

Kolegji UBT - Higher Education Institution

Shkenca Kompjuterike dhe Inxhinieri

PRISHTINE

**Punim Seminarik**

Lenda: Dizajni i Sistemit te Softuerit  
Tema: City Services Web Application

Mentori: Studentet:

Greta Ahma Rinor Agaj

Azem Kasumi

Agon Gashi

**Prishtine, Janar 2025**

1. Hyrja

Projekti "City Services" ka për qëllim krijimin e një aplikacioni web të shkallëzueshëm, të mirëmbajtshëm dhe të lehtë për përdoruesit, i cili ofron shërbime si blerje produktesh, marrje me qira të automjeteve dhe rezervim tavolinash në restorante. Ky dokument seminarik integron fazat kryesore të zhvillimit të softuerit, duke përfshirë modelimin e sistemit, projektimin e arkitekturës së softuerit (SAD), dhe projektimin e detajuar (DDD), së bashku me kodin përfundimtar të softuerit.

2. Përmbledhje e Sistemit

Sistemi "City Services" menaxhon në mënyrë efikase të dhënat dhe proceset kryesore për shërbime të ndryshme. Ai përfshin tabela për klientët, njoftimet, porositë, produktet, kategoritë, shportën dhe artikujt e shportës, duke siguruar lidhje logjike midis entiteteve për një menaxhim të organizuar të të dhënave.

3. Modeli Konceptual  
  
Projekti "City Services" është krijuar për të menaxhuar në mënyrë efikase të dhënat dhe proceset kryesore të një sistemi administrativ. Ky model konceptual përfshin tabelat Client, Notification, Order, Product, Category, Cart, dhe CartItem.

Tabela Client ruan të dhënat kryesore të klientëve, duke përfshirë emrin dhe email-in e tyre, dhe lidhet me tabelat Notification dhe Order për të menaxhuar njoftimet dhe porositë e lidhura me ta.

Tabela Product menaxhon informacionin rreth produkteve, si emrin, kategorinë dhe çmimin e tyre. Kjo tabelë lidhet me tabelën Category, e cila përmban detaje për kategoritë përkatëse.

Tabela Cart shërben për të menaxhuar shportën e blerjeve të klientëve, duke pasur lidhje direkte me tabelën CartItem, që përfshin produktet dhe sasinë e tyre brenda një shporte të caktuar.

Sistemi është projektuar për të siguruar lidhje logjike midis entiteteve kryesore dhe për të mbështetur menaxhimin e të dhënave në një mënyrë të organizuar dhe efikase, duke përfshirë funksionalitete për ruajtjen, përditësimin dhe vizualizimin e të dhënave nga përdoruesit.

A diagram of a company

Description automatically generated

4. Arkitektura e Softuerit (SAD)

Përshkrimi i ProduktitEmri: City Services

Visioni i Produktit

"City Services" është një web aplikacion multifunksional që ofron shërbime të ndryshme për përdoruesit, duke përfshirë:

* Blerjen e Produkteve: Përdoruesit mund të shfletojnë dhe të blejnë produkte të ndryshme me mundësi zgjedhjeje për metoda të transportit.
* Rent a Car**:** Përdoruesit mund të shohin automjetet e disponueshme për rent dhe të rezervojnë ato me ndihmën e një kalendari.
* Rezervimi i Tavolinave në Restorante: Rezervime të lehta në restorantet lokale me opsione për vlerësime.

Adminët kanë akses në funksionalitete shtesë si:

* Menaxhimi i produkteve (shtim, heqje, përditësim, ndryshim cmimi).
* Shtimi dhe menaxhimi i restoranteve.
* Shtimi dhe menaxhimi I automjeteve për rent.
* Shikimi i statistikave dhe të dhënave mbi përdoruesit.

PALËT E INTERESIT

Palët e interesuara përfaqësojnë një nga aspektet kyçe të menaxhimit të projekteve. Ato janë individë dhe organizata që ndikohen pozitivisht ose negativisht ne ekzekutimin dhe përmbushjen e projektit. Në projektin tonë, që është zhvillimi i një aplikacioni gjithçka-në-një për tregti elektronike,makina me qera, dhe rezervime per restaurante te ndryshme, palët e interesuara janë të ndarë në dy kategori kryesore: të brendshme dhe të jashtme. Për të siguruar që projekti të jetë i suksesshëm, është e rëndësishme të identifikohen dhe të menaxhohen këto palë të interesuara.

Në kontekstin e aplikacionit tonë, palët e interesuara do të luajnë role të ndryshme për të ndihmuar në përmbushjen e objektivave teknike dhe financiare.

Palët e Brendshme të Interesit

1. Ekipi i Projektit
   * Përgjegjës për zhvillimin teknik të aplikacionit dhe sigurimin e cilësisë.
   * Përfshin programues, testues dhe dizajnerë UI/UX që kontribojnë në funksionalitetin dhe pamjen e platformës.
2. Menaxheri i Projektit
   * Ka përgjegjësinë për planifikimin, prokurimin dhe ekzekutimin e projektit.
   * Bashkëpunon me palë të interesuara për të balancuar kërkesat dhe burimet.
3. Sponsorët
   * Sigurojnë burimet financiare dhe monitorojnë ecurinë e projektit.
   * Përfshijnë investitorë dhe aksionarë të cilët presin kthim mbi investimin.
4. Ekipet mbështetëse
   * Të tilla si ekipi i shitjeve dhe ai i marketingut, të cilët do të përfitohen nga produkti final.
   * Ndihmojnë në përcaktimin e strategjive për të targetuar përdoruesit dhe promovuar aplikacionin.

Palët e Jashtme të Interesit

1. Klientët e Jashtëm
   * Janë përdoruesit (users) të aplikacionit, të cilët përdorin platformën për të blerë, rezervuar apo për të marrë makina me qira.
   * Përfshijnë biznese lokale dhe individë që kërkojnë shërbime të shpejta dhe të besueshme.
2. Furnizuesit
   * Ofrojnë produktet dhe shërbimet që do të integrohen në aplikacion.
   * Për shembull, restorante dhe kompani të makinave me qira.
3. Nënkontraktorët
   * Për shembull, kompani që ofrojnë shërbime logjistike ose mbështetje teknologjike.
   * Sigurojnë integrimin pa probleme të funksionaliteteve të jashtme si pagesat ose dërgesat.
4. Komuniteti
   * Mund të përfitojne ose ndikohen nga ndërkëmbimi i produkteve dhe shërbimeve.
   * Përfshin edhe partnerët strategjikë si organizatat lokale që promovojnë shërbimet tona.

Plani i Menaxhimit të Stakeholderëve

Për të menaxhuar më mirë komunikimin dhe përfshirjen e palëve të interesuara, kemi krijuar një plan të detajuar:

* Frekuenca e Komunikimit:
  + Klientët: Një herë në muaj përmes sondazheve online.
  + Sponsorët: Një herë në javë përmes takimeve online.
  + Ekipet e brendshme: Përditësime ditore përmes platformave si Slack dhe Microsoft Teams.
* Mjetet e Komunikimit:
  + Takime virtuale për diskutime strategjike.
  + Email dhe dokumente zyrtare për ndarjen e planeve dhe raporteve.
  + Formulare të rregullta feedback për të kuptuar kërkesat dhe shqetësimet e palëve të jashtme.

Përfshirja e Klientëve dhe Komunitetit

Klientët dhe komuniteti luajnë një rol të rëndësishëm në përmirësimin e aplikacionit. Ne kemi planifikuar:

* Sondazhe Periodike: Për të mbledhur mendime mbi funksionalitetet ekzistuese dhe nevojat e reja.
* Evente Lokale: Partneritet me organizata lokale për të promovuar dhe testuar aplikacionin.
* Kanale të Hapura të Komunikimit: Mbështetje 24/7 për të adresuar problemet dhe për të marrë sugjerime.

Case Studies: Shembuj të Ndërveprimeve të Stakeholderëve

Për të ilustruar ndërveprimet midis stakeholderëve dhe projektit tonë:

* Shembulli 1: Një klient teston një funksionalitet të ri në aplikacion dhe dërgon feedback për përmirësime. Ky feedback përpunohet nga ekipi i zhvillimit dhe aplikohet në planifikimin e ardhshëm.
* Shembulli 2: Një sponsor kërkon një raport financiar të detajuar mbi shpenzimet e projektit. Ekipi menaxhues dërgon një raport mujore të strukturuar, duke siguruar transparencë dhe duke përforcuar besimin e sponsorit.
* Shembulli 3: Një furnizues përballet me një vonesë në dorëzimin e produkteve. Plani B aktivizohet, dhe furnizuesi alternativ siguron produktet pa vonesa të konsiderueshme.

Tabela e Stakeholdereve

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Nr** | **Emri dhe Mbiemri i Stakeholderit** | **Kontaktet** | **Roli në Projekt** | **Mënyra e Komunikimit** | **Teknologjia e Komunikimit** | **Komentet Sqaruese** |
| 1 | Rinor Rexhaj | Adresë Email | Sponsor | Takime Online | Zoom | Kërkon të jetë i informuar për të gjitha zhvillimet. |
| 2 | Orges Sadriu | Numër Telefoni | Menaxher Projekti | Email dhe Takime | Microsoft Teams | Mbikëqyr zhvillimin dhe zbatimin e projektit. |
| 3 | Agon Gashi | Adresë Email | Klient | Feedback Online | Formularë Feedback | Kërkon lehtësi për përdorimin e platformës. |
| 4 | Azem Kasumi | Numër Telefoni | Konsulent | Takime Fizike | Zoom | Jep ide për optimizimin e funksionaliteteve. |
| 5 | Rinor Agaj | Adresë Email | Nënkontraktor | Email | Microsoft Teams | Siguron shërbimet logjistike për projektin. |
| 6 | Dren Musliu | Numër Telefoni | Komunitet | Takime Online | Google Meet | Përfaqëson interesat e përdoruesve (users). |
| 8 | Zana Gashi | Adresë Email | Menaxhere Marketingu | Takime Javore | Slack | Bashkërendon strategjinë e marketingut për projektin. |

Tregu i Synuar

Aplikacioni synon qytetarët që kërkojnë lehtësi në blerjen e produkteve, rezervimin e shërbimeve dhe transport.

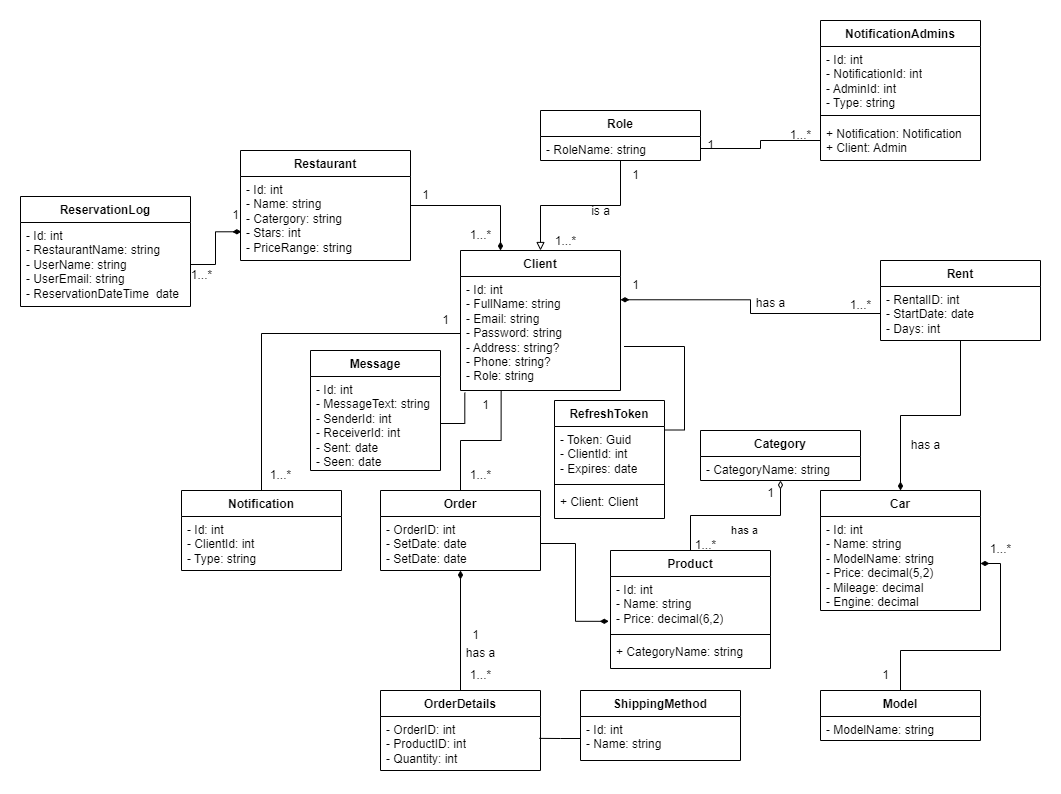
Modelet Arkitekturore

Modelet Statike

Diagramat që përshkruajn strukturën e sistemit ofrojnë një pamje të detajuar të entiteteve dhe marrëdhënieve të tyre. Këto modele janë thelbësore për të kuptuar strukturën e përgjithshme dhe komponentët kryesorë të sistemit.

Diagrami i Klasave:

Ky diagram paraqet entitetet kryesore si Përdorues, Produkt, Rezervim, Car, dhe Restaurant. Ai ilustron atributet dhe marrëdhëniet midis këtyre entiteteve, si dhe ndërlidhjet logjike që ekzistojnë midis tyre.



Detaje Kryesore:

Klasa Client: Përfaqëson përdoruesit e sistemit, përfshirë klientët dhe administratorët. Atributet mund të përfshijnë ID, Emri, Email, dhe Roli.

Klasa Produkt: Përmban informacion rreth produkteve që janë të disponueshme për blerje, si Emri i Produktit dhe Cmimi.

Klasa Car: Përfaqëson automjetet që janë të disponueshme për qira, me atribute si Cmimi, Engine, dhe Kilometrat e kaluara te vetures.

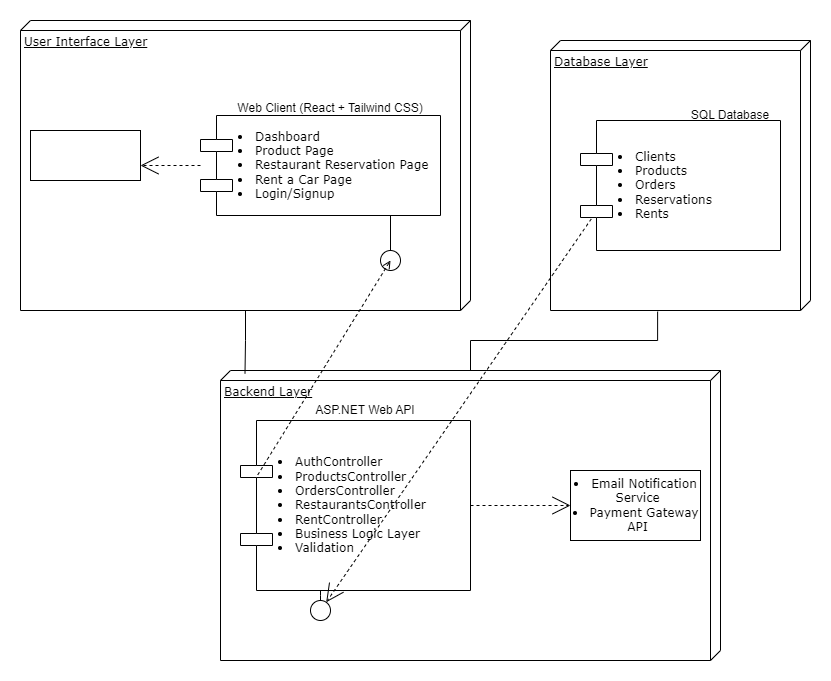
Klasa Restaurant: Menaxhon të dhënat e restoranteve të disponueshme për rezervime, përfshirë Emrin, Kategorine, dhe Yjet (Star Review).

Marrëdhëniet:

Ndërmjet Përdorues dhe Rezervim ekziston një marrëdhënie një-për-shumë, pasi një përdorues mund të bëjë shumë rezervime.

Klasa Restaurant dhe Rezervim ndërlidhen për të menaxhuar disponueshmërinë e tavolinave.   
  
  
Diagrami i Komponenteve:

Diagrami i komponentëve vizualizon arkitekturën e sistemit duke përfaqësuar komponentët kryesorë dhe ndërveprimet e tyre. Ai tregon marrëdhëniet mes shtresave të frontend-it, backend-it dhe databazes, duke treguar rrjedhën e të dhënave gjatë proceseve kryesore si shfletimi i produkteve, rezervimet në restorante dhe qiraja e makinave.



Komponentët Kryesorë:

Frontend (React UI): Përgjegjës për ndërveprimet e përdoruesit dhe shfaqjen e të dhënave. Lidhet me backend-in përmes thirrjeve API.

Backend (ASP.NET Core): Menaxhon logjikën e biznesit, trajton kërkesat dhe komunikon me bazën e të dhënave.

Databaza (SQL Server): Ruajtja e të dhënave për përdoruesit, produktet, rezervimet dhe konfigurimet e sistemit.

Rrjedha e Ndërveprimeve:

Përdoruesit nisin veprime përmes frontend-it (p.sh., rezervimi i një tavoline).

Backend-i përpunon kërkesat dhe merr të dhëna nga databaza.

Databaza kthen informacionin e kërkuar, që backend-i e perpunon dhe ia dërgon frontend-it.

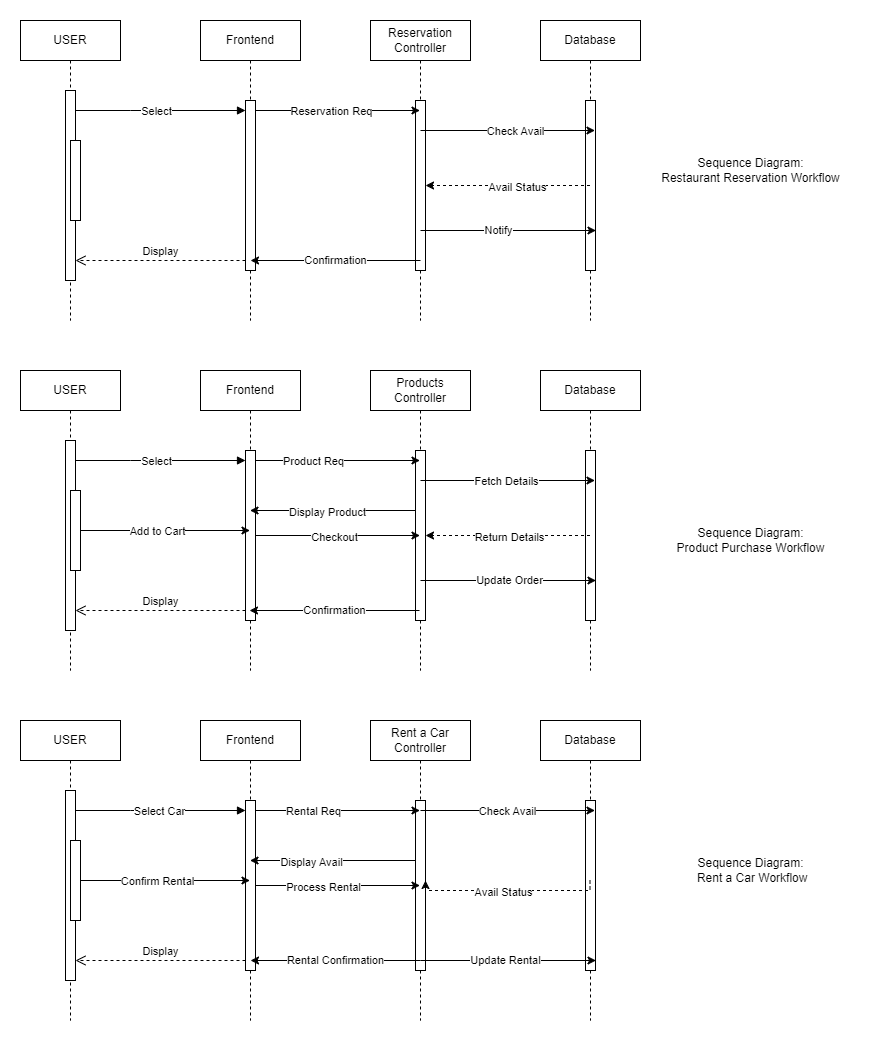
Frontend-i rifreskon ndërfaqen bazuar në përgjigjet e backend-it.

Modelet Dinamike

Modelet dinamike tregojnë se si funksionet dhe proceset rrjedhin në sistem gjatë ekzekutimit. Këto modele janë jetike për të kuptuar ndërveprimet ndërmjet komponentëve dhe renditjen e hapave të procesit.

Diagrami i Sekuencës:

Ky diagram paraqet ndërveprimet e sistemit gjatë procesit të rezervimit të një tavoline në restorant, blerjes se nje produkti apo rezervimit te nje makine. Ai përfshin aktorët dhe komponentët kryesorë që bashkëveprojnë për të përfunduar këtë proces.



Rrjedha e Procesit (Shembull: Restaurant):

Inicimi: Përdoruesi dërgon një kërkesë për rezervim përmes ndërfaqes së përdoruesit.

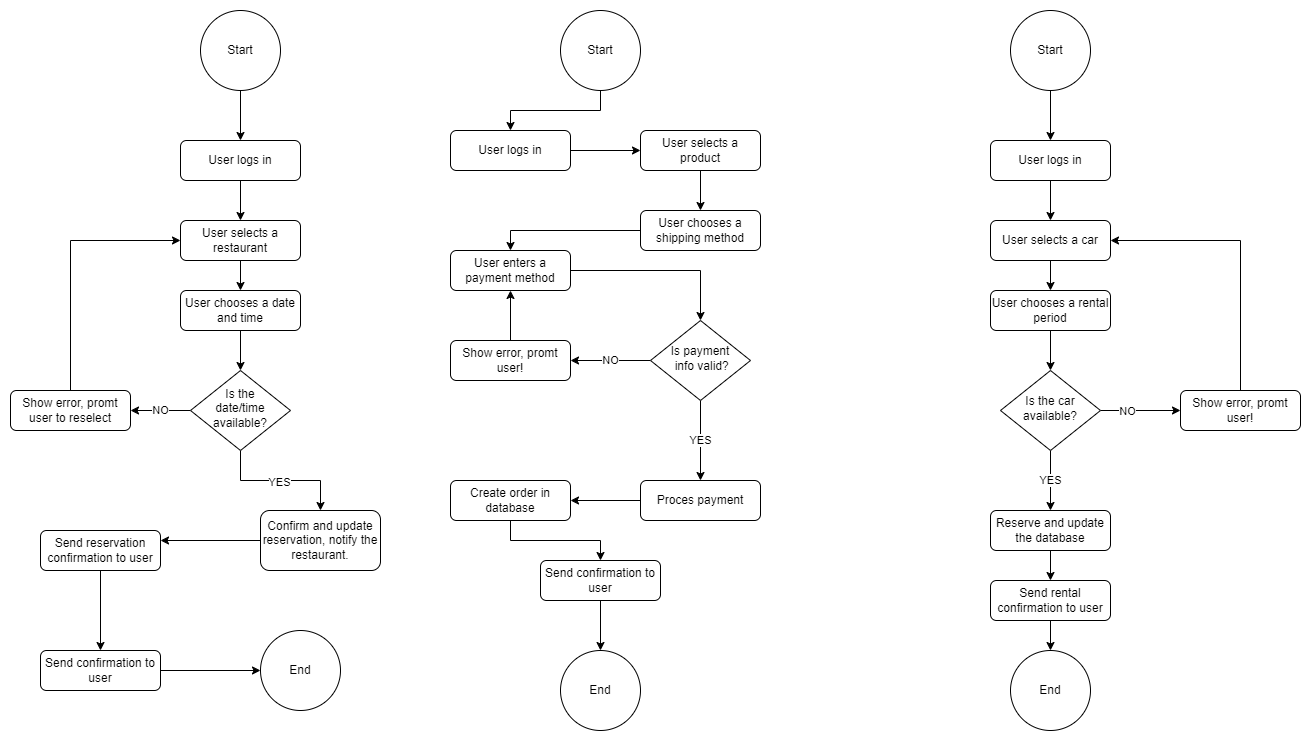
Validimi: Backend-i verifikon të dhënat e rezervimit, përfshirë datën dhe disponueshmërinë e restorantit.

Konfirmimi: Nëse të dhënat janë të sakta, sistemi krijon rezervimin dhe njofton përdoruesin përmes një mesazhi konfirmimi.

Perditesimi: Baza e të dhënave perditesohet për të reflektuar rezervimin e ri.

Diagrami i Aktivitetit:

Ky diagram përshkruan rrjedhën e aktiviteteve për blerjen e një produkti dhe zgjedhjen e metodës së transportit, rezervimin e nje tavoline ne Restaurantet lokale, si dhe rezervimin e nje makine per Rent.



Rrjedha e Procesit (Shembull: blerja e nje produktit):

Shfletimi: Përdoruesi shfleton produktet e disponueshme në sistem.

Zgjedhja: Përdoruesi zgjedh një produkt dhe e shton në shportë.

Përzgjedhja e Metodës së Transportit: Përdoruesi zgjedh opsionin më të përshtatshëm për transport.

Pagesa: Përdoruesi përfundon blerjen përmes një metode të sigurt pagese.

Konfirmimi: Sistemi dërgon një mesazh përfundimtar për konfirmimin e porosisë dhe metodës së transportit të zgjedhur.

Mapping Between Modeles

Tabela për të lidhur komponentët statikë dhe dinamikë të sistemit:

|  |  |  |
| --- | --- | --- |
| **Komponenti Statik** | **Procesi Dinamik** | **Shpjegim** |
| **Klasa Përdorues (Client)** | Sekuenca për hyrjen dhe regjistrimin e përdoruesit | Ilustron ndërveprimet e përdoruesit me sistemin për autentifikim dhe krijimin e llogarive të reja. |
| **Klasa Produkt** | Sekuenca për shfletimin dhe blerjen e produkteve | Shpjegon si përdoruesit mund të zgjedhin dhe të blejnë produkte nga lista e disponueshme. |
| **Klasa Rezervim** | Aktiviteti për rezervimin e tavolinave në restorante | Përshkruan hapat për të bërë një rezervim, duke përfshirë konfirmimin dhe perditesimin e bazës së të dhënave. |
| **Klasa Car (Automjeti)** | Sekuenca për marrjen e makinave me qera | Paraqet rrjedhën e proceseve për përzgjedhjen dhe rezervimin e automjeteve për qira. |
| **Klasa Restaurant** | Aktiviteti për menaxhimin e rezervimeve në restorante | Tregon ndërveprimet mes sistemit dhe databazes për menaxhimin e disponueshmërisë së tavolinave. |
| **Klasa Admin** | Sekuenca për menaxhimin e produkteve dhe statistikave | Shpjegon mënyrën se si administratorët mund të shtojnë, përditësojnë ose heqin produkte dhe të monitorojnë statistikat. |
| **Klasa Porosi (Order)** | Aktiviteti për përpunimin e porosive të përdoruesve | Ilustron hapat për verifikimin e pagesës, përpunimin e porosisë dhe dërgimin e konfirmimeve. |
| **Klasa Pagesa** | Sekuenca për përpunimin e pagesave | Demonstron ndërveprimet me portat e pagesave për transaksione të sigurta. |
| **Klasa Njoftime (Notification)** | Aktiviteti për dërgimin e njoftimeve | Përshkruan mënyrën se si përdoruesit informohen për rezervimet, ndryshimet dhe përditësimet. |

Arsyetimi për Dizajnin

Parimet e Aplikuara

1. Modulariteti: Sistemi është ndarë në module të veçanta si "Menaxhimi i Produkteve", "Rent a Car", dhe "Rezervimi i Restoranteve" për mirëmbajtje të lehtë.
2. Shkallëzueshmëria: Arkitektura mbështet shtimin e moduleve të reja pa ndikuar në funksionalitetet ekzistuese.
3. Përdorshmëria: Ndërfaqe intuitive për përdoruesit e rregullt dhe adminët.

Zgjedhje Kritike

* MVC (Model-View-Controller): Arkitektura e përdorur për të ndarë logjikën e biznesit nga ndërfaqja.
* Pattern-at e Aplikuara:
  + Observer Pattern për notifikimet e përdoruesve (p.sh., dërgimi i email-eve kur përfundon një rezervim).
  + Factory Pattern për krijimin e instancave të objekteve si automjete dhe produkte (siguron krijim të standardizuar dhe menaxhim më të mirë të objekteve).
  + Singleton Pattern për menaxhimin e komponentëve të përbashkët (p.sh., DataContext për qasje në bazën e të dhënave).

Utility Tree

|  |  |  |
| --- | --- | --- |
| **Kërkesa** | **Skenari** | **Zgjidhja** |
| Performanca | Përpunimi i rezervimeve brenda 2 sekondave | Optimizimi i pyetjeve në bazën e të dhënave dhe implementimi i strategjive të indeksimit |
| Performanca | Hyrjet e njëkohshme të përdoruesve pa ngadalësim të sistemit | Balancimi i ngarkesës dhe ruajtja e të dhënave të zakonshme në cache |
| Shkallëzueshmëria | Shtimi i shërbimeve të reja si rezervimet për evente | Dizajni modular për integrim të lehtë të komponentëve |
| Shkallëzueshmëria | Trajtimi i trafikut të lartë gjatë orareve të pikut | Shkallëzim horizontal dhe strategji auto-shkallëzimi në ambientet cloud |
| Përdorshmëria | Udhëzime të qarta për përdoruesit e rinj | Tutoriale ndërvepruese dhe sugjerime të integruara në UI |
| Përdorshmëria | Navigim i përshtatshëm për pajisje të ndryshme | Dizajn responsive për pajtueshmëri me pajisje mobile dhe desktop |
| Siguria | Mbrojtja e të dhënave sensitive të përdoruesve | Implementimi i HTTPS, autentifikimi me token të sigurt dhe teknika të enkriptimit të të dhënave |
| Mirëmbajtja | Përditësime dhe korrigjime të lehta të gabimeve | Përdorimi i praktikave më të mira të kodimit dhe implementimi i proceseve CI/CD |

Kërkesat jo-funksionale dhe mënyra se si plotësohen në arkitekturë:

Kërkesat jo-funksionale luajnë një rol thelbësor për të garantuar që sistemi softuerik funksionon në mënyrë të qëndrueshme, efikase dhe të shkallëzueshme. Më poshtë janë detajet e zgjeruara për secilën kategori:

Performance (Performanca): Sistemi duhet të trajtojë kërkesat në një kohë optimale për të siguruar një përvojë të kënaqshme për përdoruesit. Funksionalitetet si rezervimet në restorante dhe blerjet e produkteve kërkojnë përpunim të shpejtë të të dhënave dhe përgjigje të menjëhershme nga sistemi.  
  
  
Zgjidhjet e Zbatuara:

* + Optimizimi i SQL: " Përdorimi i indekseve dhe query-t të avancuara për të përshpejtuar marrjen e të dhënave nga baza e të dhënave."
  + Caching: Mbajtja në memorie e të dhënave që përdoren shpesh për akses më të shpejtë, si produktet më të kërkuara.
  + Asynchronous Processing: Implementimi i proceseve asinkrone për të përpunuar shumë kërkesa njëkohësisht pa ndikuar performancën.
  + Shembull: "Përpunimi i rezervimit të tavolinave brenda 2 sekondave për një përvojë optimale të përdoruesit."

Scalability (Shkallëzueshmëria): Arkitektura duhet të mbështesë zgjerimin pa ndikuar në funksionet ekzistuese, duke lejuar integrimin e shërbimeve të reja si rezervimet për evente apo funksione shtesë për menaxhimin e porosive të personalizuara.  
Zgjidhjet e Zbatuara:

* + Modular Architecture: Ndarja e sistemit në module të pavarura, që mund të përmirësohen dhe shtohen pa ndikuar në funksionalitetet ekzistuese.
  + Extensible APIs: Përdorimi i API-ve fleksibël që mundësojnë zgjerimin e funksioneve dhe integrimin e shërbimeve të reja.
  + Horizontal Scaling: Shtimi i serverëve të rinj për të trajtuar rritjen e ngarkesës pa ndërhyrje në infrastrukturën ekzistuese.
  + Shembull: "Shtimi i moduleve të reja për shërbime të tjera si rezervime eventi pa ndikim negativ në funksionalitetet ekzistuese."

Usability (Përdorueshmëria): Përdoruesit duhet të kenë akses në një ndërfaqe intuitive dhe të thjeshtë për përdorim, me navigim të lehtë dhe udhëzime të qarta për proceset kryesore.

Zgjidhjet e Zbatuara:

* + Responsive Design: Sigurimi i një ndërfaqe që funksionon pa probleme në pajisje të ndryshme, përfshirë kompjuterat desktop dhe telefonat mobil.
  + Interactive Tutorials: Integrimi i udhëzimeve për përdoruesit e rinj për të orientuar dhe lehtësuar navigimin në sistem.
  + Feedback Visualization: Përdorimi i mesazheve vizuale për të njoftuar përdoruesit për suksesin ose dështimin e veprimeve të tyre (p.sh., konfirmimi i rezervimit).
  + Shembull: "Ndërfaqja përfshin një udhëzim për blerjet e para dhe navigim të thjeshtë për çdo kategori produktesh."

Tabela për Kërkesa të Tjera Jo-Funksionale

|  |  |  |
| --- | --- | --- |
| **Kategoria** | **Përshkrimi** | **Shembull Zbatimi** |
| **Security (Siguria)** | Mbrojtja e të dhënave të përdoruesit dhe transaksioneve financiare. | "Autentifikimi përmes token-ave të sigurt dhe enkriptimi i të dhënave." |
| **Maintainability (Mirëmbajtja)** | Lehtësia për të mirëmbajtur dhe përditësuar kodin. | "Kthimi i përgjigjeve me kod të pastër dhe modular që lejon perditesime të shpejta." |
| **Reliability (Besueshmëria)** | Funksionim i qëndrueshëm pa ndërprerje. | "Kopjimi (duplikimi) i databazes për të shmangur humbjet e të dhënave." |

5. Projektimi i Detajuar (DDD)

5.1. Dokumenti i Dizajnit të Detajuar (DDD)

Ky dokument përshkruan në detaje dizajnin për aplikacionin web City Services, duke përfshirë modelet e dizajnit të nivelit të mesëm dhe të ulët dhe lidhjen midis tyre. Objektivi kryesor është të sigurohet që sistemi të jetë i fortë, i shkallëzueshëm dhe i mirëmbajtshëm, duke respektuar parimet e programimit të orientuar drejt objekteve.

5.2. Mid-Level Design Models

Modelet e dizajnit të nivelit të mesëm paraqesin strukturën dhe organizimin funksional të aplikacionit në një nivel modular. Këto diagrame paraqesin përmbajtjen dhe ndërveprimin e moduleve kryesore të sistemit.

A diagram of a company

Description automatically generated

* Përshkrimi: Ky diagram ilustron komponentët kryesorë si Menaxhimi i Përdoruesve, Menaxhimi i Porosive, Rent a Car, dhe Menaxhimi i Restoranteve. Secili modul përfaqëson një pjesë të dallueshme të funksionalitetit të sistemit, duke siguruar modularitet dhe ndarje të qartë të përgjegjësive.

5.3. Low-Level Design Models

Modelet e dizajnit të nivelit të ulët eksplorojnë më tej detajet e implementimit, duke përfshirë klasat individuale, atributet dhe operacionet e tyre, si dhe marrëdhëniet midis tyre. Këto diagrame ndihmojnë në kuptimin e mënyrës se si funksionojnë module të veçanta brenda sistemit.

A white background with black text

Description automatically generated

* Përshkrimi: Ky diagram përqendrohet te klasat individuale dhe lidhjet e tyre, duke përfshirë klasa si User, Product, Order, Car, dhe Reservation. Atributet dhe operacionet e këtyre klasave janë modeluar për të përmbushur kërkesat funksionale të sistemit.

5.4. Mapping Between Models

Lidhja midis modeleve të nivelit të mesëm dhe të ulët siguron përputhshmëri midis strukturës modulare dhe implementimit të detajuar. Tabela më poshtë përfaqëson këtë lidhje:

|  |  |  |
| --- | --- | --- |
| Komponenti i Nivelit të Mesëm | Klasa/Paketa e Nivelit të Ulët | Përshkrimi |
| Menaxhimi i Përdoruesve | User, Role, Authentication | Menaxhon regjistrimin e përdoruesve, rolet dhe funksionalitetin e autentikimit. |
| Menaxhimi i Porosive | Order, OrderDetail, Product | Menaxhon krijimin e porosive, detajet e tyre dhe inventarin e produkteve. |
| Rent a Car | Car, Rent, ReservationLog | Menaxhon disponueshmërinë e makinave, procesin e marrjes me qira dhe rezervimet. |
| Menaxhimi i Restoranteve | Restaurant, Reservation, Review | Lehtëson listimin e restoranteve, rezervimet dhe komentet nga klientët. |

5.5. Detailed Design Rationale

Parimet e Aplikuara

1. Modulariteti: Sistemi është ndarë në module të veçanta si "Menaxhimi i Produkteve", "Rent a Car", dhe "Rezervimi i Restoranteve", për të siguruar mirëmbajtje të lehtë dhe të organizuar.
2. Shkallëzueshmëria:Ne projektin tone arkitektura mbeshtet shtimin e moduleve të reja pa ndikuar në funksionalitetet ekzistuese.
3. Përdorshmëria: Ka ndërfaqe te thjeshte për përdoruesit dhe adminët, e ndërtuar me qëllim të lehtësisë së përdorimit.

Arkitektura dhe Patterns e Përdorura

1. MVC (Model-View-Controller): Ben ndarjen e logjikës së biznesit nga ndërfaqja e përdoruesit për të siguruar modularitet dhe mirëmbajtje të thjeshtë.
2. Observer Pattern: Përdoret për funksionalitete të notifikimeve, si njoftimi i përdoruesve për statusin e rezervimeve ose porosive.
3. Factory Pattern: Eshte implementuar për krijimin e objekteve si Car, Product, dhe User, duke siguruar një mënyrë të standardizuar dhe fleksibile krijimin e tyre.
4. Singleton Pattern: Aplikohet për komponentët e përbashkët si DataContext, që menaxhon qasjen në bazën e të dhënave në mënyrë të sigurt dhe të unifikuar.

Lidhjet midis Klasave

* Agregimi: Përfaqëson marrëdhënie pjesë-e-tërësisë, si Order që përmban disa OrderDetails.
* Trashëgimia: Përdorur për marrëdhënie tipologjike, si Admin dhe Customer që trashëgojnë nga klasa User.

6. Implementimi

Faza përfundimtare përfshiu implementimin e dizajnit në një aplikacion funksional. Kodi, i strukturuar në përputhje me parimet e dizajnit të projektit, u realizua për të ofruar një përvojë të pandërprerë për përdoruesit dhe për të ruajtur shkallëzueshmërinë e sistemit.  
  
*https://github.com/RinorAgaj/DizajniSistemitSoftuerit-Project.git*

7. Integrimi dhe Testimi

Sistemi iu nënshtrua testimeve rigoroze për të verifikuar funksionalitetet e tij dhe për të siguruar përputhshmëri me kërkesat e përdoruesve, duke ruajtur njëkohësisht sigurinë dhe performancën.

8. Përfundim

Zhvillimi i aplikacionit web "City Services" paraqet një qasje gjithëpërfshirëse ndaj zhvillimit modern të softuerit, duke theksuar arkitekturën modulare, dizajnin e orientuar drejt përdoruesit dhe performancën e qëndrueshme të sistemit. Kjo qasje e integruar siguron një zgjidhje të shkallëzueshme të përshtatshme për nevojat në ndryshim të tregut.