

Sistem za dostavu

CS101

Objektno orijentisano programiranje 1

Projektna dokumentacija

05.06.2024

Student: Strahinja Kešelj

Sadržaj

[1. Predlog teme 3](#_Toc138000031)

[2. Opis funkcionalnosti 3](#_Toc138000032)

[3. Struktura aplikacije 3](#_Toc138000033)

[4. Korisničko uputstvo 4](#_Toc138000034)

[5. Zaključak 7](#_Toc138000035)

[6. Literatura 7](#_Toc138000036)

# Predlog teme

Sistem za dostavu hrane.

# Opis funkcionalnosti

Ova aplikacija je sistem za naručivanje koji omogućava korisnicima da kreiraju i pregledaju porudžbine. Postoje tri vrste korisnika: admin, klijent i dostavljač.

Admin ima mogućnost da kreira proizvode, naručuje, pregleda sve porudžbine, dodeljuje dostavljače porudžbinama i pregleda sve dostavljače. Takođe, admin može da registruje nove korisnike.

Klijent može da kreira porudžbine i pregleda sve svoje porudžbine.

Dostavljač može da promeni svoj status (dostupan/nedostupan), prihvati porudžbinu, pregleda sve svoje porudžbine i promeni status porudžbine.

Svaka porudžbina ima status koji može biti: kreirana, dodeljena dostavljaču ili isporučena. Porudžbina se prvo kreira, zatim se dodeljuje dostavljaču, a na kraju se označava kao isporučena.

Svaki korisnik ima korisničko ime, lozinku i ulogu. Klijenti imaju i adresu, a dostavljači imaju status dostupnosti.

# Struktura aplikacije

Opis prema paketima:

Paket **service**:

sadrži klase koje implementiraju logiku poslovnih operacija za različite entitete u sistemu. Svaka klasa u ovom paketu predstavlja servis za određeni entitet i sadrži metode za manipulaciju tim entitetom.

* **FileService** - Ova klasa pruža metode za rad sa datotekama. Metode uključuju čitanje iz datoteke (readFile), pisanje u datoteku (writeFile), kreiranje datoteke (createFile) i brisanje sadržaja datoteke (clearFile).
* **TextFileDatabaseService** - Ova klasa implementira IDatabaseService interfejs i pruža metode za rad sa bazom podataka koja je implementirana kroz tekstualne datoteke. Metode uključuju čuvanje podataka (saveData), dohvatanje podataka (retrieveData) i brisanje svih podataka (clearData).
* **OrderService** - Ova klasa implementira IOrderService interfejs i pruža metode za rad sa porudžbinama. Metode uključuju kreiranje porudžbine (createOrder), ažuriranje porudžbine (updateOrder), dodavanje proizvoda u porudžbinu (addProductToOrder), prikaz svih porudžbina (showAllOrders) i brisanje svih porudžbina (clearOrders).
* **ProductService** - Ova klasa implementira IProductService interfejs i pruža metode za rad sa proizvodima. Metode uključuju prikaz svih proizvoda (showAllProducts), dodavanje proizvoda (addProduct) i brisanje svih proizvoda (clearProducts).
* **UserService** - Ova klasa implementira IUserService interfejs i pruža metode za rad sa korisnicima. Metode uključuju registraciju korisnika (registerUser), prijavu korisnika (login), prikaz svih dostavljača (showAllDeliveryUsers), prikaz svih aktivnih dostavljača (showAllActiveDeliveryUsers), ažuriranje korisnika (updateUser) i brisanje svih korisnika (clearUsers).

Paket **repository** sadrži klase koje implementiraju operacije za pristup podacima za različite entitete u sistemu. Svaka klasa u ovom paketu predstavlja repozitorijum za određeni entitet i sadrži metode za manipulaciju tim entitetom.

**IOrderRepository** - Ovo je interfejs koji definiše metode za rad sa porudžbinama. Metode uključuju čuvanje porudžbine (save), ažuriranje porudžbine (update), dohvatanje porudžbine po ID-u (findById), dohvatanje svih porudžbina (findAll) i brisanje svih porudžbina (deleteAll).

* **IProductRepository** - Ovo je interfejs koji definiše metode za rad sa proizvodima. Metode uključuju čuvanje proizvoda (save), dohvatanje svih proizvoda (findAll) i brisanje svih proizvoda (deleteAll).
* **IUserRepository** - Ovo je interfejs koji definiše metode za rad sa korisnicima. Metode uključuju dohvatanje korisnika po korisničkom imenu i lozinki (findByUsernameAndPassword), čuvanje korisnika (save), dohvatanje svih korisnika (findAll), ažuriranje korisnika (update) i brisanje svih korisnika (clearUsers).
* **TextFileOrderRepository** - Ova klasa implementira IOrderRepository interfejs i pruža metode za rad sa porudžbinama koje su sačuvane u tekstualnoj datoteci. Metode uključuju čuvanje porudžbine (save), ažuriranje porudžbine (update), dohvatanje porudžbine po ID-u (findById), dohvatanje svih porudžbina (findAll) i brisanje svih porudžbina (deleteAll).
* **TextFileProductRepository** - Ova klasa implementira IProductRepository interfejs i pruža metode za rad sa proizvodima koji su sačuvani u tekstualnoj datoteci. Metode uključuju čuvanje proizvoda (save), dohvatanje svih proizvoda (findAll) i brisanje svih proizvoda (deleteAll).
* **TextFileUserRepository** - Ova klasa implementira IUserRepository interfejs i pruža metode za rad sa korisnicima koji su sačuvani u tekstualnoj datoteci. Metode uključuju dohvatanje korisnika po korisničkom imenu i lozinki (findByUsernameAndPassword), čuvanje korisnika (save), dohvatanje svih korisnika (findAll), ažuriranje korisnika (update) i brisanje svih korisnika (clearUsers).

Paket **model:**

sadrži klase koje predstavljaju različite entitete u sistemu. Svaka klasa u ovom paketu predstavlja model za određeni entitet i sadrži metode za manipulaciju tim entitetom.

* **AdminUser** - Ova klasa nasleđuje UserBase klasu i predstavlja admin korisnika. Admin korisnik ima mogućnost da prikaže admin meni.
* **CustomerUser** - Ova klasa nasleđuje UserBase klasu i predstavlja klijentskog korisnika. Klijentski korisnik ima mogućnost da prikaže klijentski meni.
* **DeliveryUser** - Ova klasa nasleđuje UserBase klasu i predstavlja dostavljača. Dostavljač ima mogućnost da prikaže meni za dostavljače, da postavi svoju dostupnost, da doda porudžbinu i da ukloni porudžbinu.
* **Order** - Ova klasa predstavlja porudžbinu. Porudžbina ima mogućnost da doda proizvod, da ukloni proizvod, da izračuna ukupnu cenu i da promeni status.
* **OrderStatus** - Ovo je enumeracija koja predstavlja status porudžbine. Status porudžbine može biti: kreirana, dodeljena dostavljaču ili isporučena.
* **Product** - Ova klasa predstavlja proizvod. Proizvod ima mogućnost da postavi i dohvati svoje atribute kao što su id, ime, cena i kategorija.
* **ProductType** - Ovo je enumeracija koja predstavlja tip proizvoda. Tip proizvoda može biti: piće ili hrana.
* **UserBase** - Ova klasa predstavlja osnovu za sve korisnike. Korisnik ima mogućnost da postavi i dohvati svoje atribute kao što su id, korisničko ime, lozinka, uloga i adresa. Takođe, korisnik ima mogućnost da prikaže meni.
* **UserRole** - Ovo je enumeracija koja predstavlja ulogu korisnika. Uloga korisnika može biti: admin, klijent ili dostavljač.
* **IHandleOrder** - Ovo je interfejs koji definiše metode za rad sa porudžbinama. Metode uključuju dodavanje proizvoda, uklanjanje proizvoda, izračunavanje ukupne cene i promenu statusa.

Paket **io:**

sadrži klase koje se bave unosom i izlazom podataka, kao što su čitanje i pisanje u datoteke, komunikacija preko mreže, interakcija sa korisnikom preko konzole, itd. U vašem projektu, paket io sadrži sledeće klase:

1. **IUserInput** - Ovo je interfejs koji definiše metode za unos podataka od strane korisnika. Metode uključuju unos celog broja (intInput), unosa stringa (stringInput), unosa decimalnog broja (doubleInput) i unosa boolean vrednosti (boolInput).

2. **IUserShow** - Ovo je interfejs koji definiše metode za prikazivanje podataka korisniku. Metode uključuju prikazivanje porudžbina (showOrders), prikazivanje svih dostavljača (listAllDeliveryUsers) i prikazivanje poruke (showMessage).

3. **UserInput** - Ova klasa implementira IUserInput interfejs i pruža metode za unos podataka od strane korisnika. Metode uključuju unos celog broja (intInput), unosa stringa (stringInput), unosa decimalnog broja (doubleInput) i unosa boolean vrednosti (boolInput). Ova klasa koristi Scanner za unos podataka.

4. **UserShow** - Ova klasa implementira IUserShow interfejs i pruža metode za prikazivanje podataka korisniku. Metode uključuju prikazivanje porudžbina (showOrders), prikazivanje svih dostavljača (listAllDeliveryUsers) i prikazivanje poruke (showMessage). Ova klasa koristi System.out.println za prikazivanje podataka.

5. **IFileIO** - Ovo je interfejs koji definiše metode za rad sa datotekama. Metode uključuju čitanje iz datoteke (readFile), pisanje u datoteku (writeFile), kreiranje datoteke (createFile) i brisanje sadržaja datoteke (clearFile).

6. **FileIO** - Ova klasa implementira IFileIO interfejs i pruža metode za rad sa datotekama. Metode uključuju čitanje iz datoteke (readFile), pisanje u datoteku (writeFile), kreiranje datoteke (createFile) i brisanje sadržaja datoteke (clearFile). Ova klasa koristi java.nio.file i java.io pakete za rad sa datotekama.

Paket **handlers**:

1. **IMenuHandler** - Ovo je interfejs koji definiše metode za prikazivanje različitih menija korisnicima. Metode uključuju prikazivanje admin menija (showAdminMenu), prikazivanje klijentskog menija (showClientMenu) i prikazivanje menija za dostavljače (showDeliveryMenu). Svaka od ovih metoda može baciti FileWriteException i FileReadException ukoliko se dogodi greška prilikom čitanja ili pisanja u datoteku.

Paket **exceptions**:

1. **FileReadException** - Ova klasa je izvedena iz klase Exception i predstavlja grešku koja se javlja kada dođe do problema prilikom čitanja iz datoteke. Ova klasa sadrži konstruktor koji prima poruku o grešci i uzrok greške.

2. **FileWriteException** - Ova klasa je izvedena iz klase Exception i predstavlja grešku koja se javlja kada dođe do problema prilikom pisanja u datoteku. Ova klasa sadrži konstruktor koji prima poruku o grešci i uzrok greške.

3. **ServiceException** - Ova klasa je izvedena iz klase RuntimeException i predstavlja grešku koja se javlja kada dođe do problema u servisu. Ova klasa sadrži konstruktor koji prima poruku o grešci i uzrok greške.

4. **UserInputException** - Ova klasa je izvedena iz klase RuntimeException i predstavlja grešku koja se javlja kada dođe do problema sa unosom korisnika. Ova klasa sadrži konstruktor koji prima poruku o grešci i uzrok greške.

Paket **env** sadrži klase koje se bave putanjama do datoteka i direktorijuma.

* **DirectoryPaths** - Ova klasa je enumeracija koja sadrži putanje do različitih direktorijuma u vašem projektu. Svaki element enumeracije predstavlja jedan direktorijum i sadrži metodu getPath koja vraća putanju do tog direktorijuma kao string.
* **FilePaths** - Ova klasa je enumeracija koja sadrži putanje do različitih datoteka u vašem projektu. Svaki element enumeracije predstavlja jednu datoteku i sadrži metodu getPath koja vraća putanju do te datoteke kao string.

Paket **databases** sadrži tekstualne fajlove koji služe kao baza podataka.

Paket **app** sadrži klasu koje se bave logikom aplikacije. U ovom paketu se nalazi klasa **OrderingSystem** koja je centralna tačka aplikacije. Ova klasa upravlja svim operacijama vezanim za naručivanje proizvoda.

Klasa OrderingSystem koristi različite servise (kao što su IOrderService, IProductService, IUserService) za upravljanje podacima o narudžbinama, proizvodima i korisnicima. Takođe, koristi interfejse IUserInput i IUserShow za interakciju sa korisnikom.

Za pokretanje aplikacije:

* **Main –** U ovoj klasi se sadrzi main metoda za pokretanje aplikacije

# Korisničko uputstvo

Korisničko uputstvo za pokretanje i korišćenje aplikacije.

1. Pokrenite aplikaciju pokretanjem `Main` klase.

2. Na početnom ekranu, imate tri opcije:

- Login

- Register

- Exit

3. Ako je ovo vaš prvi put da koristite aplikaciju, trebali biste prvo da se registrujete. Izaberite opciju "Register".

4. Unesite korisničko ime, lozinku i ulogu. Uloge mogu biti "**admin**", "**customer**" ili "**delivery**".

(Napravite prvo admin korisnika)

5. Nakon što se uspešno registrujete, možete se prijaviti. Izaberite opciju "Login" i unesite svoje korisničko ime i lozinku.

6. Nakon uspešne prijave, prikazaće se meni koji zavisi od vaše uloge.

- Ako ste se prijavili kao "admin", imaćete sledeće opcije:

- **Create Product**: Ova opcija omogućava vam da kreirate novi proizvod.

- **Create Order**: Ova opcija omogućava vam da kreirate novu narudžbinu.

- **View All Orders**: Ova opcija prikazuje sve trenutne narudžbine.

- **Assign Delivery User to Orde**r: Ova opcija omogućava vam da dodelite dostavljača određenoj narudžbini.

- **View All Delivery Users:** Ova opcija prikazuje sve korisnike koji su dostavljači.

- **Create a User**: Ova opcija omogućava vam da kreirate novog korisnika.

- **Back to Main Menu**: Ova opcija vas vraća na glavni meni.

- Ako ste se prijavili kao "customer", imaćete sledeće opcije:

- **Create Order**: Ova opcija omogućava vam da kreirate novu narudžbinu.

- **View All Orders**: Ova opcija prikazuje sve vaše narudžbine.

- **Back to Main Menu**: Ova opcija vas vraća na glavni meni.

- Ako ste se prijavili kao "delivery", imaćete sledeće opcije:

- **Change Statu**s: Ova opcija omogućava vam da promenite svoj status (da li ste dostupni za dostavu ili ne).

- **Accept Order**: Ova opcija omogućava vam da prihvatite narudžbinu za dostavu.

- **View My Orders**: Ova opcija prikazuje sve narudžbine koje ste prihvatili za dostavu.

- **Set Order Status**: Ova opcija omogućava vam da promenite status narudžbine (da li je isporučena ili ne).

- **Back to Main Menu**: Ova opcija vas vraća na glavni meni.

7. Da biste se odjavili, izaberite opciju "Back to Main Menu" iz bilo kog menija. Onda možete izabrati opciju "Exit" da biste zatvorili aplikaciju.

Napomena: Ako je baza podataka prazna, prvo morate da kreirate korisnike, proizvode i narudžbine pre nego što možete da koristite ostale funkcionalnosti aplikacije. Ovo se može uraditi kroz meni za "admin" korisnika.

# Zaključak

Ova aplikacija je napravljena koristeći nekoliko principa i koncepta dizajna i programiranja:

* **Repository pattern**: je implementiran kroz interfejs `IRepository` i njegove implementacije `TextFileOrderRepository`, `TextFileUserRepository` i `TextFileProductRepository`. Ove klase pružaju apstrakciju za pristup podacima, omogućavajući ostalim delovima aplikacije da manipulišu podacima bez znanja o tome kako su podaci zapravo sačuvani. Na primer, `TextFileOrderRepository` koristi `TextFileDatabaseService` za čuvanje i dohvatanje podataka iz tekstualnih datoteka. Ovo omogućava da se logika za pristup podacima izoluje od ostatka aplikacije.
* **Composition root**: Ovo je mesto gde se sve zavisnosti kreiraju i sastavljaju, a to je `Main` klasa. U metodi `init()`, kreiraju se sve potrebne zavisnosti kao što su servisi i repozitorijumi, a zatim se prosleđuju kao argumenti prilikom instanciranja `OrderingSystem` klase. Ovo je primer Dependency Injection (DI) dizajn paterna, gde se zavisnosti objekta (u ovom slučaju `OrderingSystem` klase) prosleđuju kroz konstruktor. Ovaj pristup omogućava veću fleksibilnost i testabilnost koda, jer omogućava zamenu stvarnih zavisnosti sa lažnim (mock) objektima tokom testiranja.
* **SOLID principi**: Ovo su principi objektno-orijentisanog dizajna koji pomažu u pisanju čistog i održivog koda. Uključuju Single Responsibility Principle (SRP), Open-Closed Principle (OCP), Liskov Substitution Principle (LSP), Interface Segregation Principle (ISP) i Dependency Inversion Principle (DIP).
* **Princip nasleđivanja**: Ovaj princip je korišćen u modeliranju korisnika gde `DeliveryUser`, `AdminUser` i `CustomerUser` nasleđuju `UserBase` klasu.
* **Princip polimorfizma**: Ovaj princip je korišćen u interakciji sa korisnikom gde se metode interfejsa implementiraju na različite načine u različitim klasama.
* **Princip enkapsulacije**: Ovaj princip je korišćen za skrivanje podataka unutar klasa i pružanje pristupa tim podacima samo kroz metode.
* **Principi čistog koda**: Ovi principi su korišćeni za pisanje koda koji je lako razumljiv, održiv i testabilan. Na primer, metode su kratke i imaju jasne nazive, a kod je organizovan u male, upravljive module.
* **Korišćenje tekstualnih datoteka za skladištenje podataka**: Ova aplikacija koristi tekstualne datoteke za skladištenje podataka o korisnicima, proizvodima i narudžbinama. Za rad sa datotekama koristi se `IFileIO` interfejs i `FileIO` klasa.
* **Korišćenje servisa**: Servisi kao što su `IOrderService`, `IProductService`, `IUserService` se koriste za upravljanje podacima o narudžbinama, proizvodima i korisnicima. Ovi servisi omogućavaju odvajanje logike za upravljanje podacima od ostalih delova aplikacije.
* **Korišćenje interfejsa za interakciju sa korisnikom**: Interfejsi `IUserInput` i `IUserShow` se koriste za interakciju sa korisnikom. Ovo omogućava da se logika za interakciju sa korisnikom odvoji od ostalih delova aplikacije.

# Literatura

Kagi.com

Youtube.com