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

Obrazac za zadaću na predmetu "**Uzorci dizajna**" ak. god. 2025./2026.

Ime i prezime studenta/ice: Viktor Lovrić

Matični broj: 0016154953

Dio A. Osnovni podaci o zadaći

R.br.	Pitanje	Odgovor	
1.	Grupa na seminaru:	G1	
2.	Broj i naziv zadaće:	3	Turistička agencija
3.	Procjena vremena za realizaciju bez decimala):	12 sati	
4.	Procjena % završenosti (bez decimala):	85 / 100%	
5.	Procjena bodova za izradu zadaće (1 decimala):	11 / (DZ3 - 15)	
6.	Žalim prezentirati zadaću:	NEMA PREZENTACIJE ZADAĆE!	
7.	Koji dijelovi iz opisa zadaće nisu realizirani:	Svi su realizirani	
8.	Postoji li dio zadaće koji vrijedi posebno istaknuti i zašto:	Command uzorak mi nema veliku sinergiju s Memento uzorkom što se tiče undo operacije i općenito vjerojatno nije implementiran po GOF-u	
9.	Postoje li dijelovi zadaće koji imaju pogrešku u radu i koje:	Koliko sam svjestan, ne	
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	
12.	Da li ste koristili alat/e generativne umjetne inteligencije u izradi zadaće, ako jeste koji/e ste koristili i za koje potrebe ste ga/ih koristili:	Alati Gemini 3 Pro i GPT-5.2 za ideju za CoR funkcionalnost	

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	RepozitorijPodataka	P	RepozitorijPodataka sadrži jedinstvene podatke važne za sinkroniziran rad cijele aplikacije poput objekta koji sadrži sve aranžmane i rezervacije, određenih zastavica i slično te zbog toga mora imati samo jednu instancu i biti globalno dostupan u cijelom sustavu.
Builder	AranzmanBuilder, AranzmanBuilderConcrete, Aranzman	P	Aranzman je vrlo kompleksan objekt sa vrlo puno atributa te je stoga Builder korišten kako bi se olakšao proces stvaranja objekta aranžmana na klijentskoj strani koda.
Factory Method	CsvParserCreator, CsvParser, AranzmanCsvParserCreator, AranzmanCsvParser, RezervacijaCsvParserCreator, RezervacijaCsvParser	P	Parsiranje CSV-a se dijeli na parsiranje aranžmana i parsiranje rezervacija. Bile su potrebne klase koje će dijeliti jedan dio implementacije, ali imati lokalizirano znanje specifično za svaku podklasu tj. za validiranje i stvaranje specifičnog objekta.
Factory Method	FormaterFactory, FormaterCreator, Formater, IROFormater, IROFormaterCreator, IRTAFormater, IRTAFormaterCreator, IRTAOtkazFormater, IRTAOtkazFormaterFormater, ITAKFormater, ITAKFormaterCreator, ITAPFormater, ITAPFormaterCreator, ITASFormater, ITASFormaterCreator	P	Svaka naredba koja ispisuje objekt ili listu objekata u tablici ima svoje specifične stupce i tipove podataka koje ispisuje. Kako bi olakšao mijenjanje specifične implementacije ili dodavanje novih formata ispisa, svaki ispis ima svoju implementaciju u klasi koja nasljeđuje apstraktnu Formater klasu.
Factory Method	CsvObjectUnositelj, CsvObjectCreator, CsvAranzmanUnositelj, CsvAranzmanUnositeljCreator, CsvRezervacijaUnositelj, CsvRezervacijaUnositeljCreator	N	Validiranje i unošenje objekata Aranžman i Rezervacija treba biti ponovno iskoristivo s obzirom da se može odvijati više puta. Factory Method sam iskoristio kako bih stvorio klase koje imaju lokalizirano znanje kako validirati i unijeti objekte tipa za koje su zadužene.
Singleton	ParsiranjePomagac	N	U _lib modulu mi je trebao način kako bih pratio broj grešaka neovisno koliko se puta modul ponovno poziva, te i mjesto gdje se mogu pohranjivati ispravni redovi nakon parsiranja kako bi se kasnije mogli

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

			dohvatiti i poslati.
Facade	CsvParsiranjeFacade	N	Ovaj uzorak dizajna je bio obavezan, implementiran je na način da parsira dobiveni csv i vraća samo redove koji su ispravni, ispisujući pri tome greške kako moduli ne bi bili ovisni jedan o drugome.
Composite	AranzmanKomponenta, AranzmanKolekcija, Aranzman, Rezervacija	N	Ovaj uzorak dizajna je bio obavezan, trebao sam način kako bih imao strukturu gdje svaki aranžman sadrži svoje rezervacije i kako bi se pojednostavilo iteriranje i pozivanje funkcija nad elementima djece.
State	Aranzman, AranzmanStatus, AranzmanUPripremi, AranzmanAktivan, AranzmanPopunjen, AranzmanOtkazan	N	Ovaj uzorak dizajna je bio obavezan, htio sam delegirati upravljanje životnim ciklusom aranžmana i njegovih rezervacija na specifične statuse u kojima je aranžman u trenutku obavljanja neke operacije kako bih imao lokaliziranu i lakše održivu poslovnu logiku s puno pravila.
State	Rezervacija, RezervacijaStatus, RezervacijaNova, RezervacijaPrimljena, RezervacijaAktivna, RezervacijaNaCekanju, RezervacijaOdgodena, RezervacijaOtkazana	N	Ovaj uzorak dizajna je bio obavezan, također sam htio delegirati upravljanje životnom ciklusom rezervacije, tj. njenim otkazivanjem tako da se rezervacije ponašaju drugačije pri otkazivanju ovisno u kojem su stanju.
Decorator	Formater, FormaterDecorator, PodnozjeRezervacijaFormatDecorator, PodnozjeAranzmanFormatDecorator, IROFormater, IRTAFormater, IRTAOtkazFormater, ITAKFormater	N	Ovaj uzorak dizajna je bio obavezan, htio sam pokazati vlastitu funkcionalnost koristeći ovaj uzorak dizajna.

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	RepozitorijPodataka (***Singleton***)	S	RepozitorijPodataka sadrži jedinstvene podatke važne za sinkroniziran rad cijele aplikacije poput objekta koji sadrži sve aranžmane i rezervacije, određenih zastavica i slično te zbog toga mora imati samo jednu instancu i biti globalno dostupan u cijelom sustavu.
Builder	AranzmanBuilder *(**Builder**)*, AranzmanBuilderConcrete *(**ConcreteBuilder**)*, Aranzman *(**Product**)*	S	Aranzman je vrlo kompleksan objekt sa vrlo puno atributa te je stoga Builder korišten kako bi se olakšao proces stvaranja objekta aranžmana na klijentskoj strani koda.
Factory Method	CsvParserCreator *(**Creator**)*, CsvParser *(**Product**)*, AranzmanCsvParserCreator *(**ConcreteCreator**)*, AranzmanCsvParser *(**ConcreteProduct**)*, RezervacijaCsvParserCreator *(**ConcreteCreator**)*, RezervacijaCsvParser *(**ConcreteProduct**)*	S	Parsiranje CSV-a se dijeli na parsiranje aranžmana i parsiranje rezervacija. Bile su potrebne klase koje će dijeliti jedan dio implementacije, ali imati lokalizirano znanje specifično za svaku podklasu tj. za validiranje i stvaranje specifičnog objekta.
Factory Method	FormaterFactory *(**Factory**)*, FormaterCreator *(**Creator**)*, Formater *(**Product**)*, IROFormater *(**ConcreteProduct**)*, IROFormaterCreator *(**ConcreteCreator**)*, IRTAFormater *(**ConcreteProduct**)*, IRTAFormaterCreator *(**ConcreteCreator**)*, IRTAOtkazFormater *(**ConcreteProduct**)*, IRTAOtkazFormaterCreator *(**ConcreteCreator**)*, ITAKFormater *(**ConcreteProduct**)*, ITAKFormaterCreator *(**ConcreteCreator**)*, ITAPFormater *(**ConcreteProduct**)*, ITAPFormaterCreator *(**ConcreteCreator**)*, ITASFormater *(**ConcreteProduct**)*, ITASFormaterCreator *(**ConcreteCreator**)*	S	Svaka naredba koja ispisuje objekt ili listu objekata u tablici ima svoje specifične stupce i tipove podataka koje ispisuje. Kako bi olakšao mijenjanje specifične implementacije ili dodavanje novih formata ispisa, svaki ispis ima svoju implementaciju u klasi koja nasljeđuje apstraktnu Formater klasu.
Factory Method	CsvObjectUnositelj *(**Product**)*, CsvObjectCreator *(**Creator**)*,	S	Validiranje i unošenje objekata Aranžman i Rezervacija treba biti ponovno iskoristivo s

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

	CsvAranzmanUnositelj ***(ConcreteProduct)***, CsvAranzmanUnositeljCreator ***(ConcreteCreator)***, CsvRezervacijaUnositelj ***(ConcreteProduct)***, CsvRezervacijaUnositeljCreator ***(ConcreteCreator)***		obzirom da se može odvijati više puta. Factory Method sam iskoristio kako bih stvorio klase koje imaju lokalizirano znanje kako validirati i unijeti objekte tipa za koje su zadužene.
Singleton	ParsiranjePomagac ***(Singleton)***	S	U _lib modulu mi je trebao način kako bih pratio broj grešaka neovisno koliko se puta modul ponovno poziva, te i mjesto gdje se mogu pohranjivati ispravni redovi nakon parsiranja kako bi se kasnije mogli dohvatiti i poslati.
Facade	CsvParsiranjeFacade (***Facade***)	S	Ovaj uzorak dizajna je bio obavezan, implementiran je na način da parsira dobiveni csv i vraća samo redove koji su ispravni, ispisujući pri tome greške kako moduli ne bi bili ovisni jedan o drugome.
Composite	AranzmanKomponenta ***(Component)***, AranzmanKolekcija (***Composite***), Aranzman (***Composite***), Rezervacija ***(Leaf)***	S	Ovaj uzorak dizajna je bio obavezan, trebao sam način kako bih imao strukturu gdje svaki aranžman sadrži svoje rezervacije i kako bi se pojednostavilo iteriranje i pozivanje funkcija nad elementima djece.
State	Aranzman ***(Context)***, AranzmanStatus ***(State)***, AranzmanUPripremi ***(ConcreteState)***, AranzmanAktivan ***(ConcreteState)***, AranzmanPopunjen ***(ConcreteState)***, AranzmanOtkazan ***(ConcreteState)***	S	Ovaj uzorak dizajna je bio obavezan, htio sam delegirati upravljanje životnim ciklusom aranžmana i njegovih rezervacija na specifične statuse u kojima je aranžman u trenutku obavljanja neke operacije kako bih imao lokaliziranu i lakše održivu poslovnu logiku s puno pravila.
State	Rezervacija ***(Context)***, RezervacijaStatus ***(State)***, RezervacijaNova ***(ConcreteState)***, RezervacijaPrimljena ***(ConcreteState)***, RezervacijaAktivna ***(ConcreteState)***, RezervacijaNaCekanju ***(ConcreteState)***, RezervacijaOdgodena ***(ConcreteState)***, RezervacijaOtkazana ***(ConcreteState)***	S	Ovaj uzorak dizajna je bio obavezan, također sam htio delegirati upravljanje životnom ciklusom rezervacije, tj. njenim otkazivanjem tako da se rezervacije ponašaju drugačije pri otkazivanju ovisno u kojem su stanju.
Decorator	Formater ***(Component)***, FormaterDecorator ***(Decorator)***, PodnozjeRezervacijaFormatDecorator (***ConcreteDecorator***), PodnozjeAranzmanFormatDecorator (***ConcreteDecorator***),	S	Ovaj uzorak dizajna je bio obavezan, htio sam pokazati vlastitu funkcionalnost koristeći ovaj uzorak dizajna.

	IROFormater (***ConcreteComponent***), IRTAFormater (***ConcreteComponent***), IRTAOtkazFormater (***ConcreteComponent***), ITAKFormater (***ConcreteComponent***)		
Strategy	RezervacijaUpravitelj (***Strategy***), JdrRezervacijaUpravitelj (***ConcreteStrategy***), VdrRezervacijaUpravitelj (***ConcreteStrategy***), NistaRezervacijaUpravitelj (***ConcreteStrategy***)	N	Ovaj uzorak dizajna je bio obavezan, a omogućuje da se prema početnim zastavicama odredi koja će biti primijenjena poslovna logika za upravljanjem rezervacijama
Null Object	NistaRezervacijaUpravitelj (***Null Object***)	N	Ovaj uzorak dizajna je bio obavezan, a koristi se sa Strategy uzorkom kako ne bi bilo nikakvih ograničenja kod upravljanja rezervacijama nego je sve dopušteno ovisno o ulaznim zastavicama
Visitor	PptarVisitor (***Visitor***), PptarAranzmanVisitor (***ConcreteVisitor***), PptarRezervacijaVisitor (***ConcreteVisitor***), AranzmanKolekcija (***ObjectStructure***), PptarElement (***Element***), Aranzman (***ConcreteElement***), Rezervacija (***ConcreteElement***)	N	Ovaj uzorak dizajna je bio obavezan, omogućava pretraživaje Aranzman i Rezervacija objekata bez da ti objekti sadrže logiku pretraživanja sebe
Memento	Aranzman (***Originator***), AranzmanMemento (***Memento***), MementoSpremiste (***Caretaker***)	N	Ovaj uzorak dizajna je bio obavezan, omogućava „snapshotanje" aranžmana i njegovih rezervacija i povrat stanja aranžmana i njegovih rezervacija.
Command	KomandePomocnik (***Invoker***), STARCommand (***Command***), PstarCommand (***ConcreteCommand***), VstarCommand (***ConcreteCommand***), MementoSpremiste (***Receiver***)	N	Ovaj uzorak dizajna je bio obavezan, svaka konkretna komanda sadrži logiku i pozivanje MementoSpremiste objekta.
Chain of Responsibility	BrisanjeHandler (***Handler***), AutentifikacijaHandler (***ConcreteHandler***), PohranaHandler (***ConcreteHandler***), IzvrsiBrisanjeHandler (***ConcreteHandler***)	N	Ovaj uzorak dizajna je bio obavezan, proširuje postojeću funkcionalnost brisanja tako što slijedno vrši autentifikaciju i backup aranžmana prije brisanja

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

Iako sam mijenjao neku logiku unutar klasa prethodne zadaće, nisam mijenjao samu strukturu uzoraka pa time nema promjena što se tiče samih uzoraka.

Dio C.2. Opis funkcionalnosti za uzorak dizajna Decorator (kopirano i nepromijenjeno iz 2. zadaće)

Funkcionalnost koja koristi uzorak dizajna Decorator je dodavanje statistika u podnožje tablica za komande IRO, IRTA i ITAK.
Komanda je sintakse: **POD**, primjer: „**POD**”.
Izvršavanjem ove komande u aplikaciji uključuje ili isključuje se dodano statističko podnožje tablicama IRO, IRTA i ITAK. Prilikom pokretanja aplikacije je ovo podnožje isključeno. Statističko podnožje ispisuje koliko je rezervacija (IRO, IRTA) ili aranžmana (ITAK) ukupno prikazano u tablici te broj rezervacija/aranžmana po statusima. Primjerice, ukoliko izvršavanjem komande ITAK dobijemo tablicu u kojoj su 50 aranžmana od kojih su 10 primljeni, 30 aktivni i 10 otkazani, ukoliko je prije toga ikada izvršena komanda „POD” i time uključeno statističko podnožje, ispod tablice će se prikazati da je ukupan broj aranžmana 50, da ima 10 primljenih, 30 aktivnih i 10 otkazanih.
Ova funkcionalnost je odabrana jer za svaku od tablica (IRO, IRTA, ITAK) koje u stvarnom sustavu mogu imati nepregledan broj redova pruža jednostavan sažetak koji može biti od velike koristi, ali također i jer se prilikom razvoja aplikacije i njene evaluacije može jednostavnije provjeriti očekivani rezultat s dobivenim bez ručnog brojanja ili pregleda redova i njihovih statusa.

Dio C.3. Opis funkcionalnosti za uzorak dizajna Chain-of-Responsibility

Chain-of-Responsibility je korišten kako bi se dodala funkcionalnost na komandu ***BP*** tj. na brisanje aranžmana ili rezervacija iz sustava. S obzirom da je brisanje nepovratna i destruktivna radnja, CoR je korišten kako bi se proveo niz koraka prije stvarnog brisanja iz sustava. Prilikom pokretanja naredbe ***BP*** pokreće se prvi korak u nizu što je autentifikacija gdje se korisnika traži da upiše lozinku kako bi potvrdio brisanje. Lozinka je hardkodirana kao „**admin123**" čisto radi primjera. Ako korisnik upiše krivu lozinku, proces ovdje staje i brisanje se ne izvršava. Ako je lozinka ispravna, pokreće se drugi korak u procesu. Drugi korak je pohrana svih aranžmana u Memento objekte koji već nemaju snapshot spremljen. Nakon ovog koraka se brišu odabrani podatci iz sustava. S obzirom da su svi podatci spremljeni, moguće ih je vratiti koristeći naredbu ***VSTAR*** ukoliko se želi vratiti neki od obrisanih podataka.

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