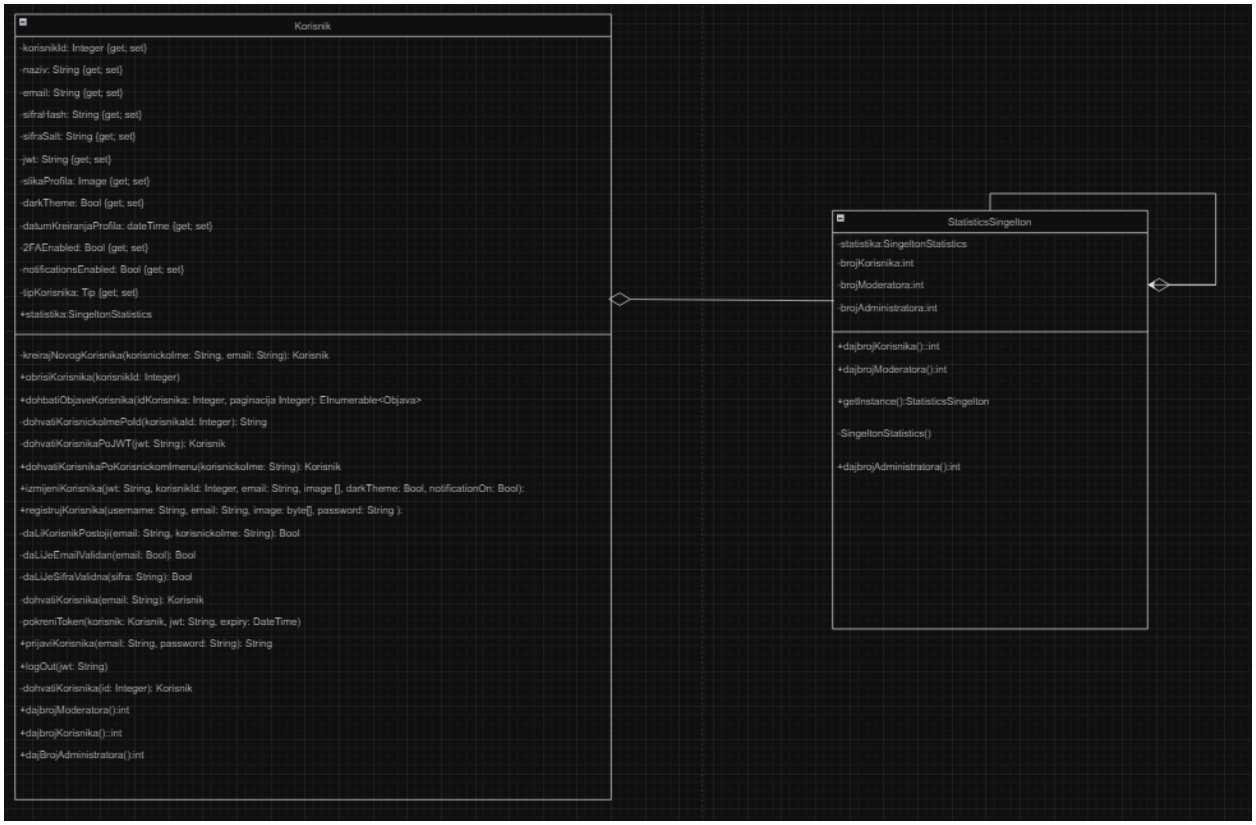
KREACIJSKI PATERNI

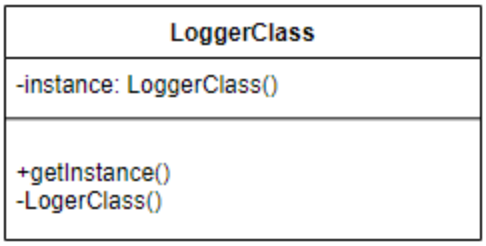
Singleton šablon

Singleton šablon osigurava a da se klasa može instancirati samo jednom i da osigura globalni pristup kreiranoj instanci klase. Postoje objekti čija je samo jedna instanca potrebna i nad kojima je potrebna jedinstvena kontrola pristupa. Instanciranje više nego jednom može prouzrokovati probleme kao što su nekorektno ponašanje programa, neadekvantno korištenje resursa ili nekonzistentan rezultat.

U našem sistemu trebali bismo kreirati klasu SingeltonStatitistics, koja bi trebala vraćati statistiku za korisnički profil.



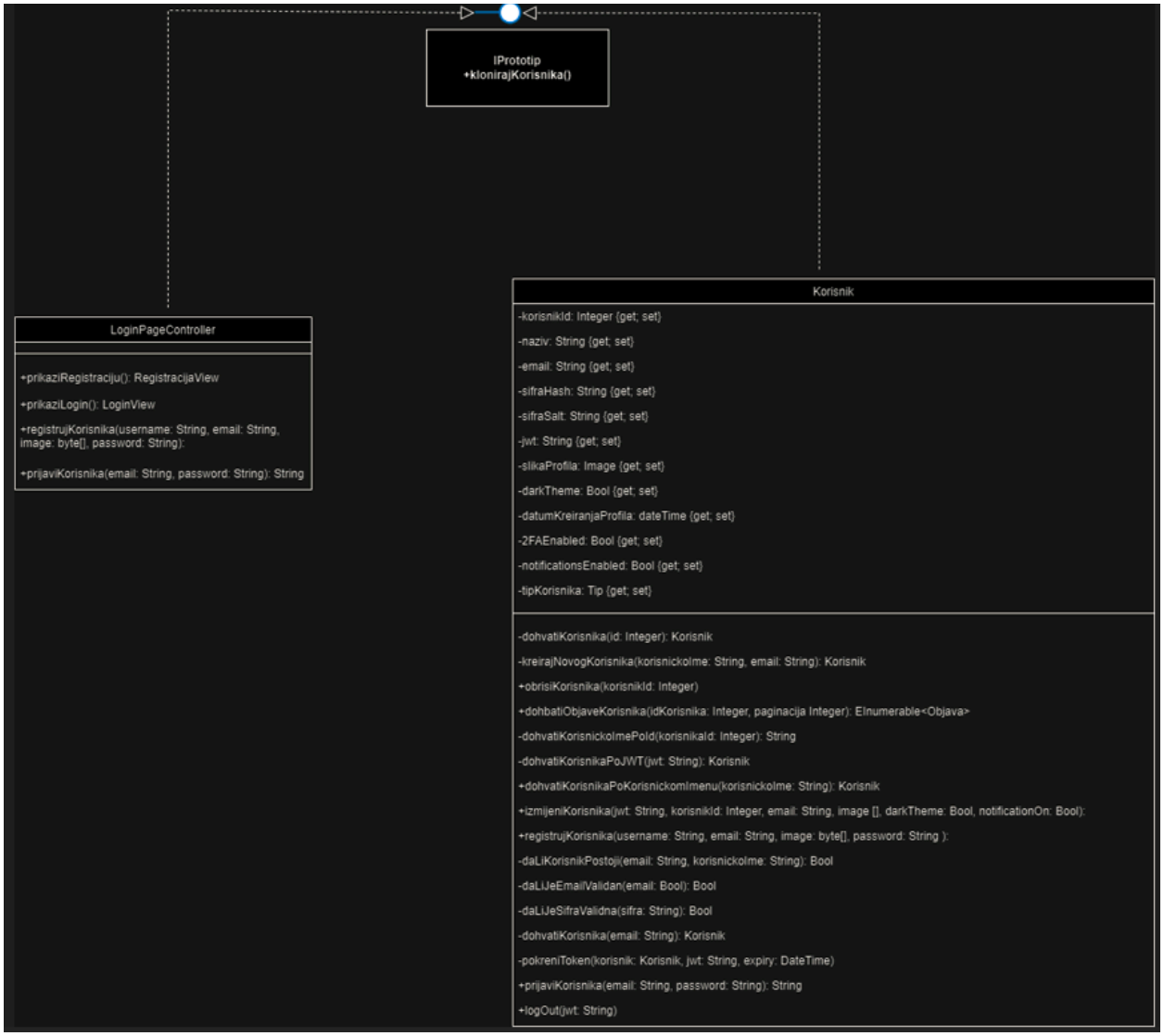
Također, Singleton patern bi mogli primjeniti pri registraciji korisnika na sistem. Mogli bi iskoristiti na način da se bilježe pristupi sistemu korisnika. Napravili bi novu klasu npr. LoggerClass koja bi bila singleton klasa. Potrebno je kreirati jednu instancu da se ne bi otvaralo više log fajlova. Sve klase će imati pristup instanci LoggerClass klase i dobijati željene informacije.



Također, Singleton pattern bi se u našem projektu mogao iskoristiti ako bi se odlučili implementovati neku vrstu globalnog tajmera koji bi se ponašao kao logger. Biljezio bi koliko je korisnik bio aktivan na aplikaciji u toku dana…Pošto se vrijeme računa od registracije korisnika, nema smisla da ovih tajmera bude više, već samo jedan

Prototype patern

Osnovna funkcija ovog paterna je da olakša kreiranje objekata koristeći postojeću instancu, koja se ponaša kao prototip. Novokreiranom objektu možemo promijeniti određene osobine po potrebi. U našem sistemu, ovaj patern bi mogli iskoristiti nad klasom Korisnik. Ukoliko postoje dvije ili više osoba sa istim imenom i prezimenom, umjesto kreiranja novog objekta, možemo klonirati prvu instancu i izmijeniti odgovarajuće podatke koji se razlikuju (email, username, password). Kreirali bi interfejs IPrototip sa metodom kloniraKorisnikaj koja implementira način kloniranja objekta.



Factory Method patern

Uloga Factory Method paterna je da omogući kreiranje objekata na način da podklase odluče koju klasu instancirati. Različite podklase mogu na različite načine implementirati interfejs. Factory Method patern instancira odgovarajuću podklasu(izvedenu klasu) preko posebne metode na osnovu informacije od strane klijenta ili na osnovu tekućeg stanja. U našem sistemu ovaj patern možemo iskoristiti kod registracije korisnika, tj. instanciramo objekte različitih tipova (Korisnik, Zaposlenik, Administrator). To možemo uraditi na način da kreiramo interface IRegistracija, nakon toga klase RegsitracijaKorsinik i RegistracijaZaposlenik, RegistacijaAdministrator koje će implementirati interfejs.

Builder patern

Ovaj patern bismo mogli iskoristiti kod kreiranja objekata tipa Korisnik jer klasa Korisnik sadrži veći broj atributa koji bi se mogli odvojeno definisati, ali idalje ne prevelik broj atributa tako da nije kompleksno ni naše postojeće rješenje. No to bi se moglo promijeniti ukoliko se klasa Korisnik nadogradi sa mnogo više informacija o korisniku i u tom slučaju bi ovaj patern bio itekako od koristi. U slučaju implementacije Builder paterna, postojao bi interfejs IBuilder sa različitim dijelovima kreiranja korisnika, dodajMjestoRodjenja(String drzava, String mjestoRodjenja), dodajDatumRodjenja(int datumRodjenja) itd. Također postojali bi i različiti builderi - BuilderUser bi pozvao samo osnovne metode, dok bi BuilderUseBiznis pozvao i dodatne metode koje bi bile u skladu sa mogućnostima tog korisnika (trenutno takav korsnik nije predviđen, no za potrebe dodavanja ovog paterna u naš sistem, ta funkcionalnost bi se mogla uvesti).

Abstract Factory patern

Abstract Factory patern omogućava da se kreiraju familije povezanih objekata/produkata. Na osnovu apstraktne familije produkata kreiraju se konkretne fabrike (factories) produkata različitih tipova i različitih kombinacija. Patern odvaja definiciju (klase) produkata od klijenta. Zbog toga se familije produkata mogu jednostavno izmjenjivati ili ažurirati bez narušavanja strukture klijenta. Ovaj patern bi bio iskoristiv kada bismo željeli ubaciti filtere za objavu(najpopularnije, filteri po tipu sadržaja, vremenski filteri …). Bilo bi potrebno kreirati interfejs IFactory koji bi na jednostavniji način omogućio dobivanje instance Objava na osnovu datih filtera.