- | --- |
| Artefact | Sistemul |
| Starea | Normală |
| Evenimentul declanșator | Adăugarea unei noi funcționalități la aplicația software de management de restaurante |
| Strategia de răspuns | Locul din arhitectura unde se realizează modificările |
| Măsura răspunsului | Cost/efortul necesar pentru implementarea funcționalităților noi sa nu depășească bugetul (10% din profit sau 5000 lei/luna) |

# Tactici arhitecturale

Pentru fiecare dintre scenariile de calitate definite mai sus, se va alege și descrie tactica sau tacticile (soluțiile, metodele de implementare) arhitecturale adecvate.

T1. Utilizabilitate: Pentru a crea un mediu intuitiv, vom utiliza tactici avansate de UX, cum ar fi organizarea aplicației în pagini și formulare dedicate fiecărei funcționalități.

T2. Utilizabilitate: Interfața va ghida utilizatorul pe tot parcursul procesului, adoptând un model de tip „step wizard”, des întâlnit în aplicațiile moderne.

T3. Disponibilitate: Pentru a asigura o disponibilitate optimă, vom implementa mecanisme de capturare și gestionare a erorilor în cadrul claselor și vom folosi aplicații de monitorizare și analiză în timp real precum Grafana.

T4. Disponibilitate: În cazul apariției unei erori în cadrul aplicației software, vom notifica administratorul prin diverse metode (de exemplu, email) în cazul apariției unor probleme.

T5. Securitate: Vom menține confidențialitatea datelor utilizatorilor prin utilizarea unui furnizor terț de securitate, cum ar fi 3D Secure de la ING/BCR, pentru a asigura protecția tranzacțiilor.

T6. Modificabilitate: Vom anticipa posibile extinderi ale aplicației prin adoptarea principiilor SOLID și prin scrierea unui cod clar și bine structurat, facilitând astfel adăugarea de noi funcționalități.

# Specificațiile de proiectare

Specificațiile de analiză descrise sub formă de user stories se vor descompune mai departe în specificații de proiectare tot sub formă de user stories care să descrie tacticile arhitecturale adoptate pentru fiecare dintre scenariile de calitate.

US1. Utilizabilitate: Ca și client, vreau să pot face o rezervare rapid la o anumită masă și la o anumită oră, astfel încât să îmi asigur un loc la restaurant în mai puțin de câteva minute pentru a mări numărul de comenzi.

US1.1. Interfețe intuitive: Ca și client, vreau să pot face o rezervare rapid la o anumită masă și la o anumită oră utilizând o interfață intuitivă organizată în pagini și formulare dedicate, astfel încât să îmi asigur un loc la restaurant în mai puțin de câteva minute pentru a mări numărul de comenzi./

US1.2. Step Wizard: Ca și client, vreau să fiu printr-un model de tip „step wizard” pentru a face o rezervare la o anumită masă și la o anumită oră astfel încât să îmi asigur un loc la restaurant în mai puțin de câteva minute pentru a mări numărul de comenzi.

US2. Disponibilitate: Ca și client, vreau să pot accesa aplicația 24/7, astfel încât să pot face o rezervare chiar și în afara orelor de funcționare ale restaurantului pentru a mări numărul de comenzi.

US2.1. Sisteme de monitorizare: Ca și administrator, vreau ca sistemul să fie monitorizat în timp real de sisteme robuste precum Grafana pentru orice eroare ce poate duce la downtime în cadrul aplicației pentru a maximiza timpul în care aplicația este disponibilă.

US2.2. Sisteme de monitorizare: Ca și administrator, vreau ca sistemul mă notifice imediat ce a fost monitorizată o eroare ce poate duce la downtime în cadrul aplicației pentru a maximiza timpul în care aplicația este disponibilă.

US3. Securitate: Ca și client, vreau să pot plăti nota într-un mod facil din aplicație într-un mod securizat ce-mi va păstra confidențialitatea datelor, pentru a putea reveni în viitor și să cresc numărul de comenzi.

US3.1. Secure payment provider: Ca și client, vreau să pot plăti nota într-un mod facil din aplicație prin intermediul unui secure payment provider (precum 3D Secure), ce-mi va păstra confidențialitatea datelor, pentru a putea reveni în viitor și să cresc numărul de comenzi.

US4. Modificabilitate: Ca și dezvoltator, vreau ca aplicația să utilizeze tehnologii și practici de dezvoltare moderne, astfel încât să pot implementa eficient noi funcționalități pentru a adapta aplicația la posibilele schimbări din industrie.

US4.1. Adoptarea principiilor SOLID: Ca și dezvoltator, vreau ca aplicația să adoptare principiile SOLID, astfel încât să pot implementa eficient noi funcționalități pentru a adapta aplicația la posibilele schimbări din industrie.

# Perspective arhitecturale

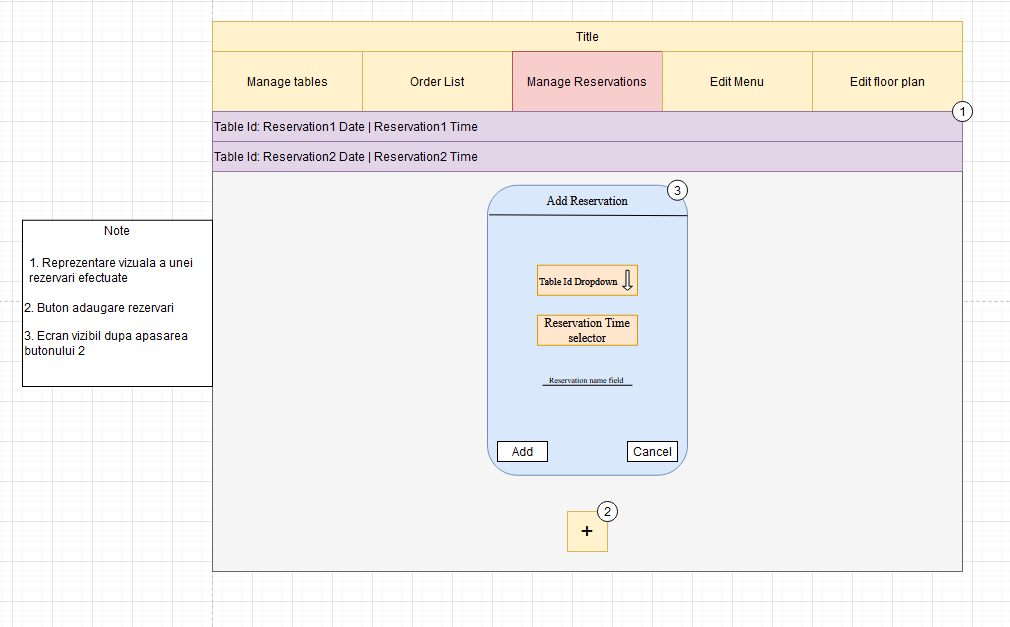
Se va descrie parțial arhitectura aplicației prin intermediul a minim 3 perspective arhitecturale care acoperă semnificativ tacticile arhitecturale alese pentru implementarea scenariilor de calitate descrise pentru cerințele din lista redusă. Perspectivele pot fi alese din lista de mai jos, dar pot fi completate și cu alte perspective considerate relevante pentru aplicație.

# Perspectiva de utilizare

Se va descrie perspectiva arhitecturală asupra aplicației care se referă la interacțiunea utilizatorului cu aplicația, evidențiindu-se aspectele relevante pentru acoperirea specificațiilor de analiză sau proiectare selectate anterior. Se pot folosi diagrame de stări UML sau wireframes de interfață.

Interfața utilizatorului trebuie să fie intuitivă și ușor de navigat, oferind acces rapid la funcționalitățile esențiale. Astfel, perspectiva de utilizare acoperă tactica [T1], prin structurarea aplicației în pagini și formulare separate pentru fiecare funcționalitate. Analog, perspectiva de utilizare acoperă și tactica [T2], ce ghidează utilizatorul printr-un anumit flux printr-o interfață de tip “step wizard”.

Diagrama wireframe de mai jos prezintă interfața principală prin care personalul unui restaurant gestionează rezervările:



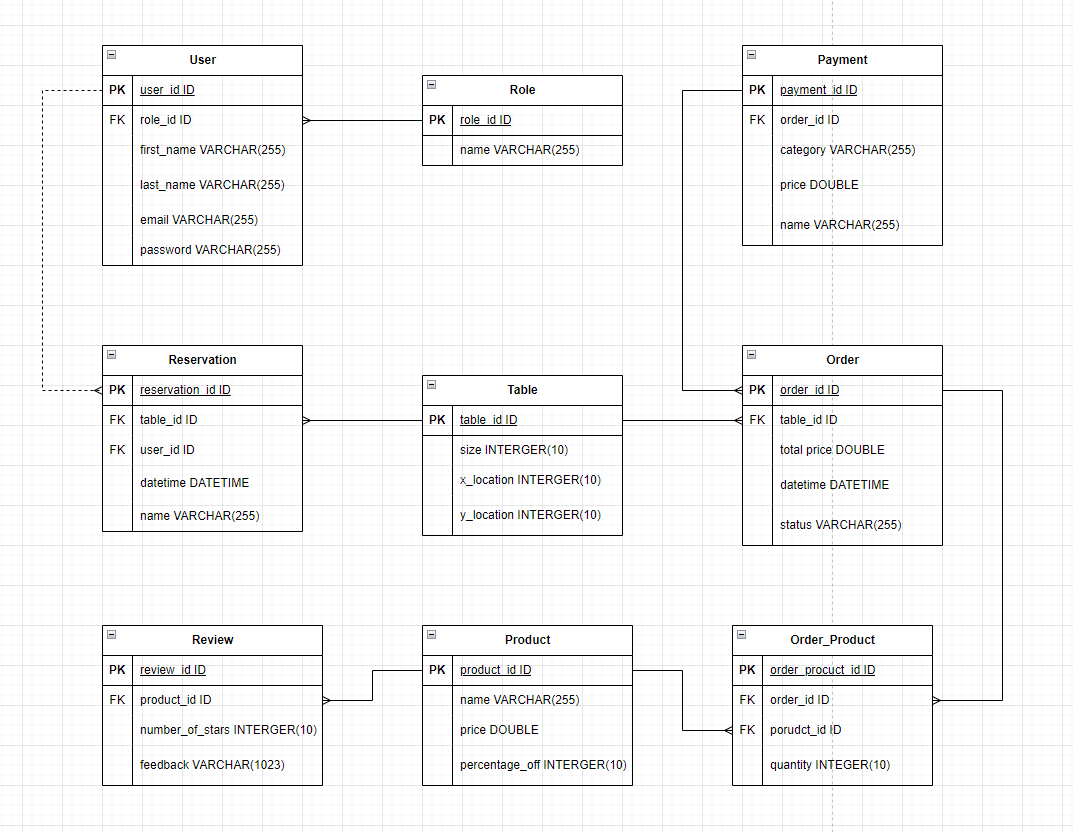
# Perspectiva asupra datelor

Se va descrie perspectiva arhitecturală asupra aplicației care se referă la modelul conceptual al datelor utilizate în aplicație, evidențiindu-se aspectele relevante pentru acoperirea specificațiilor de analiză sau proiectare selectate anterior. Se pot folosi diagrame de date UML sau alte diagrame de prezentare adecvate modelului de date folosit.

Tactica [T5.] este acoperită de perspectiva datelor, deoarece implementarea tacticii a plecat de la nevoia menținerii confidențialității datelor.

Tactica [T6.] este acoperită de perspectiva datelor, deoarece nu putem asigura modificabilitatea unui sistem fără a implementa un model de date ce permite potențiale extinderi.

Diagrama ERD de mai jos prezintă entitățile aplicației:



# Perspectiva structurală

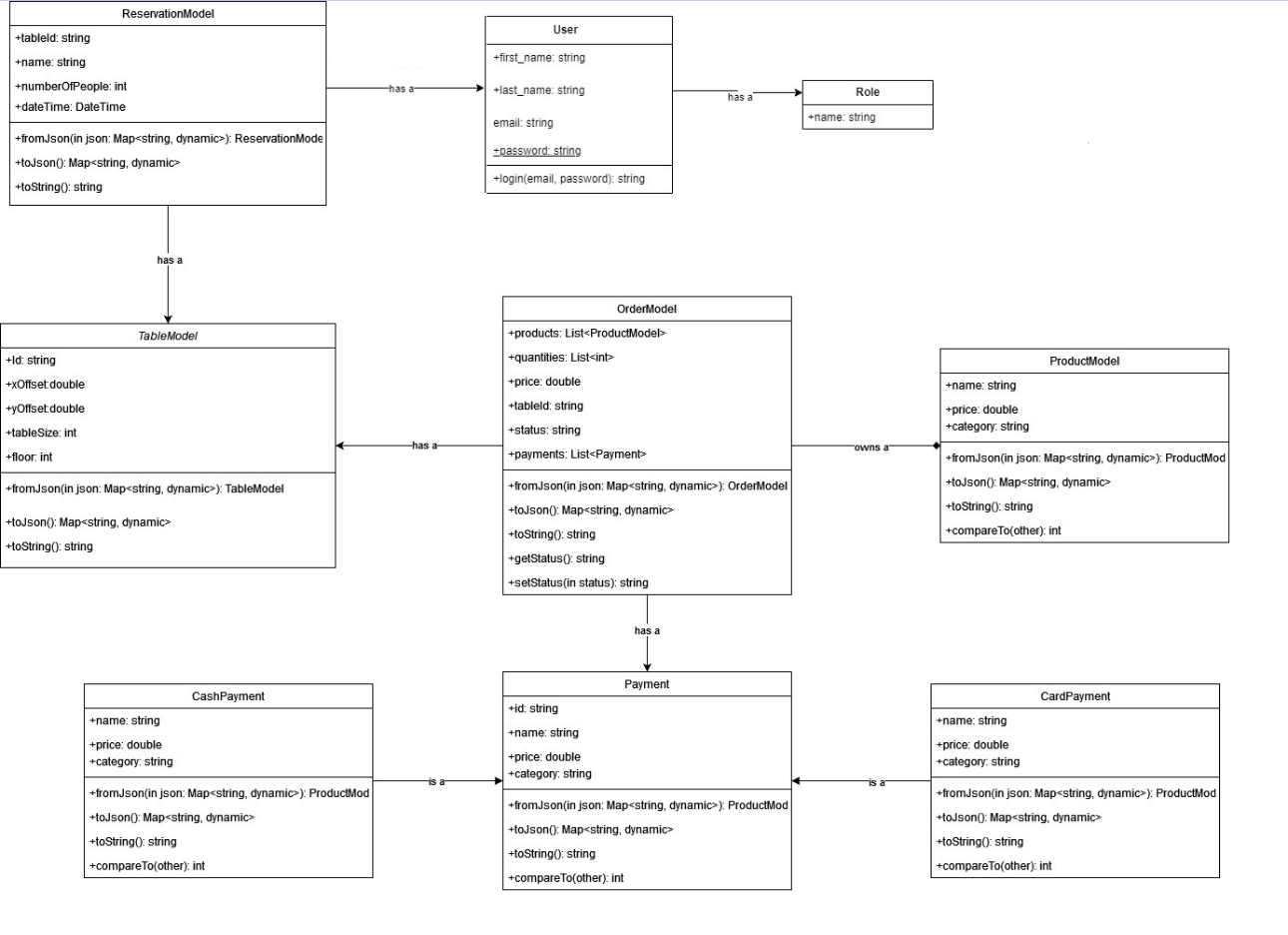
Se va descrie perspectiva arhitecturală asupra aplicației care se referă la componentele ei principale și modul de conectare a acestora, evidențiindu-se aspectele relevante pentru acoperirea specificațiilor de analiză sau proiectare selectate anterior. Se pot folosi diagrame structurale UML sau alte diagrame de prezentare adecvate.

Tactica [T3.] este integrată în perspectiva structurală, deoarece modul în care implementăm clasele poate influența mecanismul de gestionare a erorilor.

Tactica [T5.] este inclusă în perspectiva structurală, deoarece o structurare corespunzătoare a claselor care gestionează informațiile utilizatorilor este crucială pentru o comunicare eficientă cu un furnizor securizat de plăți terț.

Tactica [T6.] este integrată în perspectiva structurală, deoarece organizarea corectă a claselor este fundamentală pentru respectarea principiilor SOLID.

Următoare diagramă de clase prezintă relațiile dintre clase:



# Perspectiva comportamentală

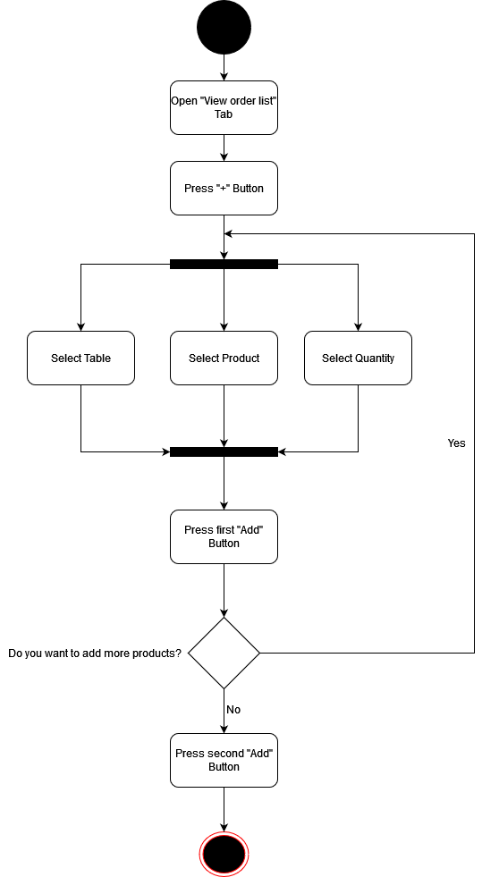
Se va descrie perspectiva arhitecturală asupra aplicației care se referă la modul intern de funcționare a acesteia, evidențiindu-se aspectele relevante pentru acoperirea specificațiilor de analiză sau proiectare selectate anterior. Se pot folosi diagrame comportamentale UML, cum ar fi diagrame de stări, de activități sau de tranziții.

Tacticile [T1.] și [T2.] sunt abordate din perspectiva comportamentală, deoarece fluxul aplicației este strâns legat de interfața UI.

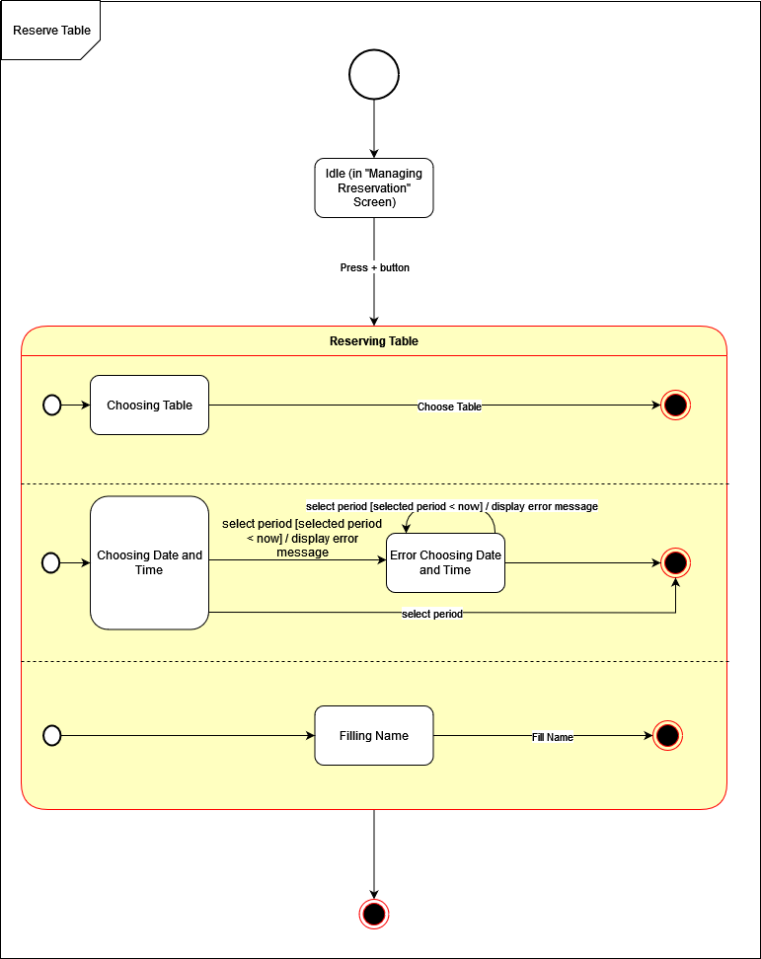
Tactica [T3.] este acoperită de perspectiva comportamentală, deoarece înțelegerea fluxurilor aplicației este esențială pentru implementarea mecanismelor de gestionare a erorilor.

Tactica [T5.] este acoperită de perspectiva comportamentală, deoarece trebuie integrată în fluxul de achiziționare al comenzilor.

Diagrama de activitate de mai jos prezintă fluxul de adăugare al unei comenzi:



Iar următoarea diagramă de stări prezintă fluxul de adăugare al unei rezervări:



# Alegerea tehnologiilor

Se vor propune tehnologia sau tehnologiile în care să fie dezvoltată aplicația, care să corespundă scopului aplicației și a cărei arhitectură de referință să acopere semnificativ specificațiile arhitecturale de analiză și proiectare prezentate mai sus.

Tehnologia pentru Back End: Spring

Pentru back end-ul aplicației de management al restaurantelor, propun utilizarea Spring Framework datorită capacității sale de a aborda eficient tacticile arhitecturale principale. Spring oferă o modularitate excelentă și permite implementarea tacticilor avansate de capturare și gestionare a erorilor [T3.]. Prin utilizarea aplicațiilor de monitorizare și analiză în timp real, precum Grafana, Spring asigură o disponibilitate optimă și gestionarea eficientă a incidentelor.

Spring Security joacă un rol vital în menținerea confidențialității datelor utilizatorilor [T5.]. Utilizând furnizori terți de securitate, cum ar fi 3D Secure, Spring oferă o protecție robustă pentru tranzacțiile utilizatorilor. În plus, adoptarea principiilor SOLID și structura clară a codului în Spring facilitează modificabilitatea [T6.], permițând extinderea ușoară a aplicației cu noi funcționalități.

Din perspectiva persistenței datelor, am optat pentru utilizarea unei baze de date relaționale, MySQL, datorită beneficiilor sale semnificative în contextul aplicației noastre. Bazele de date relaționale oferă performanțe superioare și scalabilitate, aspecte importante pentru cerințele noastre. Alegerea unei soluții SQL ne permite, de asemenea, să garantăm integritatea și consistența datelor, asigurând astfel o fundamentare solidă pentru arhitectura sistemului.

Tehnologia pentru Front End: Flutter

Pentru front end-ul aplicației, recomand utilizarea Flutter datorită capacității sale de a crea interfețe de utilizator intuitive și eficiente. Flutter facilitează organizarea aplicației în pagini și formulare dedicate fiecărei funcționalități, abordând astfel tacticile avansate de UX pentru utilizabilitate [T1.]. De asemenea, prin adoptarea modelului de tip „step wizard”, Flutter poate ghida utilizatorul pe tot parcursul procesului, îmbunătățind astfel utilizabilitatea [T2].

Prin performanța ridicată și suportul cross-platform, Flutter asigură notificarea administratorului în cazul apariției unor probleme [T4.]. Acest aspect contribuie la menținerea unei disponibilități optime a aplicației. Funcționalitatea de hot reload și dezvoltarea rapidă oferite de Flutter susțin tacticile de testare și menținere, permițând îmbunătățirea constantă a interfeței utilizatorului.

Concluzii:

Utilizarea Spring pentru back end și Flutter pentru front end reprezintă o alegere optimă pentru dezvoltarea aplicației “Restaurant Management Application”. Spring asigură o arhitectură modulară, securizată și ușor de modificat, în timp ce Flutter garantează o interfață de utilizator performantă și accesibilă pe multiple platforme. Această combinație de tehnologii acoperă tacticile arhitecturale principale enunțate anterior, oferind o soluție robustă și flexibilă care răspunde cerințelor complexe ale unui restaurant modern.