Sveučilište u Zagrebu
Fakultet organizacije i informatike
Pavlinska 2, 42000 Varaždin

Obrazac za zadaću na predmetu "**Uzorci dizajna**" ak. god. 2024./2025.

Ime i prezime studenta/ice: _____Patricio Poldrugač_____

Matični broj: _____0016142508_____

Dio A. Osnovni podaci o zadaći

R.br.	Pitanje	Odgovor	
1.	Grupa na seminaru:	G1	
2.	Broj i naziv zadaće:	3	Željeznički promet s voznim redom
3.	Procjena vremena za realizaciju bez decimala):	___20___ sati	
4.	Procjena % završenosti (bez decimala):	__40__ / 100%	
5.	Procjena bodova za izradu zadaće (1 decimala):	__6__ / (DZ3 - 15)	
6.	Žalim prezentirati zadaću:	NEMA PREZENTACIJE ZADAĆE!	
7.	Koji dijelovi iz opisa zadaće nisu realizirani:	Nisu realizirane funkcionalnosti vezane uz komande UKP2S, PSP2S i IRPS. i nije napravljena vlastita funkcionalnost korištenjem Command uzorka.	
8.	Postoji li dio zadaće koji vrijedi posebno istaknuti i zašto:	Ne.	
9.	Postoje li dijelovi zadaće koji imaju pogrešku u radu i koje:	Kod prodaje karata nije provjeravan status pruge kad taj dio sa State uzorkom nisam napravio.	
10.	Da li ste koristili tuđi programski kod u realizaciji zadaće izvan spomenutih izvora na nastavi:	Ne.	
11.	Da li ste koristili programska rješenja ili dijelove programskog koda od drugih kolega:	Ne.	

Dio B.1. Dokumentacija rješenja 2. zadaće (kopirano i nepromijenjeno)

Naziv uzorka dizajna	Klase koje sudjeluju u uzorku dizajna i u kojim ulogama	Status[1]	Opis razloga odabira uzorka dizajna
Singleton	Tvrtka	P	Odabrao sam ovaj uzorak jer je potrebno imati samo jednu instancu koja centralno upravlja svim podacima. Na taj način omogućavam globalni pristup istom objektu kroz cijelu aplikaciju, što pojednostavljuje upravljanje podacima i resursima. Dodani novi podaci i metode.
Singleton	UpraviteljGresaka	S	Za još jedan Singleton uzorak sam se odlučio jer mi je dovoljan samo jedan objekt koji centralno bilježi sve greške tijekom rada aplikacije. Na taj način osiguravam jedinstvenu kontrolu nad prikazivanjem i pohranjivanjem grešaka.
Builder	VoziloDirector, VoziloBuilder, VoziloBuilderImpl, Vozilo	S	Odabrao sam Builder uzorak za Vozilo kako bih omogućio fleksibilno definiranje različitih tipova vozila, s obzirom da svaki tip ima specifične opcionalne atribute. Builder olakšava dodavanje potrebnih atributa bez stvaranja dodatnih klasa za svaki tip vozila.
Builder	KompozicijaBuilder, KompozicijaBuilderImpl, Kompozicija	S	Odabrao sam ovaj uzorak za Kompoziciju jer omogućuje postupno dodavanje različitih vozila te olakšava sastavljanje složenih objekata, dok se provjere valjanosti obavljaju zasebno u drugim dijelovima sustava.
Factory Method	**Product**: CitacDatoteka **Creator:** DatotekaFactory **ConcreteProduct:** CitacStanica, CitacPruga, CitacVozila, CitacKompozicija, CitacOznakDana, CitacVoznogReda **ConcreteCreator:** StaniceFactory, PrugeFactory, VozilaFactory, KompozicijeFactory, OznakeDanaFactory, VozniRedFactory	P	Odabrao sam ovaj uzorak jer sam ga smatrao najpogodnijim za fleksibilno učitavanje datoteka. Iako sve datoteke slijede isti postupak učitavanja, svaka ima specifičan format zapisa pa sam za svaku datoteku implementirao zasebne validacije za sve atribute, osiguravajući ispravan unos podataka. Nove datoteke učitane i validirane na isti način kao i datoteke iz 1. zadaće.
Facade	SustavFacade	N	Odabrao sam Facade uzorak kako bih pojednostavio interakciju između glavne klase i složenijih podsustava poput Tvrtka i Chain of Responsibility. Facade pruža jedinstven API koji sakriva detalje implementacije, smanjuje kompleksnost i čini kod preglednijim. Unutar klase SustavFacade centraliziram inicijalizaciju sustava, uključujući učitavanje podataka iz datoteka, postavljanje lanca komandi i pripremu

[1] N – dodan u 2. zadaći, P – promijenjen u 2. zadaći, S – bez promjena u 2. zadaći

			ključnih modula poput korisničkog registra i mediatora. Osim toga, olakšava proširivanje sustava jer promjene unutar podsustava ne utječu na glavnu klasu, čime se postiže bolja organizacija i održavanje koda. Ovaj pristup omogućuje smanjenje dupliciranja logike i centralizaciju ključnih operacija u jednoj kohezivnoj klasi.
Composite	VozniRedComponent, EtapaLeaf, VlakComposite, VozniRedComposite	N	U opisu zadatka 2 striktno je navedeno da se struktura voznog reda mora temeljiti na uzorku dizajna Composite. Ovaj uzorak omogućuje modeliranje hijerarhijske strukture, pri čemu se vozni red sastoji od vlakova, vlakovi od etapa, a svaka etapa predstavlja pojedinačni list unutar hijerarhije.
Decorator	**Component:** Vrijeme **ConcreteComponent:** OsnovnoVrijeme **Decorator:** VrijemeDecorator **ConcreteDecorator:** KasnjenjeDecorator	N	Odabrao sam Decorator uzorak jer omogućuje proširivanje funkcionalnosti vremena bez mijenjanja osnovne implementacije. Trenutno je korišten za formatiranje i parsiranje vremena kroz klasu OsnovnoVrijeme, dok je uzorak pripremljen za buduće proširenje, poput dodavanja kašnjenja kroz KasnjenjeDecorator. Ovaj uzorak osigurava modularnost i fleksibilnost, omogućujući dodavanje novih funkcionalnosti bez izmjene postojećih klasa.
Chain of Responsibility	**Handler:** LanacKomandi **ConcreteHandler:** KomandaDK, KomandaDPK, KomandaIEV, KomandaIEVD, KomandaIK, KomandaIP, KomandaISP, KomandaISI2S, KomandaIV, KomandaIVRV, KomandaNO, KomandaPK	N	Za obradu komandi odabrao sam uzorak dizajna Chain of Responsibility jer omogućuje modularno i fleksibilno rukovanje različitim vrstama komandi. Svaka komanda (KomandaIK, KomandaIV, itd.) je implementirana kao zasebna klasa, što osigurava čistoću koda i olakšava dodavanje novih komandi bez mijenjanja postojećeg sustava. Refaktoriranjem uvođenjem ovog uzorka, postigao sam jasnu hijerarhiju i sekvencijalno procesiranje komandi, gdje svaka komanda odlučuje može li je obraditi ili prosljeđuje sljedećoj u lancu. Uzorak također smanjuje spregu između korisničkog unosa i logike obrade, omogućujući jednostavnu integraciju s Facade klasom koja inicijalizira i pokreće cijeli lanac komandi.
Observer	Observer, Korisnik (**ConcreteObserver**), Subject, KorisnickiRegistar (**ConcreteSubject**),	N	Odabrao sam Observer uzorak jer omogućuje praćenje promjena stanja i automatsko obavještavanje korisnika (Observera). KorisnickiRegistar održava listu korisnika koji prate dolaske vlakova ili stanica, a promjene se automatski reflektiraju na sve promatrače. Uzorak osigurava slabu povezanost između subjekta i promatrača, što olakšava proširenje i održavanje sustava te omogućuje intuitivnu implementaciju komandi DK, PK i DPK.
Mediator	Mediator, KonkretniMediator	N	U zadatku je bilo striktno zadano koristiti Mediator uzorak za implementaciju vlastite funkcionalnosti. Osmislio sam komandu NO za slanje obavijesti korisnicima pretplaćenim na određene vlakove, čime se omogućuje ciljana komunikacija o promjenama poput kašnjenja ili otkazivanja.

Dio B.2. Dokumentacija rješenja 3. zadaće

Naziv uzorka dizajna	Klase koje sudjeluju u uzorku dizajna i u kojim ulogama	Status[2]	Opis razloga odabira uzorka dizajna
Singleton	Tvrtka	P	Odabrao sam ovaj uzorak jer je potrebno imati samo jednu instancu koja centralno upravlja svim podacima. Na taj način omogućavam globalni pristup istom objektu kroz cijelu aplikaciju, što pojednostavljuje upravljanje podacima i resursima. Dodani novi podaci i metode.
Singleton	UpraviteljGresaka	S	Za još jedan Singleton uzorak sam se odlučio jer mi je dovoljan samo jedan objekt koji centralno bilježi sve greške tijekom rada aplikacije. Na taj način osiguravam jedinstvenu kontrolu nad prikazivanjem i pohranjivanjem grešaka.
Builder	VoziloDirector, VoziloBuilder, VoziloBuilderImpl, Vozilo	S	Odabrao sam Builder uzorak za Vozilo kako bih omogućio fleksibilno definiranje različitih tipova vozila, s obzirom da svaki tip ima specifične opcionalne atribute. Builder olakšava dodavanje potrebnih atributa bez stvaranja dodatnih klasa za svaki tip vozila.
Builder	KompozicijaBuilder, KompozicijaBuilderImpl, Kompozicija	S	Odabrao sam ovaj uzorak za Kompoziciju jer omogućuje postupno dodavanje različitih vozila te olakšava sastavljanje složenih objekata, dok se provjere valjanosti obavljaju zasebno u drugim dijelovima sustava.
Factory Method	**Product**: CitacDatoteka **Creator:** DatotekaFactory **ConcreteProduct:** CitacStanica, CitacPruga, CitacVozila, CitacKompozicija, CitacOznakDana, CitacVoznogReda **ConcreteCreator:** StaniceFactory, PrugeFactory, VozilaFactory, KompozicijeFactory, OznakeDanaFactory, VozniRedFactory	S	Odabrao sam ovaj uzorak jer sam ga smatrao najpogodnijim za fleksibilno učitavanje datoteka. Iako sve datoteke slijede isti postupak učitavanja, svaka ima specifičan format zapisa pa sam za svaku datoteku implementirao zasebne validacije za sve atribute, osiguravajući ispravan unos podataka. Nove datoteke učitane i validirane na isti način kao i datoteke iz 1. zadaće.
Facade	SustavFacade	P	Odabrao sam Facade uzorak kako bih pojednostavio interakciju između glavne klase i složenijih podsustava poput Tvrtka i Chain of Responsibility. Facade pruža jedinstven API koji sakriva detalje implementacije, smanjuje kompleksnost i čini kod preglednijim. Unutar klase SustavFacade centraliziram inicijalizaciju sustava, uključujući učitavanje podataka iz datoteka, postavljanje lanca komandi i pripremu

[2] N – dodan u 3. zadaći, P – promijenjen u 3. zadaći, S – bez promjena u 3. zadaći

			ključnih modula poput korisničkog registra i mediatora. Osim toga, olakšava proširivanje sustava jer promjene unutar podsustava ne utječu na glavnu klasu, čime se postiže bolja organizacija i održavanje koda. Ovaj pristup omogućuje smanjenje dupliciranja logike i centralizaciju ključnih operacija u jednoj kohezivnoj klasi. Projene u vidu novih metoda.
Composite	VozniRedComponent, EtapaLeaf, VlakComposite, VozniRedComposite	S	U opisu zadatka 2 striktno je navedeno da se struktura voznog reda mora temeljiti na uzorku dizajna Composite. Ovaj uzorak omogućuje modeliranje hijerarhijske strukture, pri čemu se vozni red sastoji od vlakova, vlakovi od etapa, a svaka etapa predstavlja pojedinačni list unutar hijerarhije.
Decorator	**Component:** Vrijeme **ConcreteComponent:** OsnovnoVrijeme **Decorator:** VrijemeDecorator **ConcreteDecorator:** KasnjenjeDecorator	S	Odabrao sam Decorator uzorak jer omogućuje proširivanje funkcionalnosti vremena bez mijenjanja osnovne implementacije. Trenutno je korišten za formatiranje i parsiranje vremena kroz klasu OsnovnoVrijeme, dok je uzorak pripremljen za buduće proširenje, poput dodavanja kašnjenja kroz KasnjenjeDecorator. Ovaj uzorak osigurava modularnost i fleksibilnost, omogućujući dodavanje novih funkcionalnosti bez izmjene postojećih klasa.
Chain of Responsibility	**Handler:** LanacKomandi **ConcreteHandler:** KomandaDK, KomandaDPK, KomandaIEV, KomandaIEVD, KomandaIK, KomandaIP, KomandaISP, KomandaISI2S, KomandaIV, KomandaIVRV, KomandaNO, KomandaPK	p	Za obradu komandi odabrao sam uzorak dizajna Chain of Responsibility jer omogućuje modularno i fleksibilno rukovanje različitim vrstama komandi. Svaka komanda (KomandaIK, KomandaIV, itd.) je implementirana kao zasebna klasa, što osigurava čistoću koda i olakšava dodavanje novih komandi bez mijenjanja postojećeg sustava. Refaktoriranjem uvođenjem ovog uzorka, postigao sam jasnu hijerarhiju i sekvencijalno procesiranje komandi, gdje svaka komanda odlučuje može li je obraditi ili prosljeđuje sljedećoj u lancu. Uzorak također smanjuje spregu između korisničkog unosa i logike obrade, omogućujući jednostavnu integraciju s Facade klasom koja inicijalizira i pokreće cijeli lanac komandi. Nove komande dodane.
Observer	Observer, Korisnik (**ConcreteObserver**), Subject, KorisnickiRegistar (**ConcreteSubject**),	P	Odabrao sam Observer uzorak jer omogućuje praćenje promjena stanja i automatsko obavještavanje korisnika (Observera). KorisnickiRegistar održava listu korisnika koji prate dolaske vlakova ili stanica, a promjene se automatski reflektiraju na sve promatrače. Uzorak osigurava slabu povezanost između subjekta i promatrača, što olakšava proširenje i održavanje sustava te omogućuje intuitivnu implementaciju komandi DK, PK i DPK. Ispravljeno slanje obavijesti pretplaćenim korisnicima.
Mediator	Mediator, KonkretniMediator	S	U zadatku je bilo striktno zadano koristiti Mediator uzorak za implementaciju vlastite funkcionalnosti. Osmislio sam komandu NO za slanje obavijesti korisnicima pretplaćenim na određene vlakove, čime se omogućuje ciljana

			komunikacija o promjenama poput kašnjenja ili otkazivanja.
Strategy	**Strategy:** StrategijaCijene **ConcreteStrategy:** OsnovnaStrategija, VikendStrategija, VlakStrategija, WebAplikacijaStrategija, KompozitnaStrategija **Context:** KontekstCijena		Bilo je zadano da se izračun cijene karte s obzirom na način kupovine treba temeljiti na uzorku dizajna Strategy.
Memento	**Originator:** ProdajaKarata **Memento:** KartaMemento **Caretaker:** SkladisteKarata		Bilo je zadano da svaku kupovinu karte potrebno je pohraniti kako bi se moglo do nje kasnije pristupiti i da se to treba temeljiti na uzorku dizajna Memento.

Dio C.1. Opis promjena u odnosu na prethodnu zadaću

Dodane dvije nove pomoćne klase. Enum „NacinKupovine" i „Cjenik" kao klasa koja sadrži sve cijene i popuste. Napravljena komanda SSV vezana uz simulaciju iz prošle zadaće jer tad nije bila implementirana. Popravljeno slanje obavijesti pretplaćenim korisnicima i testirano sada kada postoji simulacija. Napravljeno određivanje cijene vožnje putnika vlakom €/km, popust za subotu i nedjelju, % popusta za kupovinu karte putem web/mobilne aplikacije i % uvećanja za kupovinu karte u vlaku. Napravljena i kupovina karte za putovanje između dviju stanica određenim vlakom na određeni datum s odabranim načinom kupovanja karte. I napravljen ispis kupljenih karata za putovanje vlakom.

Dio C.2. Opis funkcionalnosti za uzorak dizajna Command

Nije napravljeno.

Dio D. Dijagram klasa s naglašavanjem klasa koje sudjeluju u pojedinom uzorku dizajna