Univerzitet u Sarajevu
Elektrotehnički fakultet u Sarajevu
Odsjek za računarstvo i informatiku

Objektno-orjentirana analiza i dizajn

Projekat: GameHub

Paterni ponašanja

Grupa 4 – Oštri Momci (#momci):
Elvedin Smajić
Vedad Grbo
Tarik Beganović

1.) Chain of Responsibility

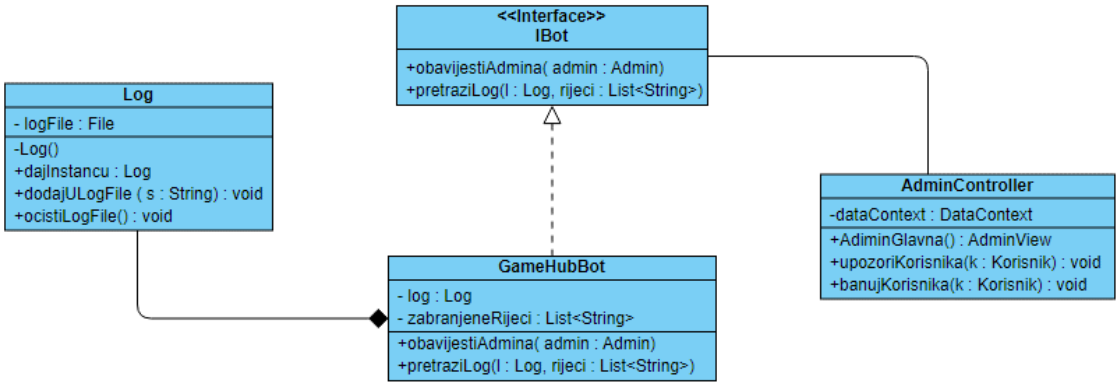
Ovo je patern koji nam omogućava da proslijedimo zahtjeve duž lanca handlera. Nakon što primi zahtjev, svaki handler odlučuje ili da obradi zahtjev ili da ga proslijedi sledećem handleru u lancu. Prilikom registracije dolazimo do provjere na više nivoa kao što su: validacija e-maila, provjera korisničkog imena, provjera lozinke, provjera datuma rođenja, provjera prihvatanja uslova korištenja aplikacije...

2.) Command

Command je patern ponašanja koji pretvara zahtjev u samostalni objekt koji sadrži sve informacije o zahtjevu. Ova transformacija vam omogućava da prosljeđujete zahtjeve kao argumente metode, odlažete ili stavljate u red izvršenja zahtjeva i podržavate operacije koje se ne mogu izvršiti. Ovaj patern možemo iskoristiti za neke komande, koje se nalaze na različitim mjestima (npr. Login dugme na Homepage-u, na formularu za registraciju) a proizvode isti kod.

3.) Mediator

Mediator je patern ponašanja koji vam omogućava da smanjimo zavisnosti između objekata. Mediator ograničava direktnu komunikaciju između objekata i prisiljava ih da sarađuju samo preko posredničkog objekta. Mediator patern možemo iskoristiti na način da napravimo nekog „bota“ koji bi pregledavao sve tekstove objava iz log file-a, tražeći ključne riječi koje bi se sastojale od neprikladnih riječi koje krše pravila korištenja aplikacije. Nakon što bi pronašao takvu riječ, poslao bi notifikaciju adminu koji bi dalje odlučivao šta da uradi sa korisnikom koji je autor teksta(upozorenje ili ban).



4.) Observer

Observer je patern ponašanja koji nam omogućava da definiramo mehanizam pretplate za obavještavanje više objekata o svim događajima koji se događaju objektu koji posmatraju. Ovaj pattern možemo iskoristiti za slanje notifikacija između korisnika. INotifikacija bi bio handler preko kojeg bi recimo admin mogao poslati nekom korisniku određenu vrstu notifikacije (upozorenje, odobravanje igrice itd.)

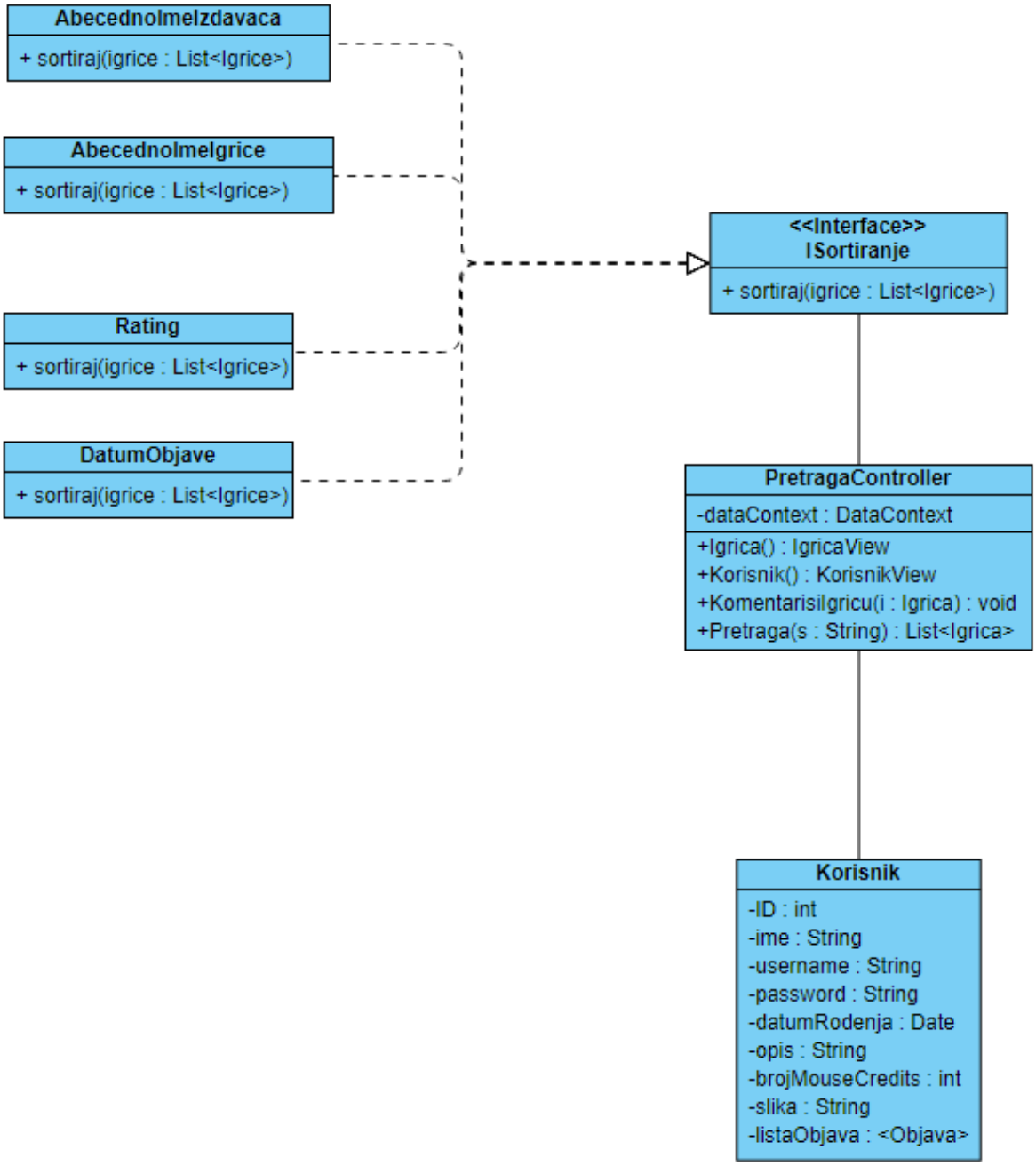
5.) State

State je patern ponašanja koji omogućava objektu da promijeni svoje ponašanje kada se njegovo unutrašnje stanje promijeni te nam se čini kao da je objekt promijenio svoju klasu. State patern možemo iskoristiti u procesu objave igrice od strane developera, koji nakon što pošalje zahtjev, može vidjeti u kojem je stanju taj zahtjev – u stanju obrade od strane admina ili već obrađeno. Ukoliko je u stanju obrade, developer ima mogućnost edit-ovanja zahtjeva.

6.) Strategy

Strategy je patern ponašanja koji nam omogućava da definiramo porodicu algoritama, stavimo svaki od njih u zasebnu klasu i učinimo njihove objekte zamjenjivim. Prilikom pretrage igrica, možemo odabrati način na koji želimo da

budu sortirane, kao recimo po abecednom redoslijedu(naziv igrice), po rejtingu, po datumu objave, po abecednom redoslijedu(naziv izdavača).



7.) Template Method

Template Method je patern ponašanja koji definira kostur algoritma u superklasi, ali dozvoljava podklasama da nadjačaju određene korake algoritma bez promjene

njegove strukture. Ovaj patern možemo implementirati u klasi Komentar koja ima atribute vlasnik te tekstKomentara, a onda smo naslijedili ovu klasu u klasama komentarIgrica i komentarObjava. Komentar igrice pored ovih atributa ima i rating igrice koju komentarišemo, a komentar objave ima broj lajkova. Metoda ispisiKomentar() će imati drugačiji ispis u zavisnoti od klase npr.

KomentarObjava – „Tekst komentara – Vlasnik – Broj lajkova“
KomentarIgrica – „Tekst komentara – Vlasnik – Rating igrice – Igrica“