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:	2	Željeznički promet s voznim redom
3.	Procjena vremena za realizaciju bez decimala):	___40___ sati	
4.	Procjena % završenosti (bez decimala):	___70__ / 100%	
5.	Procjena bodova za izradu zadaće (1 decimala):	__9.1___ / (DZ2 - 13)	
6.	Žalim prezentirati zadaću:	NE	
7.	Koji dijelovi iz opisa zadaće nisu realizirani:	Nisu realizirane funkcionalnosti vezane uz komande:IVI2S i SSV.	
8.	Postoji li dio zadaće koji vrijedi posebno istaknuti i zašto:	Ne.	
9.	Postoje li dijelovi zadaće koji imaju pogrešku u radu i koje:	Vjerovatno neke detaljne provjere kod učitavanja voznog reda vezane uz vremena vlakova i brisanja neispravnih iz voznog reda.	
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 1. zadaće (kopirano i nepromijenjeno)

Naziv uzorka dizajna	**Klase koje sudjeluju u uzorku dizajna**	**Opis razloga odabira uzorka dizajna**
Singleton	Tvrtka	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.
Singleton	UpraviteljGresaka	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	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	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	CitacDatoteka, DatotekaFactory, CitacStanica, StaniceFactory, CitacPruga, PrugeFactory, CitacVozila, VozilaFactory, CitacKompozicija, KompozicijeFactory	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.

Dio B.2. Dokumentacija rješenja 2. zadaće

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 ključnih modula poput korisničkog registra i

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

			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 C.1. Opis promjena u odnosu na prethodnu zadaću

U odnosu na prvu zadaću, ispravio sam nekoliko nedostataka. Dodana je provjera kod kreiranja pruga da moraju imati minimalno dvije stanice, čime sam riješio problem rušenja programa prilikom obrade komandi. Također, nepoznate opcije sada se tretiraju kao greške, što je bio još jedan nedostatak iz prve zadaće.

Uz ispravke, implementirane su nove funkcionalnosti i dodani uzorci dizajna. Obrada komandi refaktorirana je korištenjem uzorka **Chain of Responsibility**, što omogućuje modularno i fleksibilno rukovanje komandama. Dodana su učitavanja i validacije za dvije nove datoteke kroz proširenje uzorka **Factory Method**. Za organizaciju voznog reda implementiran je **Composite**, dok su za vrijeme uvedene dekoracije pomoću **Decorator** uzorka. Uzorak **Observer** korišten je za obavještavanje korisnika o promjenama, a uzorak **Mediator** za centralizaciju komunikacije i obavijesti korisnicima.

Također, uveden je uzorak **Facade**, koji centralizira inicijalizaciju sustava, upravljanje podacima i pokretanje lanca komandi, čime je pojednostavljena organizacija i održavanje koda.

Dio C.2. Opis funkcionalnosti za uzorak dizajna Mediator

Za korištenje uzorka dizajna Mediator osmislio sam komandu čija bi funkcionalnost bila obavještavanje korisnika o promjenama u statusu vlakova koje prate. Funkcionalnost omogućuje slanje obavijesti samo korisnicima koji su prethodno pretplaćeni na određeni vlak, čime se osigurava ciljana komunikacija. Pretplata na vlak omogućuje korisniku praćenje specifičnih obavijesti vezanih uz kašnjenja ili druge promjene u voznom redu.
Mediator uzorak je implementiran kako bi centralizirao komunikaciju između korisnika i obavijesti o vlakovima. Sva interakcija između korisnika i sustava odvija se kroz Mediator, koji služi kao posrednik između korisnika i vlakova te omogućuje lako upravljanje pretplatama i obavijestima.

Funkcionalnost
Korisnici se mogu pretplatiti na praćenje vlakova pomoću komande DPK, dok se obavijesti korisnicima šalju pomoću komande NO. Sustav provjerava postoje li vlakovi i korisnici prije nego što se obavi bilo kakva akcija, a obavijesti se šalju isključivo korisnicima pretplaćenim na određeni vlak. o NO znači "Nova Obavijest".

Sintaksa:

- NO oznakaVlaka - poruka

Primjer:

- NO 3301 - Vlak 3301 kasni 15 minuta.
- NO 3302 - Vlak 3302 je otkazan.

Opis primjera:

- Korisnicima koji su pretplaćeni (dodani komandom DPK) na vlak 3301 šalje se poruka: "Vlak 3301 kasni 15 minuta."
- Ispisuje se: Obavijest za korisnika Pero Peric: Vlak 3301 kasni 15 minuta
- Ako nijedan korisnik nije pretplaćen na taj vlak, ispisujemo: ”Nema korisnika pretplaćenih na vlak 3301.”
- Ako vlak ne postoji ispisuje se greška: Greška: Vlak s oznakom *oznakaVlaka* ne postoji.

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