NoSQL baze podataka – projekt u MongoDB

Bašić, Erik

Sveučilište u Rijeci – Fakultet informatike i digitalnih tehnologija

Diplomski studij informatike – smjer informacijski i komunikacijski sustavi

Erik Bašić

NoSQL baze podataka – projekt u MongoDB

Diplomski rad

Mentor: doc. dr. sc. Danijela Jakšić

Rijeka, rujan 2023.

Sveučilište u Rijeci
Fakultet informatike
i digitalnih tehnologija
www.inf.uniri.hr

Rijeka, 12. lipanj 2023.

Zadatak za diplomski rad

Pristupnik: Erik Bašić

Naziv diplomskog rada: NoSQL baze podataka - projekt u MongoDB

Naziv diplomskog rada na eng. jeziku: NoSQL data stores - MongoDB project

Sadržaj zadatka: Cilj je diplomskog rada istražiti NoSQL baze podataka s fokusom na MongoDB. NoSQL baze podataka dobivaju sve veću pažnju posljednjih godina zbog mogućnosti obrade veće količine nestrukturiranih podataka. U prvom djelu ovog rada biti će dan pregled NoSQL baza podataka, objasniti će se osnovni koncepti, koje sve vrste NoSQL baza postoje te prednosti i nedostaci u usporedbi s relacijskim bazama podataka. U drugom dijelu rada istražiti će se MongoDB, koja je jedna od najpopularnijih dokumentnih nerelacijskih baza koja sprema podatke u JSON formatu. U sklopu ovog dijela istražiti će se i usporediti dva alata. Prvi alat je MongoDB Atlas koji omogućava rad u oblaku, a drugi alat je MongoDB Compass koji sadrži grafičko korisničko sučelje koje omogućava vizualnu interakciju sa kolekcijama i dokumentima.

Mentor:

Doc. dr. sc. Danijela Jakšić

Komentor:

Voditeljica za diplomske radove:

Prof. dr. sc. Ana Meštrović

Zadatak preuzet: 12. lipanj 2023.

(potpis pristupnika)

Adresa: Radmile Matejčić 2
51000 Rijeka, Hrvatska
Tel: **+385(0)51 584 700**
E-mail: **ured@inf.uniri.hr**
OIB: **64218323816**
IBAN: **HR1524020061400006966**

Sažetak

Unutar ovog diplomskog rada opisane su nerelacijske baze podataka s naglaskom na MongoDB. Opisano je što su to nerelacijske baze, koje vrste postoje, na koji način rade i kakve prednosti i nedostatke donose u odnosu na relacijske baze. Nakon toga prikazan je i opisan model podataka koji će se koristiti pri izradi projekta u MongoDB. Slijede koraci preuzimanja MongoDB-a na lokalno računalo te postavljanje okruženja. Projekt u MongoDB-u će se izraditi paralelnim korištenjem MongoDB Compass-a i MongoDB Shell-a. Prikazani su načini unosa dokumenata u oba programa. Nakon toga objašnjeni su načini izrade jednostavnih i kompleksnijih upita u MongoDB-u. Na kraju prikazan je rad u MongoDB Atlasu koji predstavlja MongoDB bazu u oblaku. Objašnjeno je otvaranje računa te prikazan je način izrade grafikona nad podacima.

Ključne riječi

MongoDB, Compass, Shell, kolekcija, dokument

Abstract

Within this thesis, non-relational databases are described with an emphasis on MongoDB. It is described what non-relational databases are, what types exist, how they work and what advantages and disadvantages they bring compared to relational databases. After that, the data model that will be used when creating the project in MongoDB is presented and described. The following are the steps for downloading MongoDB to your local computer and setting up the environment. The project in MongoDB will be created using MongoDB Compass and MongoDB Shell in parallel. The ways of entering documents in both programs are shown. After that, the ways of creating simple and more complex queries in MongoDB were explained. At the end, the work in MongoDB Atlas, which represents the MongoDB database in the cloud, is presented. Opening an account is explained and the way to create a graph of the data is shown.

Key words

MongoDB, Compass, Shell, collection, document

Sadržaj

1. UVOD

Cilj ovog rada je istražiti nerelacijsku bazu podataka MongoDB. Potrebno je istražiti način i koncept rada ovakve nerelacijske baze podataka. Cilj je izraditi vlastiti model podataka te ga unijeti u bazu putem odgovarajućih alata. Alati koji će se koristiti za unos modela i stvaranje upita su MongoDB Compass te MongoDB Shell. Nakon unosa i stvaranja upita istražit će se i alat u oblaku naziva MongoDB Atlas.

U 2. poglavlju će biti opisane nerelacijske baze podataka. Objasniti će se kakve su to baze i što one donose u odnosu na klasične relacijske baze podataka. Prikazati će se neke od vrsta takvih bazi te sve prednosti i mane koje one donose.

U 3. poglavlju prikazati će se model podataka koji će se koristiti pri radu s MongoDB bazom. Objasniti će se način rada modela te svaka od kolekcija zasebno.

U 4. poglavlju započeti će rad s MongoDB bazom. Prikazati će se način instalacije programa MongoDB Compass te MongoDB Shell lokalno. Slijedi prikaz rada s MongoDB Compass-om u kojemu će se prikazati način spajanja na bazu te načini unosa jednog ili više dokumenata putem grafičkog sučelja. Usporedno sa time prikazati će se način spajanja na bazu putem Shell-a koristeći Windows PowerShell te načini unosa jednog ili više dokumenata. Nakon toga slijedi prikaz jednostavnijih upita u kojima će se kombinirati Compass i Shell. Prikazati će se jednostavniji upiti nad jednom kolekcijom. Zatim će se ti jednostavniji upiti zakomplicirati korištenjem operatora pomoću kojih je moguće izvoditi matematičke i logičke operacije.

U 5. poglavlju prikazati će način povezivanja dokumenata. Iako se ovdje radi o nerelacijskoj bazi koja nema strogo definirane veze i dalje se mogu stvoriti veze pomoću kojih će biti moguće izvoditi kompleksnije upite nad više kolekcija. Nakon povezivanja dokumenata prikazati će se par kompleksnijih upita u kojima će se spajati dvije kolekcije. U ovim upitima također će biti prikazano i dodavanje vlastitih izračuna.

U 6. poglavlju prikazati će se način ažuriranja i brisanja dokumenata s glavnim fokusom na Shell jer je brisanje i ažuriranje jednostavnije u Compass-u. Prikazati će se načini brisanja i ažuriranja jednog ili više dokumenata putem naredbi te način na koji uz to možemo koristiti i operatore.

U 7. poglavlju prikazati će se MongoDB Atlas koji je zapravo MongoDB baza u oblaku. Prikazati će se način stvaranja besplatnog računa te neke od opcija koje on nudi u usporedbi s Compass-om i Shell-om. Zatim će se prikazati način stvaranja grafikona putem ovog alata.

2. NOSQL BAZE PODATAKA

Pojam NoSQL (engl. Not Only SQL) prvi put je korišten 1998. godine za relacijsku bazu podataka koja je izostavila korištenje SQL-a. Pojam se ponovno koristi 2009. godine na konferenciji zagovornika nerelacijskih baza podataka održane u San Franciscu. Na toj konferenciji kritizirane su relacijske baze podataka – sudionici konferencije opisuju ih kao spore i skupe. Tvrde da Web 2.0. startupi započinju svoje poslovanje bez Oraclea i MySQL-a. Umjesto takvih baza, izgrađuju vlastita spremišta podataka pod utjecajem Amazonovog Dynama i Googleovog Bigtable-a koja služe za pohranjivanje i obradu ogromne količine nestrukturiranih podataka [1].

Wikipedija NoSQL baze podataka definira kao mehanizme za pohranjivanje i dohvaćanje podataka koji su modelirani na drugačije načine u odnosu na relacijske baze podataka [2]. To znači da su podaci nerelacijski (nestrukturirani ili polustrukturirani) i da ne koriste SQL kao upitni jezik. Ovakve baze pokušavaju riješiti probleme skalabilnosti i dostupnosti u suprotnosti od atomičnosti i konzistentnosti. Skalabilnost podrazumijeva sposobnost sustava da se nosi sa porastom količine podataka [4]. Relacijske baze podataka sadrže ACID svojstva (atomičnost, konzistentnost, izoliranost i trajnost), a nerelacijske baze podataka sadrže BASE svojstva [3]:

- Osnovna dostupnost (engl. Basic availabillity) – svaki zahtjev će dobiti odgovor koji može biti uspješan ili neuspješan)
- Promjenjivo stanje (engl. Soft state) – stanje sistema se može mijenjati tokom vremena
- Eventualna konzistentnost (engl. Eventual consistency) – baza podataka može biti privremeno nekonzistetna no biti će konzistentna u jednom trenutku.

Uz rješavanje problema skalabilnosti, nerelacijske baze podataka imaju i sljedeće prednosti [3]:

- Prikaz podataka bez sheme – skoro svaka NoSQL baza sadrži ovakav prikaz implementiran. To znači da nije potrebno razmišljati o definiranju strukture podataka jer je olakšano dodavanje novih polja ili čak ugnježđivanja podataka, u slučaju npr. JSON dokumenata
- Razvojno vrijeme – mnogi tvrde da se izbjegavanjem korištenja kompleksnijih SQL upita smanjuje vrijeme razvoja baze podataka
- Brzina – NoSQL baze pružaju puno veću brzinu isporuke podataka

Postoji više vrsta nerelacijskih baza podataka a neke od tih vrsta su [3]:

- Stupčaste baze
- Dokumentne baze
- Ključ – vrijednost baze
- Graf baze

Stupčaste nerelacijske baze spremaju podatke u obliku stupaca za razliku od relacijskih baza gdje se podaci spremaju u redove.

U relacijskoj bazi podataka podaci bi bili prikazani u obliku dvodimenzionalnog polja:

Tablica 1 Relacijska tablica

OIB_djelatnika	Ime_djelatnika	Prezime_djelatnika	Adresa_djelatnika	Email_djelatnika
43940384294	Ivana	Grgurić	Krešimirova 4	Ivana@gmail.com
65122789569	Milivoj	Milić	Tizianova 12	milivoj@gmail.com
98934126541	Josip	Jurić	Rastočine 16	milivoj@gmail.com

U stupčastoj bazi podaci su pohranjeni na sljedeći način:

```
43940384294, 65122789569, 98934126541

Ivana, Milivoj, Josip

Grgurić, Milić, Jurić

Krešimirova 4, Tizianova 12, Rastočine 16

Ivana@gmail.com, milivoj@gmail.com, milivoj@gmail.com
```

Prednost ovakvih baza je to što omogućavaju lako dodavanje novih stupaca bez brige da moramo popuniti vrijednost svakog retka što omogućava dodatnu fleksibilnost. Prilikom izračunavanja maksimuma, minimuma, prosjeka ili sumiranja ovakve baze pokazale su vrlo dobre performanse [3].

Dokumente baze podataka su baze koje omogućavaju ubacivanje, dohvaćanje i manipulaciju polu-strukturiranih podataka. Ovakve baze najčešće koriste XML, JSON, BSON ili YAML formate. Podacima u bazi pristupa se preko HTTP protokola koristeći RESTful API. U ovakvim bazama podaci su spremljeni u dokumente koji ne moraju pratiti bilo kakvu shemu podataka. U dokumentnim bazama podataka ID dokumenta se dodjeljuje prilikom pristupanja bazi putem URL-a, no svejedno je dobra praksa dodijeliti ID dokumentu u bazi. Neke od baza koje koriste dokumenti pristup su **MongoDB**, CouchDB, Apache Cassandra i Redis [3].

U dokumentnoj bazi podaci su prikazani na sljedeći način (JSON format):

```
{

„DocumentID“: „DOC-1“,

„OIB_djelatnika“: 43940384294,

„Ime_djelatnika“ : „Ivana“,

„Prezime_djelatnika: „Grgurić“,

„Adresa_djelatnika“ : „Krešimirova 4“,

„Email_djelatnika“ : Ivana@gmail.com }
```

Zapis podataka bez sheme koristan je kod web aplikacija gdje se sprema sadržaj koji se može mijenjati tokom vremena. Kod nekih implementacija postoji mogućnost dohvaćanja ili mijenjanja dokumenta pojedinačno. Izvođenje upita nad ovakvim bazama podataka jednostavnije je od upita nad relacijskim bazama ili stupčastim bazama. Osim što je jednostavnije, puno je i brže jer se svi podaci nalaze u jednom dokumentu u usporedbi sa relacijskim bazama gdje se podaci nalaze u više tablica [3].

JSON format omogućava i ugnježđivanje podataka. Kada bi u prošlom zapisu htjeli dodati na primjer prošlo radno mjesto to bi napravili na sljedeći način:

```
{
„DocumentID“: „DOC-1“,
„OIB_djelatnika“: 43940384294,
„Ime_djelatnika“ : „Ivana“,
„Prezime_djelatnika: „Grgurić“,
„Adresa_djelatnika“ : „Krešimirova 4“,
„Email_djelatnika“ : „Ivana@gmail.com“
„Prošlo_radno_mjesto“: {
„naziv_tvrtke“: „Tvrtka1“,
„godina_pocetka“: 2021,
„godina_kraja“: 2022,}
}
```

Ključ-vrijednost baze podataka vrlo su slične dokumentnim bazama. Uz svaku spremljenu vrijednost nalazi se i ključ te vrijednosti. Kao i kod dokumentne baze ne postoji nikakva shema nad vrijednostima u bazi. Za razliku od dokumentnih baza u ključ-vrijednost bazama potrebno je definirati ključ svake vrijednosti. U dokumentnim bazama vrijednosti se mogu indeksirati i moguće je raditi upite nad njima, a u ključ-vrijednost bazama potrebno je znati ključ kako bi dohvatili vrijednost [3].

Graf baze su posebna vrsta nerelacijskih baza podataka. Graf baze sastoje se od čvorova i veza između čvorova. Čvorovi predstavljaju entitete a veze odnose između entiteta. Ovakve baze prikladne su za prikazivanje odnosa između ljudi, transportnih ruta između gradova ili topologija računalnih mreža [3].

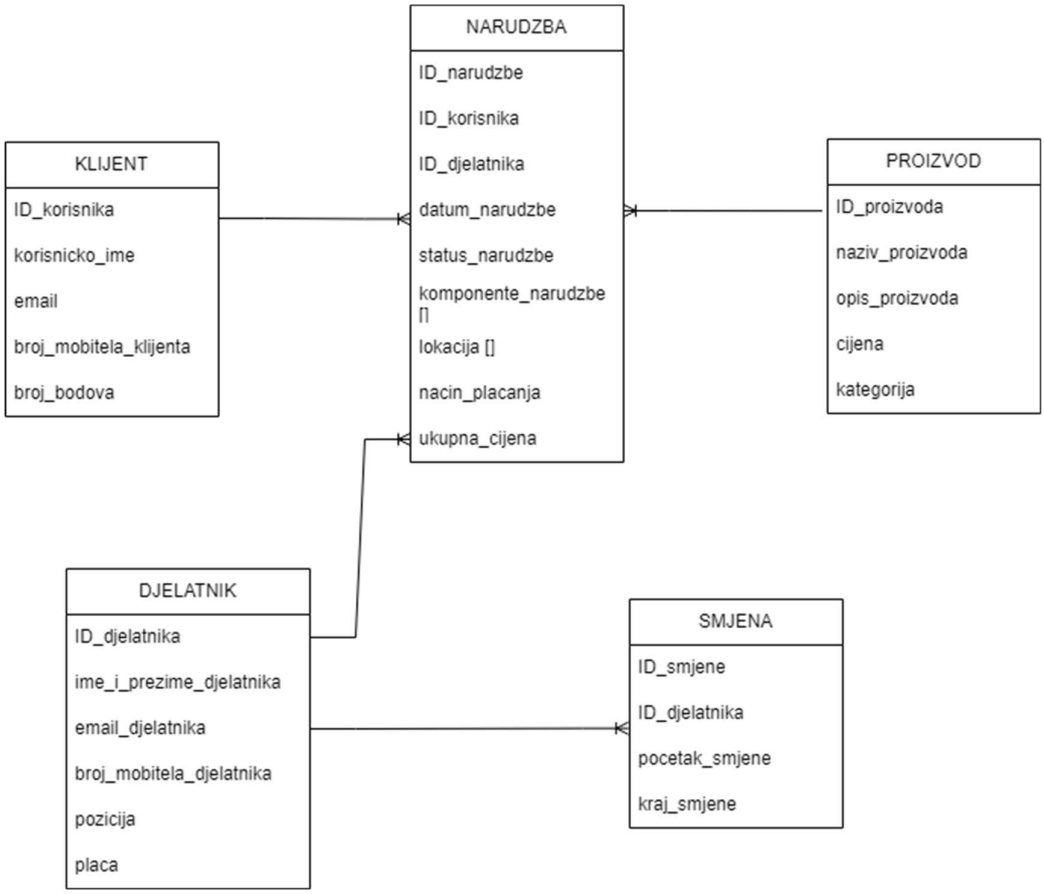
3. MODEL PODATAKA

Prikazani model podataka predstavlja kolekcije u bazi podataka za restoran. Kolekcija „Klijent“ sadrži informacije o korisnicima. Svaki klijent ima jedinstveni identifikator „id_korisnika“. Uz taj identifikator sadrži i podatke korisničkog imena, email-a, broja mobitela i broja bodova u aplikaciji.

Sljedeća kolekcija je „proizvodi“ koja sadrži podatke o svim proizvodima koje restoran nudi. Uz identifikator „id_proizvoda“ sadrži podatke naziva, opisa, cijene i kategorije proizvoda.

Slijedi kolekcija „narudžba“ koja sadrži informacije o narudžbama. Svaka narudžba ima svoj identifikator a dodatno sadrži atribut „id_korisnika“ i „id_djelatnika“ koji su reference na klijenta i djelatnika koji su sudjelovali u narudžbi. Uz navedene atribute sadrži i atribut „komponente_narudzbe“ koji je niz koji se sastoji od identifikatora proizvoda i lokaciju koja će sadržavati GPS koordinate na koju će biti potrebno dostaviti narudžbu.

Model sadrži i kolekciju „djelatnik“ koja sadrži podatke o svakom djelatniku. Uz tu kolekciju nalazi se i kolekcija „smjena“ koja sadrži identifikator smjene i djelatnika te atribute koji predstavljaju početak i kraj smjene.



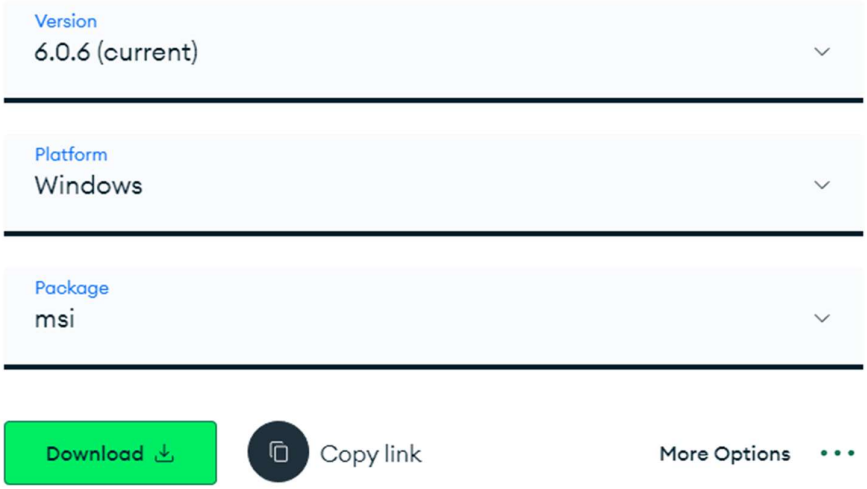
Slika 1 Model podataka

4. MONGODB

4.1. INSTALACIJA MONGODB-A

Rad u MongoDB-u započinjem instalacijom MongoDB Compass-a i MongoDB Shell-a. Instalacijom ova dva programa omogućiti ću učinkovito provođenje projekta. Oba dva programa su u potpunosti besplatna.

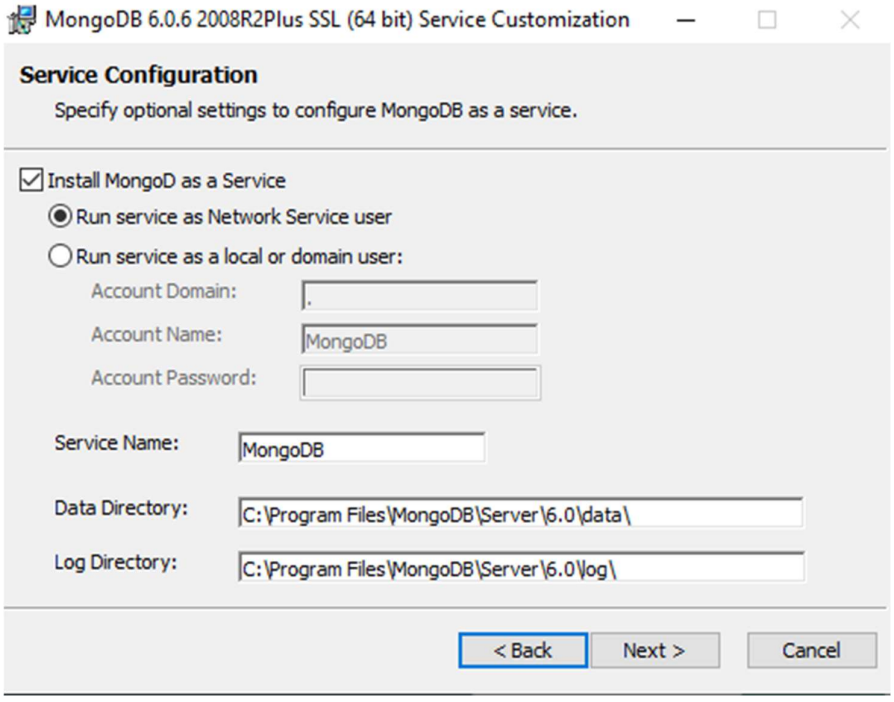
MongoDB Compass nalazi se na sljedećoj poveznici:
https://www.mongodb.com/try/download/community



Slika 2 Odabir verzije Compass-a

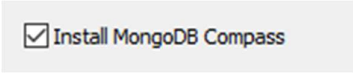
Trenutna verzija je 6.0.6 na platformi Windows. Kao instalacijski paket moguće je odabrati msi (Microsoft Software Installer) ili zip.

Pokretanjem instalacijskog paketa prvo je potrebno prihvatiti uvjete korištenja. Nakon toga moguće je odabrati način instalacije. Postoje opcije kompletne ili prilagođene instalacije.



Slika 3 Instalacijski prozor

Odabirom kompletne instalacije otvara se novi prozor u kojemu je potrebno označiti da ćemo instalirati MongoDB kao servis [5]. Moguće je promijeniti ime servisa te direktorije u koje se spremaju podaci i logovi baze.

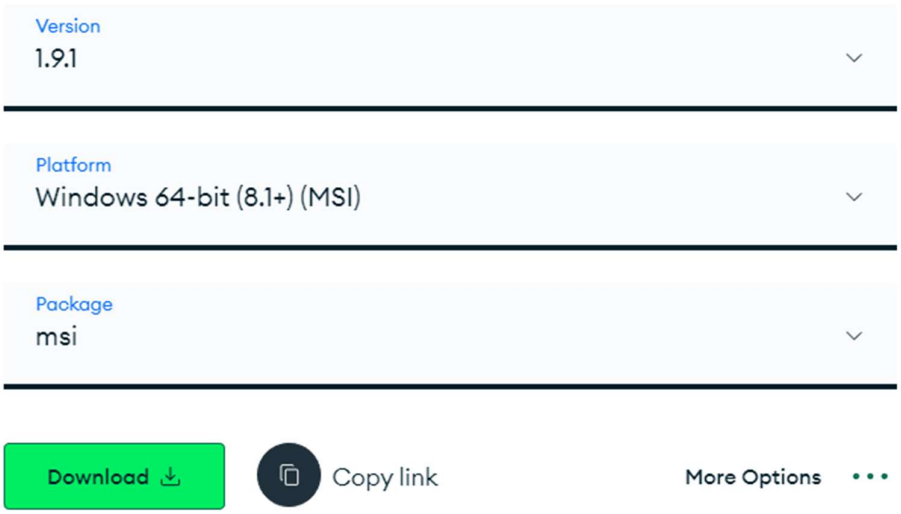


Slika 4 Opcija instalacije Compass-a

Na sljedećem prozoru potrebno je označiti da želimo instalirati MongoDB Compass. Nakon toga instalacijski paket će instalirati MongoDB lokalno na računalo.

Ukoliko je instalacija završila uspješno, automatski se otvara MongoDB Compass.

Sljedeći korak je instalacija MongoDB Shell-a preko kojega ćemo upisivati naredbe [6]. Shell je dostupan na sljedećoj poveznici: https://www.mongodb.com/try/download/shell



Slika 5 Odabir verzije Shell-a

Trenutna verzija je 1.9.1. MongoDB Shell također je moguće instalirati putem msi-a. Tokom instalacije ovog programa nije potrebno ništa posebno označavati.

Kako bi provjerili je li se Shell uspješno instalirao potrebno je otvoriti Windows PowerShell te upisati naredbu **`mongosh`**.

Ukoliko je sve uspješno instalirano, kao odgovor dobivamo sljedeće:

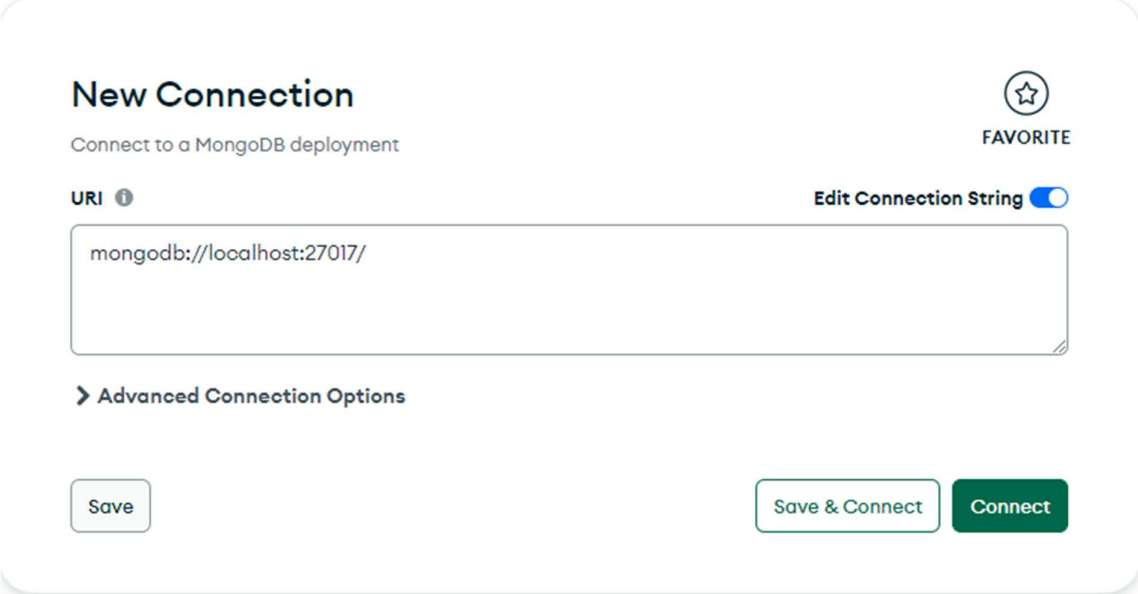
```
PS C:\Users\basic> mongosh
Current Mongosh Log ID: 647086d92900e031385e4a7f
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSe
lectionTimeoutMS=2000&appName=mongosh+1.9.1
Using MongoDB:          6.0.6
Using Mongosh:          1.9.1

For mongosh info see: https://docs.mongodb.com/mongodb-shell/


To help improve our products, anonymous usage data is collected and sent to Mongo
DB periodically (https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

------
   The server generated these startup warnings when booting
   2023-05-26T12:05:10.921+02:00: Access control is not enabled for the database.
 Read and write access to data and configuration is unrestricted
------

test>
```

Slika 6 Verzije MongoDB-a i Shell-a

PowerShell ispisuje trenutnu verziju MongoDB-a (6.0.6) te trenutnu verziju Shella (1.9.1).

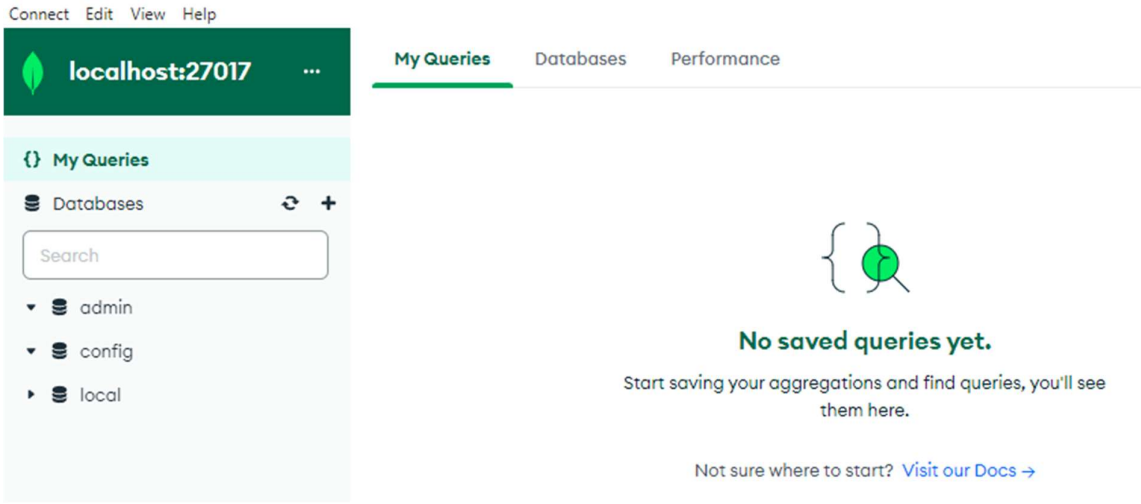
4.2. MONGODB COMPASS

Nakon pokretanja MongoDB Compass-a otvara se prozor u kojem je ponuđeno spajanje na bazu koja je lokalno pokrenuta.



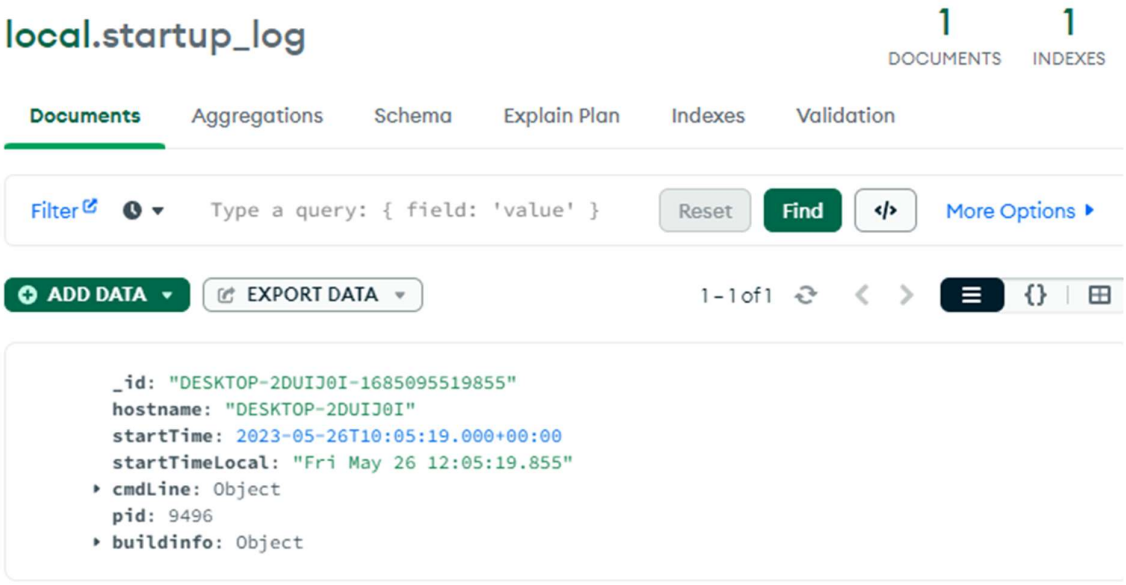
Slika 7 Spajanje na lokalnu bazu

Potrebno je kliknuti na gumb „Connect“ te time je uspješno izvršeno spajanje. Otvara se novi prozor:



Slika 8 Početni prozor Compass-a

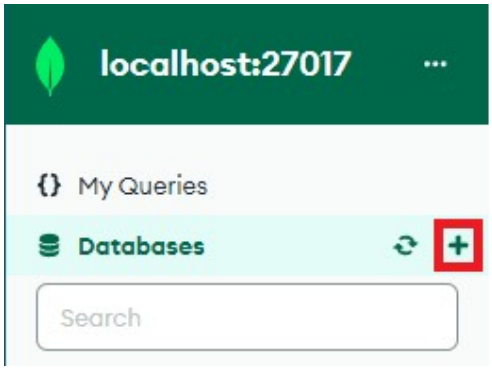
Sa lijeve strane nalaze se baze podataka *admin*, *config* i *local*. Ove baze su unaprijed napravljene od MongoDB-a. U bazi podataka *local* spremaju se instance pokretanja MongoDB-a [7]. Otvaranjem startup_loga moguće je vidjeti prvo pokretanje baze:



Slika 9 Startup log

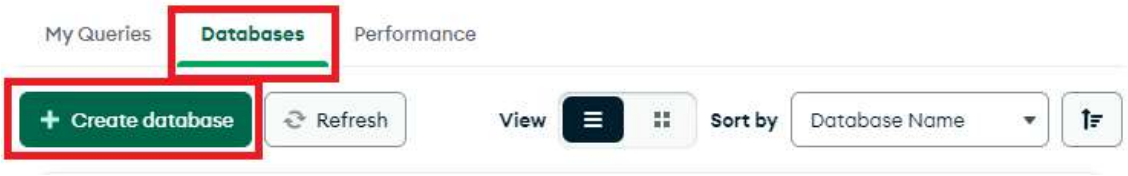
Svoju bazu podataka putem sučelja možemo dodati na 2 načina [8]:

1. Klikom na znak „+“ na lijevom izborniku



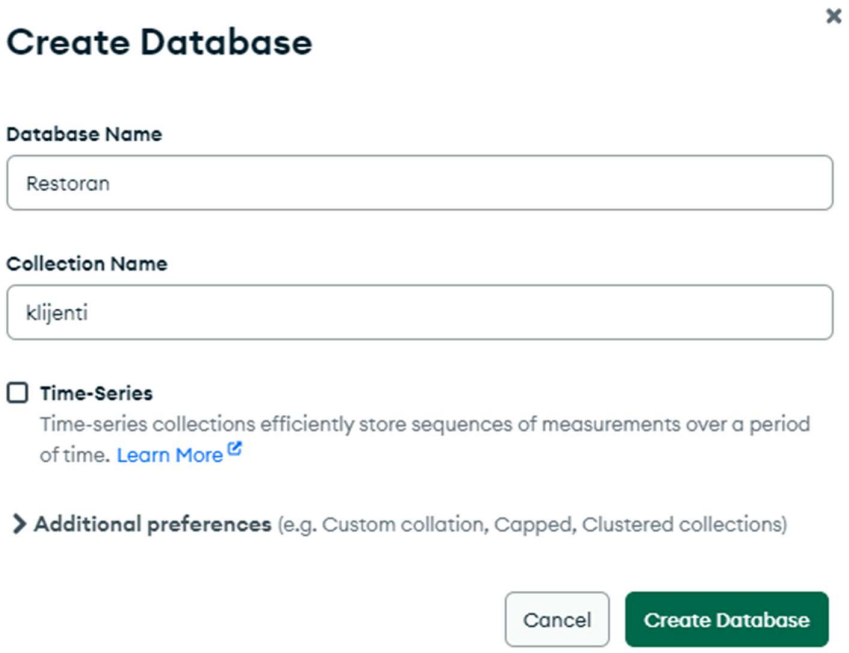
Slika 10 Prvi način stvaranja baze

2. Odabirom izbornika „Databases“ te klikom na gumb „ + Create Database“



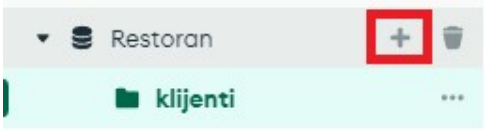
Slika 11 Drugi način stvaranja baze

Otvara se novi prozor u kojemu je potrebno unijeti naziv baze i naziv prve kolekcije.



Slika 12 Unos imena i kolekcije

Klikom na gumb „Create Database“ stvaramo novu bazu. Ukoliko želimo dodati još kolekcija, tada je potrebno pritisnuti znak „+“ pored baze „Restoran“.



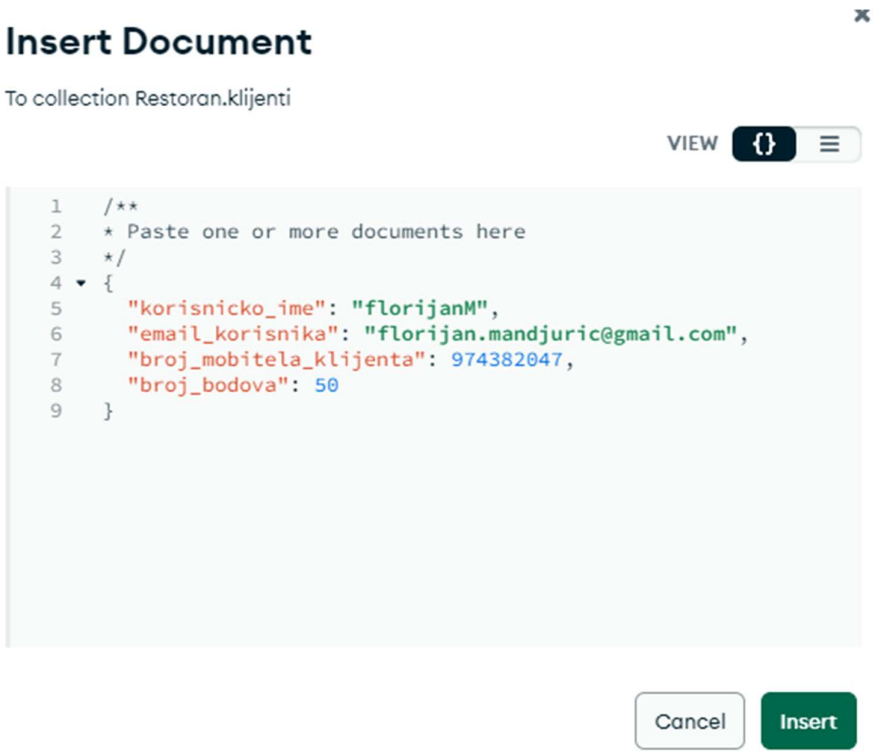
Slika 13 Dodavanje nove kolekcije

Otvara se sličan prozor kao i kod kreiranja baze no ovaj put je potrebno unijeti samo ime kolekcije.

Kolekcije su trenutno prazne tj. nemaju dokumenata. Dokument možemo dodati klikom na gumb „ADD DATA“ [9].

Slika 14 Gumb za dodavanje novog dokumenta

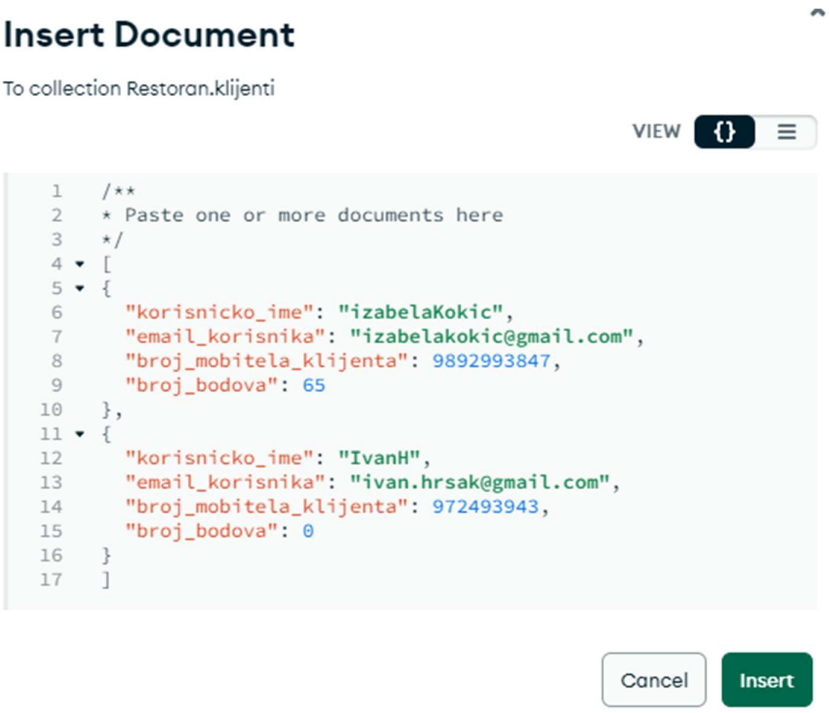
Otvara se prozor u kojemu je potrebno unijeti dokument u JSON obliku.



Slika 15 Dodavanje jednog dokumenta

Na slici 15 prikazan je unos podataka za jednog klijenta. Nema potrebe za unosom ID-a korisnika zato jer taj podatak MongoDB dodaje automatski prilikom unosa. Ukoliko bi izbrisali taj podatak, on bi se svejedno prikazivao prilikom pregleda podataka.

Moguće je unijeti i više dokumenata odjednom, to se radi na sljedeći način:



Slika 16 Dodavanje više dokumenata

Unos je potrebno staviti u uglastu zagradu jer je to zapravo niz koji se sastoji od dokumenata.

```
[ {prvi element}, {drugi element} , … ]
```

Iako kod unosa elementa nije određen ID dokumenta, MongoDB je napravio vlastiti ID što možemo vidjeti na sljedećoj slici.

Slika 17 ObjectId

Prelaskom kursora preko dokumenta dobivamo četiri opcije:

Slika 18 Opcije uređivanja dokumenta

1. Izmjena dokumenta – moguće je promijeniti vrijednost atributa
2. Kopiranje dokumenta – kopira trenutni dokument u JSON formatu
3. Kloniranje dokumenta – otvara unos kojim ponovno unosimo isti dokument
4. Brisanje dokumenta – briše dokument iz kolekcije

Klikom na prvu opciju mijenja se prozor te je moguće promijeniti naziv, vrijednost i tip atributa.

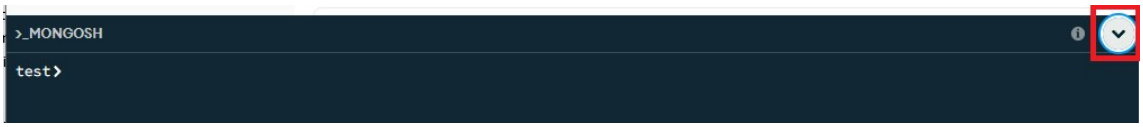
Slika 19 Tipovi atributa

Iz slike je moguće zaključiti da MongoDB automatski prepoznaje vrijednost atributa. Pod poljem „broj_mobitela_klijenta“ postavljena je vrijednost Int32.

4.4. MONGODB SHELL

MongoDB Shell-u moguće je pristupiti na više načina. U poglavlju 4.2. prikazana je instalacija MongoDB Shella, te pomoću komandnog sučelja (PowerShell) moguće je spojiti se na MongoDB server te unositi komande.

No MongoDB Shell-u moguće je pristupiti i putem MongoDB Compass-a. On se nalazi u donjem dijelu prozora. Klikom na strelicu u donjem desnom kutu otvaramo komandnu liniju.



Slika 20 Shell u Compass-u

U nastavku rada koristiti će se Shell preko Windows PowerShell-a.

Nakon unosa komande „mongosh“ Shell je spojen na bazu „test“. Dostupne baze podataka moguće je pregledati naredbom „show databases“ (također funkcionira i „show dbs“) [10].

```
test> show databases
Restoran    72.00 KiB
admin       40.00 KiB
config     108.00 KiB
local       72.00 KiB
test>
```

Slika 21 Dostupne baze

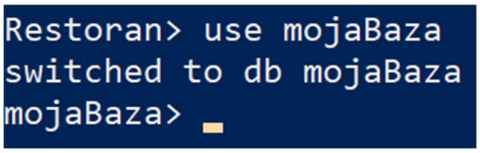
Na bazu „Restoran“ moguće je prebaciti se s sljedećom naredbom:

```
test> use Restoran
switched to db Restoran
Restoran>
```

Slika 22 Prebacivanje baze

Potrebno je obratiti pozornost prilikom unosa imena, jer MongoDB Shell funkcionira tako da je moguće koristiti bilo koje ime baze, čak i ako baza još nije kreirana.

To znači da ukoliko bi htjeli koristiti bazu „mojaBaza“, tada je potrebno samo unijeti sljedeće

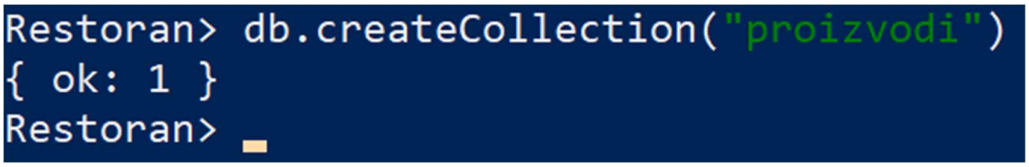


Slika 23 Stvaranje baze

Iako se prebacivanje u drugu bazu uspješno dogodilo, MongoDB Shell neće kreirati bazu sve dok ne unesemo prve podatke.

Stvaranje nove kolekcije i unos jednog dokumenata u nju putem MongoShell-a funkcionira na sljedeći način:

1. Opcionalno – stvaranje nove kolekcije



Slika 24 Stvaranje kolekcije putem Shell-a

„ok:1“ u odgovoru znači da je stvaranje kolekcije prošlo uspješno. Također provjeru je moguće napraviti otvaranjem Compass-a. Ukoliko je odgovor ok:0 to znači da je došlo do greške pri stvaranju kolekcije.

Ovaj korak je opcionalan jer je moguće izvesti korak 2 te će MongoDB stvoriti novu kolekciju koja sadrži dokument koji je unesen.

2. Unos jednog dokumenta u kolekciju

```
db.proizvodi.insertOne({_id: "1", naziv_proizvoda: "Hamburger",
opis_proizvoda: "Govedja pljeskavica sa povrcem i umakom po zelji",
cijena: 3, kategorija: "glavno jelo"})
```

„acknowledged: true“ u odgovoru znači da je dokument unesen uspješno. Prilikom unosa ovog podataka postavljen je vlastiti ID. Ukoliko želimo izbjeći automatsko postavljanje ID-a od strane MongoDB-a potrebno je polje nazvati „_id“. Atribut „slika_proizvoda“ je string slike koji je pretvoren u Base64 format.

Analogno unosu jednog dokumenta moguće je unijeti i više dokumenata odjednom koristeći insertMany() naredbu:

```
db.proizvodi.insertMany([{_id: "2", naziv_proizvoda: "Cheeseburger",
opis_proizvoda: "Govedja pljeskavica sa sirom i umakom po zelji",
cijena: 3.5, kategorija: "glavno jelo"}, {_id: "3", naziv_proizvoda:
"Chickenburger", opis_proizvoda: "Pileci file sa povrcem i umakom po
zelji", cijena: 3.75, kategorija: "glavno jelo"} ])
```

4.5. JEDNOSTAVNI UPITI

Izvođenje upita u MongoDB Compass-u izvodi se upisivanjem filtera u označenu tražilicu [11]:

Slika 25 Unos upita u Compass-u

Kako bi izvršili neki jednostavniji upit kao npr. pronaći sve proizvode koji su prilozi tada je potrebno unijeti sljedeći filter:

```
{ kategorija: 'prilog'}
```

Nakon pritiska na gumb „Find“ prikazuju se rezultati upita:

Slika 26 Rezultati upita u Compass-u

Osim pretraživanja po jednom podatku, moguće je izvesti i pretraživanje po više podataka. Kako bi npr. pronašli sva glavna jela koja koštaju 3 eura potrebno je vrijednosti odvojiti zarezom na sljedeći način:

```
{kategorija: "glavno jelo", cijena: 3}
```

Analogno pretraživanju u Compass-u, istu stvar je moguće napraviti i u Shell-u. Nakon odabira baze podataka koju koristimo (Sveučilište) unosimo sljedeću komandu [12]:

```
db.proizvodi.find({kategorija: "prilog"})
```

Shell daje isti odgovor kao i Compass:

```
Restoran> db.proizvodi.find({kategorija: "prilog"})
[
  {
    _id: '200',
    naziv_proizvoda: 'Pomfri',
    opis_proizvoda: 'Pomfri s umakom po zelji',
    cijena: 2,
    kategorija: 'prilog'
  },
  {
    _id: '201',
    naziv_proizvoda: 'Kolutici luka',
    opis_proizvoda: 'Frigani kolutici luka s umakom',
    cijena: 2,
    kategorija: 'prilog'
  }
]
Restoran>
```

Slika 27 Rezultati upita u Shell-u

Kada bi izvođenjem upita htjeli dobiti samo neka polja, npr. želimo prikazivati sve podatke osim ID-a jela i kategorije. Potrebno je kliknuti na opciju „More Options“:

Slika 28 Dodatne opcije

Nakon toga otvara se više opcija prilikom upisa upita:

Slika 29 Prikaz dodatnih opcija

Pod poljem „Project“ potrebno je upisati sljedeće:

```
{_id: 0, kategorija: 0}
```

Oznaka „0“ predstavlja da se polja neće prikazivati u upitu. Inverzno tome moguće je upisati „1“ za sva polja koja želimo prikazati.

Na sličan način potrebno je upisati upit u Shell:

```
db.proizvodi.find({ kategorija: "glavno jelo" }, { _id: 0,
kategorija: 0, slika_proizvoda: 0 })
```

```
Restoran> db.proizvodi.find({kategorija: "glavno jelo"}, { _id: 0, kategorija: 0})
[
  {
    naziv_proizvoda: 'Hamburger',
    opis_proizvoda: 'Govedja pljeskavica sa povrcem i umakom po zelji',
    cijena: 3
  },
  {
    naziv_proizvoda: 'Cheeseburger',
    opis_proizvoda: 'Govedja pljeskavica sa sirom i umakom po zelji',
    cijena: 3.5
  },
  {
    naziv_proizvoda: 'Chickenburger',
    opis_proizvoda: 'Pileci file sa povrcem i umakom po zelji',
    cijena: 3.75
  }
]
```

Slika 30 Pretraga glavnih jela

Kada bi cilj upita bio dobiti jedan dokument, tada je moguće koristiti naredbu findOne umjesto find.

Ova naredba je korisna pri izvedbi upita gdje se očekuje jedan rezultat, npr. pretraživanje po ID-u:

```
db.proizvodi.findOne({_id: "1"})
```

```
Restoran> db.proizvodi.findOne({_id: "1"})
{
  _id: '1',
  naziv_proizvoda: 'Hamburger',
  opis_proizvoda: 'Govedja pljeskavica sa povrcem i umakom po zelji',
  cijena: 3,
  kategorija: 'glavno jelo'
}
```

Slika 31 Pretraga po ID-u

Osim dohvaćanja sadržaja dokumenta moguće je dohvatiti i ukupan broj dokumenata u upitu. Tada je potrebno na kraj upita staviti naredbu count().

Dohvaćanje ukupnog broja klijenata izvodi se sljedećom naredbom:

```
db.klijenti.find().count()
```

```
Restoran> db.klijenti.find().count()
13
Restoran>
```

Slika 32 Pretraga broja klijenata

Rezultate upita moguće je i sortirati. Moguće je sortirati stringove i integere.

Sortiranje je moguće izvesti naredbom sort(). Unutar naredbe potrebno je unijeti po kojoj varijabli želimo sortirati. Nakon varijable potrebno je staviti vrijednost 1 (uzlazno sortiranje) ili -1 (silazno sortiranje).

Kada bi izvodili upit kao npr. sortiranje proizvoda od najskupljeg prema najjeftinijem tada bi upit izgledao:

```
db.proizvodi.find().sort({cijena: -1})
```

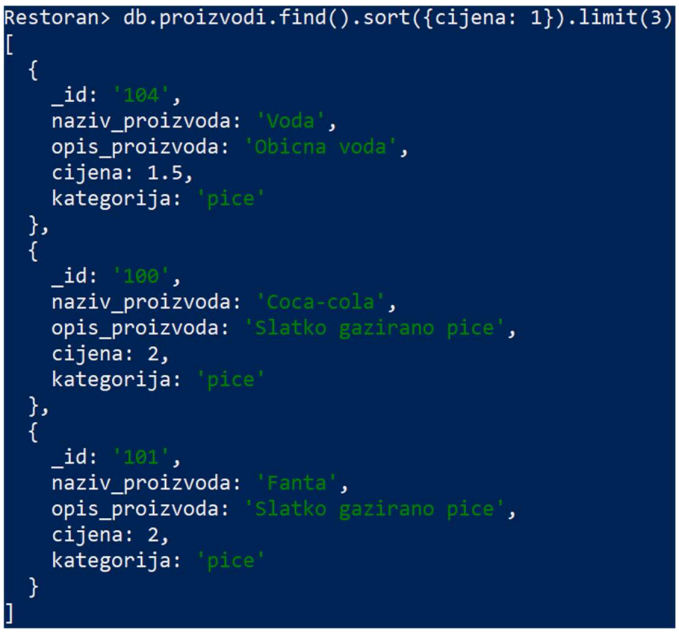
```
Restoran> db.proizvodi.find().sort({cijena: -1})
[
  {
    _id: '3',
    naziv_proizvoda: 'Chickenburger',
    opis_proizvoda: 'Pileci file sa povrcem i umakom po zelji',
    cijena: 3.75,
    kategorija: 'glavno jelo'
  },
  {
    _id: '2',
    naziv_proizvoda: 'Cheeseburger',
    opis_proizvoda: 'Govedja pljeskavica sa sirom i umakom po zelji',
    cijena: 3.5,
    kategorija: 'glavno jelo'
  },
  {
    _id: '1',
    naziv_proizvoda: 'Hamburger',
    opis_proizvoda: 'Govedja pljeskavica sa povrcem i umakom po zelji',
    cijena: 3,
    kategorija: 'glavno jelo'
  },
  {
    _id: '103',
    naziv_proizvoda: 'Monster',
    opis_proizvoda: 'Slatko energetsko pice',
    cijena: 2.5,
    kategorija: 'pice'
  },
```

Slika 33 Pretraga po padajućoj cijeni

Ukoliko je cilj upita pronaći 3 najjeftinija proizvoda tada je moguće koristiti naredbu `limit()`.

Upit za 3 najjeftinija proizvoda:

```
db.proizvodi.find().sort({cijena: 1}).limit(3)
```



Slika 34 Pretraga po rastućoj cijeni

4.6. OPERATORI

U jednostavnim upitima dani su primjeri gdje se traži konkretna vrijednost (npr. glavno jelo kojemu je cijena 3 eura). Kako bi izveli upite koji ne traže konkretne vrijednosti potrebno je koristiti operatore.

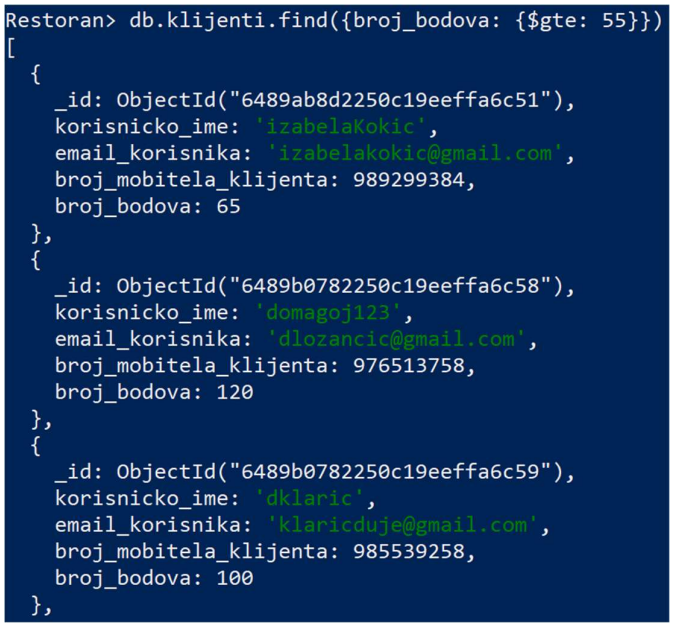
Operatori se označavaju znakom dolara ($). U MongoDB-u postoji mnogo vrsta operatora. Osnovni operatori su operatori usporedbe i logički operatori [13].

Operatorima usporedbe moguće je tražiti npr. sve proizvode koji imaju vrijednost veću ili manju od zadane vrijednosti u upitu.

Kada bi htjeli naći sve klijente koji imaju 55 ili više bodova upit bi izgledao:

```
db.klijenti.find({broj_bodova: {$gte: 55}})
```

Pri korištenju operatora potrebno je otvoriti novu vitičastu zagradu. U ovom primjeru korišten je operator „$gte“ koji predstavlja operator veći ili jednak (>=)



Slika 35 Pretraga svih klijenata sa brojem bodova većim od 55

Na isti način funkcioniraju i sljedeći operatori: $gt (traženje isključivo veće vrijednosti), $lt (traženje manje vrijednosti), $lte (traženje jednake ili manje vrijednosti).

Kada bi htjeli pronaći sve klijente koji imaju 50 ili 100 bodova tada je potrebno koristiti operator $or. Ovaj operator predstavlja logičko ILI što znači da će vratiti dokument u upitu ukoliko je zadovoljen barem jedan od uvjeta.

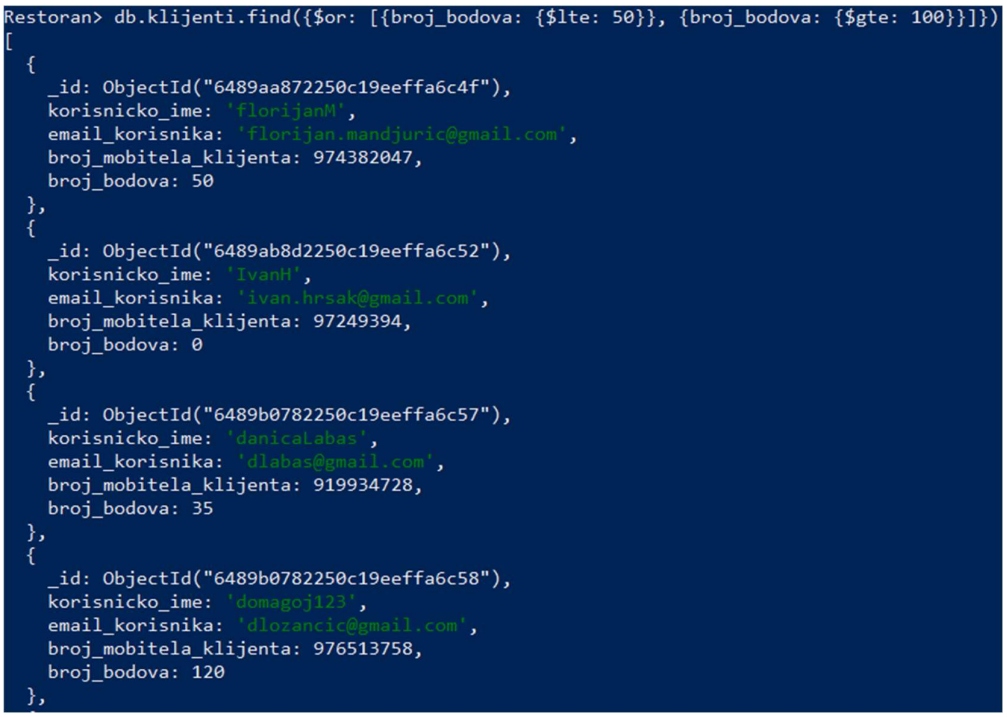
Upit za pretragu klijenata sa 50 ili 100 bodova izgleda ovako:

```
db.klijenti.find({$or: [{broj_bodova: 50}, {broj_bodova:100}]})
```

Moguće je i kombinirati više operatora u jednom upitu. Kada bi npr. htjeli pronaći sve klijente koji imaju manje od 50 ili više od 100 bodova tada je potrebno kombinirati različite operatore.

Takav upit izgleda ovako:

```
db.klijenti.find({$or: [{broj_bodova: {$lte: 50}}, {broj_bodova:
{$gte: 100}}]})
```

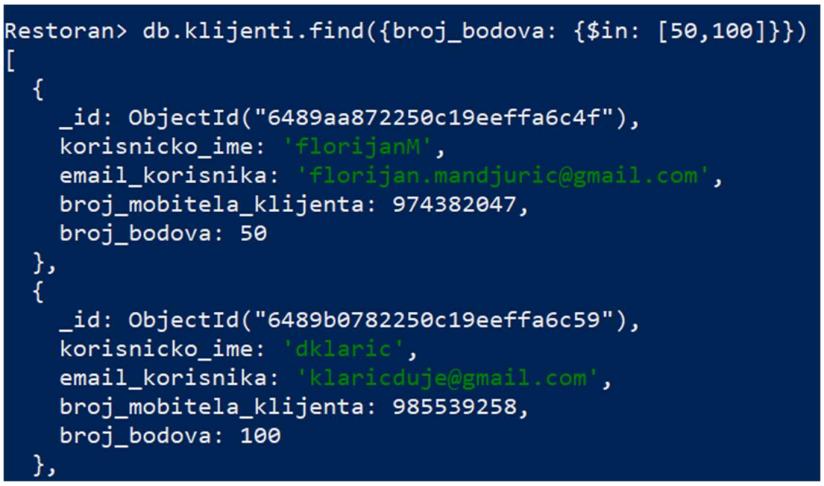


Slika 36 Pretraga korištenjem operatora $or

Upit gdje su se pretraživali klijenti s 50 ili 100 bodova može se izvesti i na bolji način. Moguće ga je optimizirati korištenjem operatora $in. Kod ovog operatora potrebno je upisati sve vrijednosti koje tražimo u polje na sljedeći način:

```
db.klijenti.find({broj_bodova: {$in: [50,100]}})
```

Suprotno operatoru $in postoji operator $nin (not in) koji ima inverznu funkciju što znači da traži sve vrijednosti koje nisu u polju.



Slika 37 Pretraga korištenjem operatora $in

5. POVEZIVANJE DOKUMENATA

MongoDB najoptimalnije radi s denormaliziranim podatkovnim modelima. No u nekim slučajevima ima više smisla pohraniti informacije u različite dokumente. U podatkovnom modelu restorana kolekcije „Smjena“ i „Narudzba“ sadržavati će podatke iz drugih kolekcija. Postoje 2 načina povezivanja dokumenata [14]:

- Korištenjem manualnih referenci – spremanje ID-a jednog dokumenta kao vrijednost u drugi dokument. Aplikacija će tada raditi 2 upita kako bi dohvatila relevantne podatke.
- Korištenjem dokumentnih referenci – to su reference jednog dokumenta na drugi. Potrebno je koristiti ID prvog dokumenta, naziv kolekcije i opcionalno naziv baze podataka.

Kolekcija „smjena“ sadrži „id_djelatnika“ koji odrađuje smjenu. Slijedi kreiranje dokumenta korištenjem manualnih referenci:

```
db.smjene.insertOne({ id_djelatnika: "64493024917",  pocetak_smjene: ISODate("2023-08-01T07:00:00Z"),  kraj_smjene: ISODate("2023-08-01T15:00:00Z") })
```

U kolekciji smjene pušteno je automatski dodjeljivanje ID-a, pod id_djelatnika potrebno je samo upisati isti ID. Pod poljima početak i kraj smjene korišten je format ISODate.

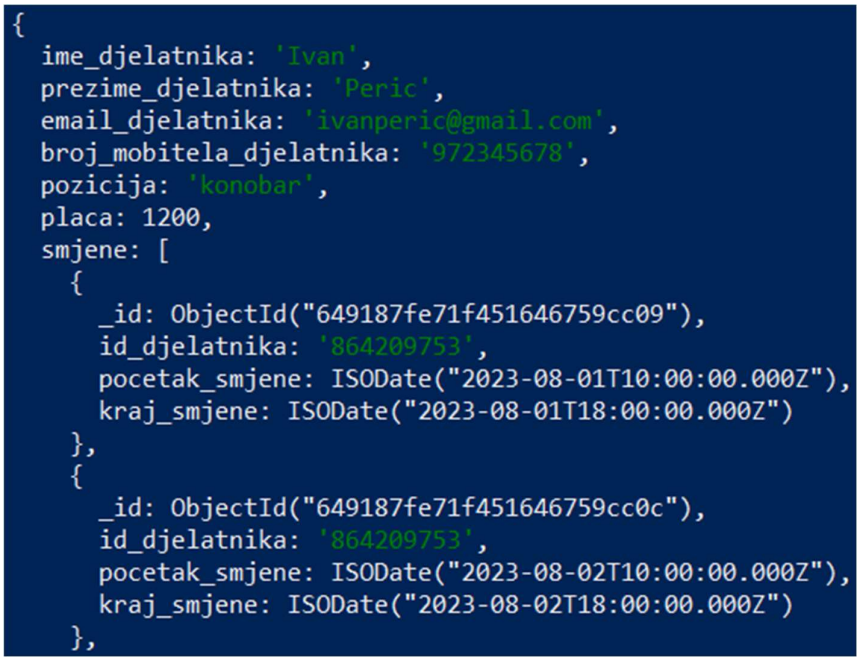
Format ISODate redom sadrži: „godina – mjesec – dan – T (početak oznake za vrijeme) – sat (u formatu 0-24) – minute – sekunde – Z (vremenska zona)“

Kao i kod relacijskih baza podataka, moguće je izvoditi upite nad dvije ili više kolekcija. U ovom slučaju korisno bi bilo dobiti ispis svih radnika zajedno sa njihovim smjenama [15].

```
db.djelatnici.aggregate([

{$lookup: { from: "smjene", localField: "_id", foreignField: "id_djelatnika", as: "smjene"}},

{$project: {_id: 0, id_djelatnika: "$_id", ime_djelatnika: 1, prezime_djelatnika: 1, email_djelatnika: 1, broj_mobitela_djelatnika: 1, pozicija: 1, placa: 1, smjene: 1}} ])
```

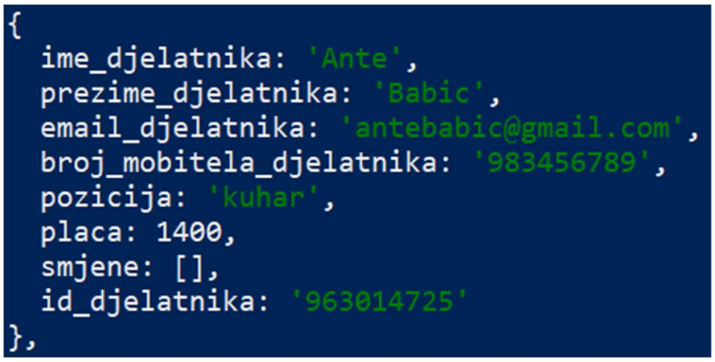
Upit započinje naredbom aggregate. Ovom naredbom moguće je dohvatiti vrijednosti iz više dokumenata i vratiti rezultate. Također, moguće je raditi i vlastite izračune.

Pod $lookup potrebno je definiriati kako će se spajati kolekcije. from označava iz koje kolekcije se spajaju vrijednosti. localField označava polje u lokalnoj kolekciji (u ovom slučaju to je kolekcija „djelatnici“). foreignField označava polje u kolekciji „smjene“. Pod poljem as definira se naziv spojenog polja. U ovom slučaju to će biti smjene te će rezultat izgledati:



Slika 38 Rezultat upita djelatnika i smjena

Ukoliko radnik nema upisanih smjena, tada će rezultat biti prazno polje.



Slika 39 Radnik bez smjena

Kolekcija „narudzbe“ sadrži id klijenta koji je naručio hranu te sve komponente koje se nalaze u narudžbi. Prilikom unosa stvaramo nove polje koje sadrži sve ID-eve proizvoda:

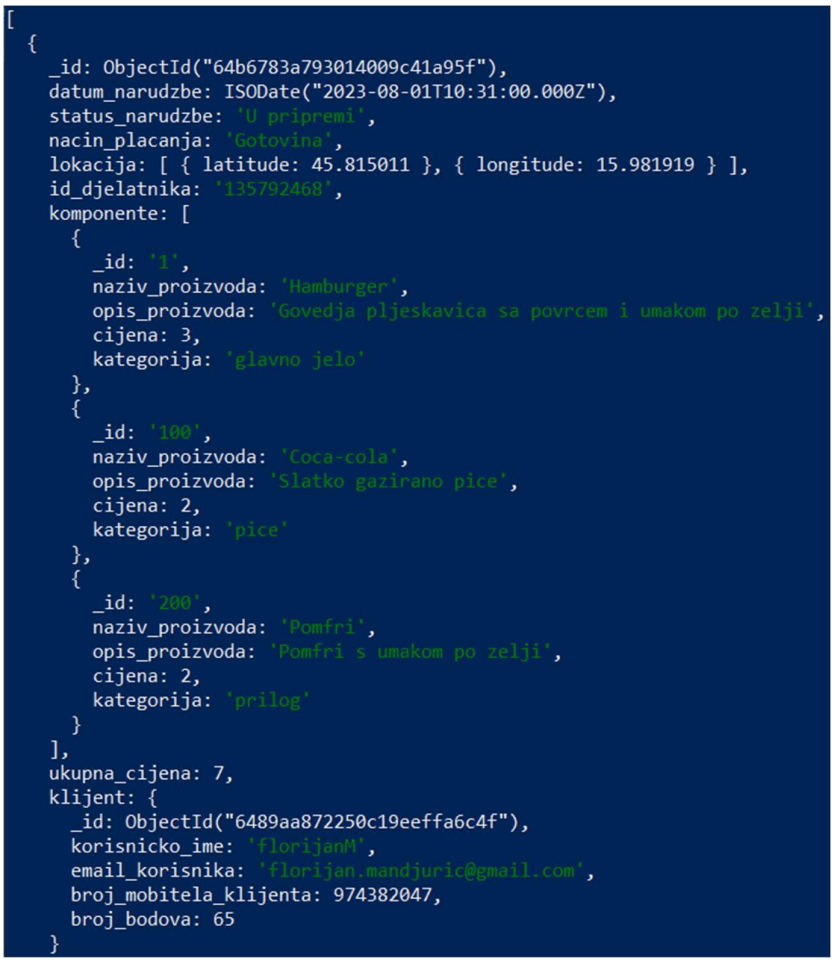
```
db.narudzbe.insertOne({

id_klijenta: ObjectId("6489aa872250c19eeffa6c4f"), id_djelatnika:
„135792468“, datum_narudzbe: ISODate("2023-08-01T10:31:00Z"),
status_narudzbe: "U pripremi", nacin_placanja:
"Gotovina",komponente_narudzbe: [ { id_komponente: "1" },
{ id_komponente: "100" }, { id_komponente: "200" }], lokacija:
[ {latitude: 45.815011}, {longitude: 15.981919}] });
```

U stvarnom okruženju možda će biti potrebno izračunati ukupnu cijenu narudžbe. To se također radi agregacijskim upitom kojim je moguće dodati nova polja.

Slijedi upit za kolekciju narudžba koji radi 2 spajanja. Prvo spajanje biti će sa kolekcijom „klijenti“ gdje dohvaćamo podatke o klijentu koji je izvršio narudžbu. Drugo spajanje je sa kolekcijom „proizvodi“ gdje dohvaćamo podatke o svakom proizvodu koji je u narudžbi. Zatim slijedi izračun vlastite vrijednosti, u ovom slučaju to će biti ukupna cijena narudžbe.

```
db.narudzbe.aggregate([ { $lookup: { from: "proizvodi", localField: "komponente_narudzbe.id_komponente", foreignField: "_id", as: "komponente" } }, { $lookup: { from: "klijenti", localField: "id_klijenta", foreignField: "_id", as: "klijent" } }, { $addFields: { ukupna_cijena: { $sum: "$komponente.cijena" } } }, { $project: { _id: 1, datum_narudzbe: 1, status_narudzbe: 1, nacin_placanja: 1, ukupna_cijena: 1, komponente: 1, lokacija: 1, id_djelatnika: 1, klijent: { $arrayElemAt: ["$klijent", 0] } } }]);
```

Rezultat upita je vrlo kompleksan:



Slika 40 Rezultat upita narudžbe

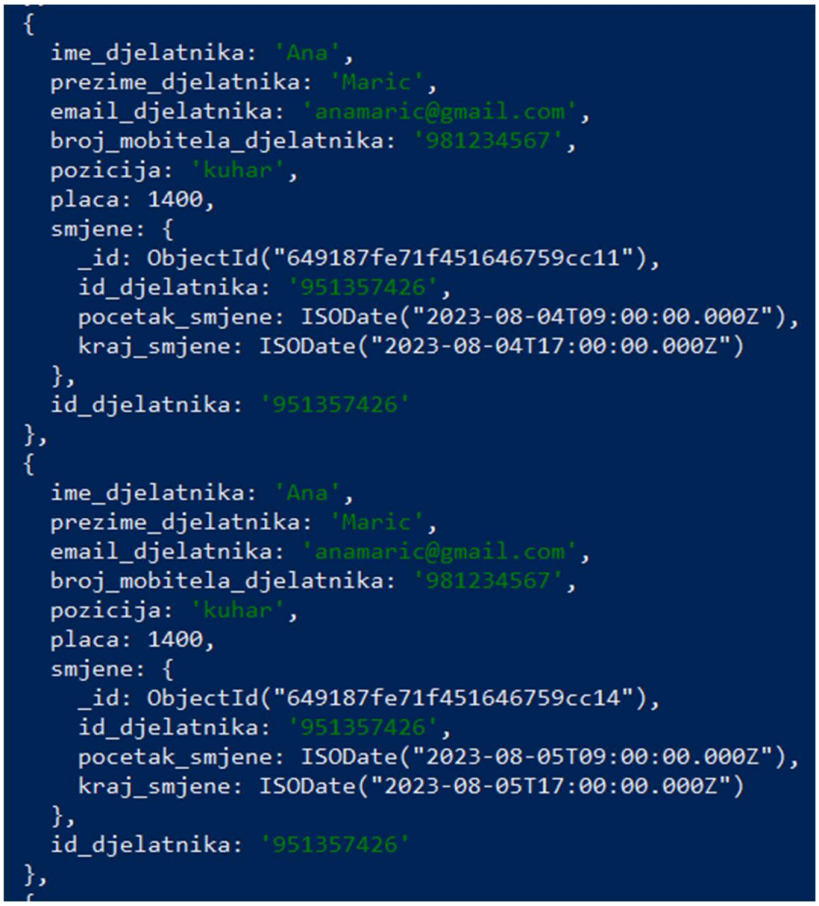
U složenim upitima moguće je koristiti operatore. Kako bi pronašli npr. sve smjene za djelatnike kojima je plaća veća od 1100 potrebno je dodati $match u upitu na sljedeći način:

```
db.djelatnici.aggregate([ { $lookup: { from: "smjene", localField: "_id", foreignField: "id_djelatnika", as: "smjene" } }, { $match: { pozicija: "konobar", placa: { $gt: 1100 } } }, { $project: { _id: 0, id_djelatnika: "$_id", ime_djelatnika: 1, prezime_djelatnika: 1, email_djelatnika: 1, broj_mobitela_djelatnika: 1, pozicija: 1, placa: 1, smjene: 1 } }]);
```

Operatore je moguće koristiti i nad datumima. Ukoliko bi htjeli dobiti upit koji pokazuje sve smjene pojedinačno za svakog djelatnika nakon 3.8. to bi napravili na sljedeći način:

```
db.djelatnici.aggregate([{ $lookup: { from: "smjene", localField: "_id", foreignField: "id_djelatnika", as: "smjene" } }, { $unwind: "$smjene" }, { $match: { "smjene.pocetak_smjene": { $gt: ISODate("2023-08-03T00:00:00.000Z") } } }, { $project: { _id: 0, id_djelatnika: "$_id", ime_djelatnika: 1, prezime_djelatnika: 1, email_djelatnika: 1, broj_mobitela_djelatnika: 1, pozicija: 1, placa: 1, smjene: 1 } }]);
```

U ovom upitu možemo primijetiti dio $unwind. Ovaj dio razdvaja svaku smjenu kako bismo mogli primijeniti operator usporedbe datuma. Kada ne bi razdvojili smjene tada bi dobili i smjene prije 3.8. ako radnik ima smjenu nakon 3.8.



Slika 41 Prikaz odvojenih smjena nakon 3.8.

6. BRISANJE I AŽURIRANJE DOKUMENATA

U poglavlju 4.3. na slici 18 prikazano je brisanje jednog dokumenta u Compass-u. No dokumente je moguće i brisati putem Shell-a. U nekim situacijama možda će biti potrebno izbrisati i više dokumenata odjednom. Brisanje jednog dokumenta putem Shell-a vrši se sljedećom naredbom [16]:

```
db.klijenti.deleteOne({_id: ObjectId("6489b0782250c19eeffa6c5b")})
```

Brisanje se izvršilo putem ID-a jer je to unikatno polje svakog dokumenta. U shell-u dobivamo i odgovor je li brisanje uspješno izvršeno i koliko je dokumenata izbrisano.

```
Restoran> db.klijenti.deleteOne({_id: ObjectId("6489b0782250c19eeffa6c5b")})
{ acknowledged: true, deletedCount: 1 }
Restoran>
```

Slika 42 Brisanje jednog dokumenta

Moguće je koristiti i operatore pri brisanju podataka, npr. ukoliko želimo obrisati sve klijente koji imaju više od 100 bodova to je moguće napraviti na sljedeći način:

```
db.klijenti.deleteMany({broj_bodova: {$gt: 100}})
```

Kao odgovor dobivamo potvrdu da je obrisano 2 klijenta:

```
Restoran> db.klijenti.deleteMany({broj_bodova: {$gt: 100}})
{ acknowledged: true, deletedCount: 2 }
```

Slika 43 Brisanje više dokumenata

Ažuriranje dokumenata također se može izvršiti nad jednim ili nad više dokumenata odjednom. Kada bi htjeli određenom voditelju podignuti plaću sa 1500 na 1650 to bi napravili sljedećom naredbom [17]:

```
db.djelatnici.updateOne({_id: "64493024917"}, {$set: {placa: 1650}})
```

Korištenjem operatora $set moguće je promijeniti vrijednost postojećeg polja ili dodati nova polja.

```
Restoran> db.djelatnici.updateOne({_id: "64493024917"}, {$set: {placa: 1650}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Slika 44 Ažuriranje jednog dokumenta

Alternativno operatoru $set moguće je koristiti operator $inc koji predstavlja inkrementiranje polja za određenu vrijednost. Ukoliko bi htjeli npr. povećati broj bodova svih klijenata koji imaju manje od 100 bodova za 15 to bi napravili na sljedeći način:

```
db.klijenti.updateMany({broj_bodova: {$lt: 100}}, {$inc:
{broj_bodova: 15}})
```

Kako bi ažurirali više polja odjednom potrebno je koristiti naredbu updateMany. U odgovoru shell-a moguće je vidjeti da je ažuriranje uspješno napravljeno za 9 dokumenata.

```
Restoran> db.klijenti.updateMany({broj_bodova: {$lt: 100}}, {$inc: {broj_bodova: 15}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 9,
  modifiedCount: 9,
  upsertedCount: 0
}
Restoran>
```

Slika 45 Ažuriranje više dokumenata

Moguće je ažurirati i elemente u poljima. Kada bi htjeli npr. izbrisati jednu komponentu narudžbe (npr. klijent više ne želi pomfrit u svojoj narudžbi) tada je to moguće napraviti korištenjem operatora $pull na sljedeći način [18]:

```
db.narudzbe.updateOne( { _id: ObjectId("649ab1a30e5f58f7f9864d12") },
{ $pull: { "komponente_narudzbe": { id_komponente: "200" } } } )
```

Pregledom narudžbe u Compass-u moguće je utvrditi da je pomfrit stvarno maknut iz narudžbe:

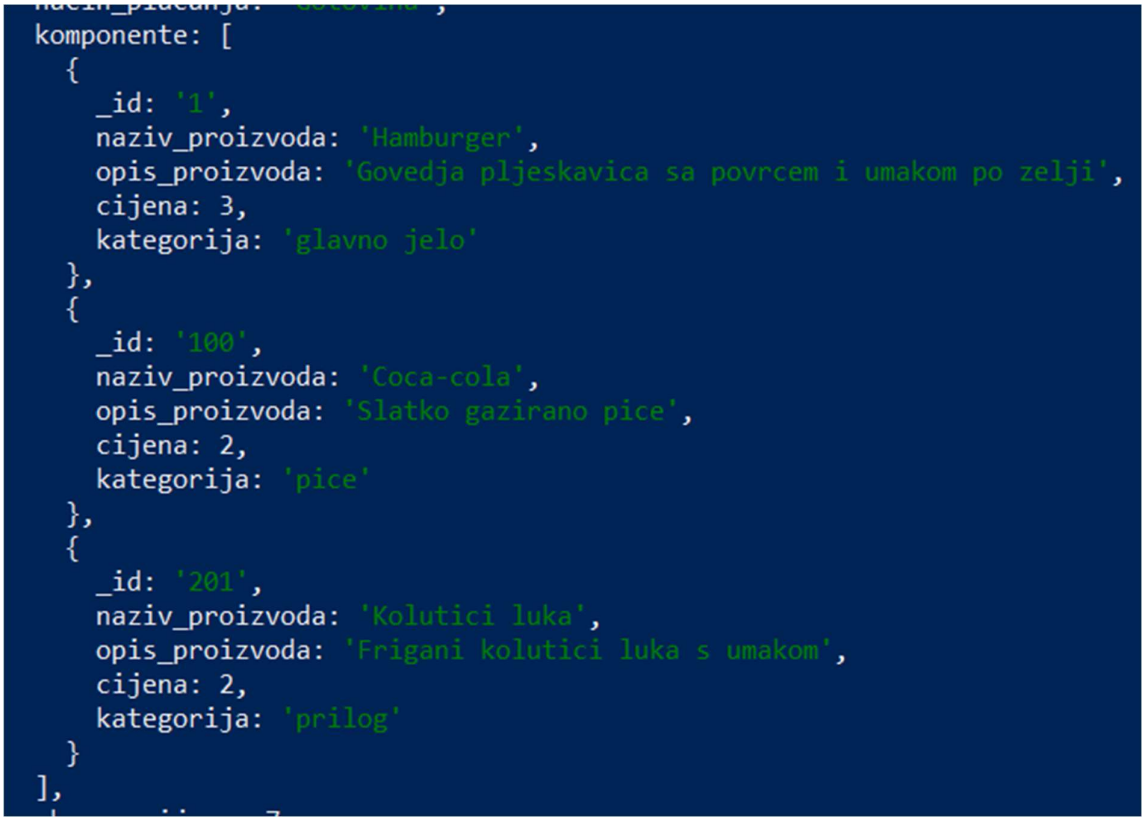
```
▾ komponente_narudzbe: Array
  ▾ 0: Object
      id_komponente: "1"
  ▾ 1: Object
      id_komponente: "100"
```

Slika 46 Komponente narudžbe

Suprotno $pull naredbi moguće je koristiti $push naredbu kojom je moguće dodati elemente u polje. Kada bi klijent želio dodati kolutiće luka u svoju narudžbu tada bi ažuriranje izgledalo ovako [19]:

```
db.narudzbe.updateOne( {_id: ObjectId ("649ab1a30e5f58f7f9864d12") },
{$push: {"komponente_narudzbe": {id_komponente: "201"} } } )
```

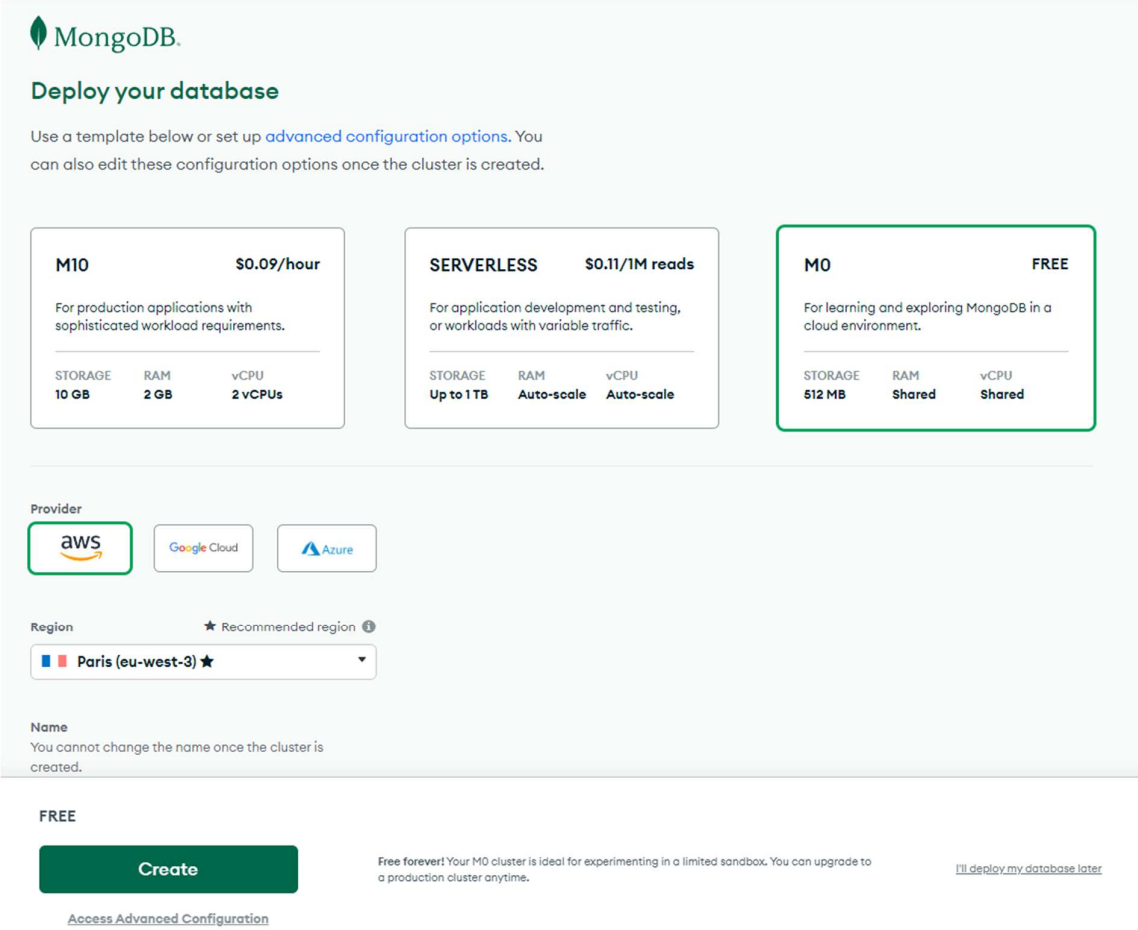
Upotrebom prošlog upita moguće se uvjeriti da se novi proizvod stvarno nalazi u narudžbi:



Slika 47 Pregled ažurirane narudžbe

7. MONGODB ATLAS

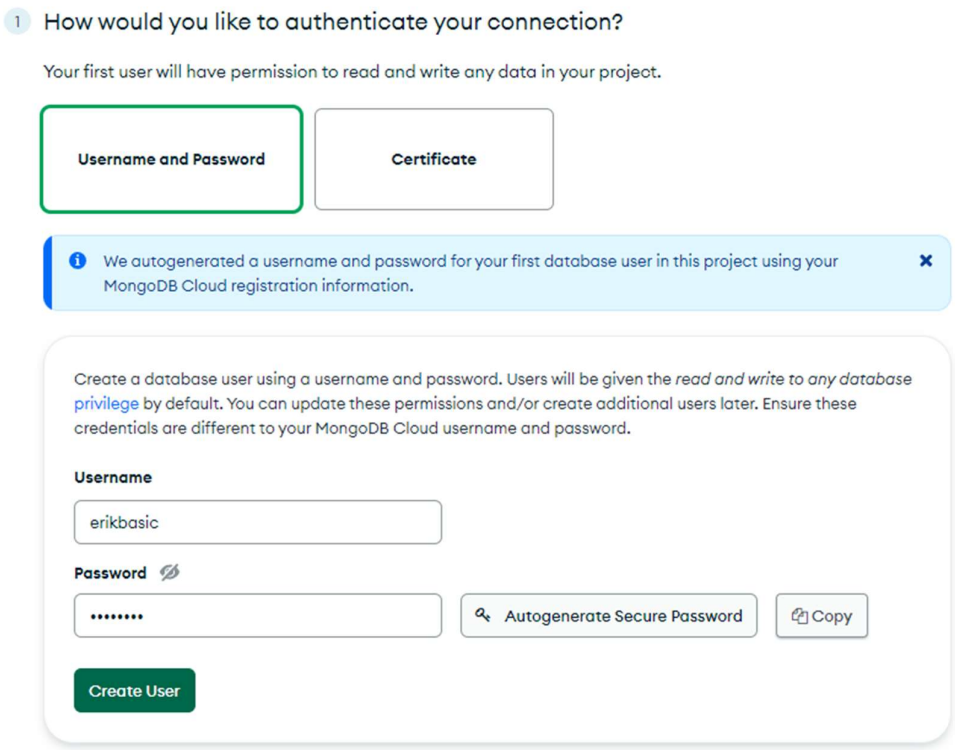
MongoDB Atlas je baza podataka u oblaku od strane istih ljudi koji su napravili MongoDB. Atlas pojednostavljuje implementaciju i upravljanje bazama podataka. Kao poslužitelje za bazu oblaku moguće je odabrati AWS, Google Cloud ili Azure.



Slika 48 Odabir poslužitelja

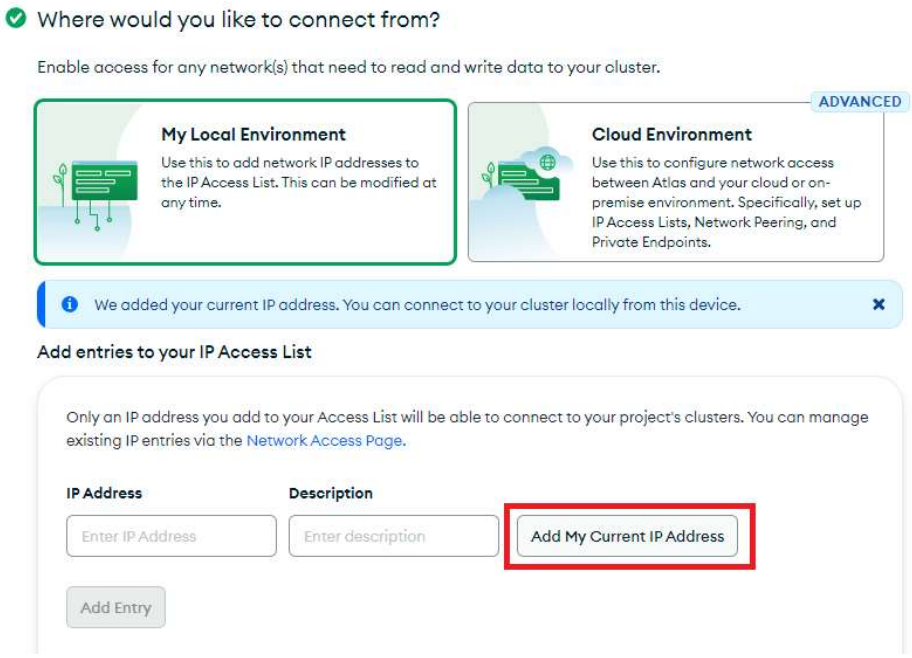
M0 je besplatni klaster sa prostorom za spremanje do 512 MB. Kao poslužitelj odabrati ćemo Amazon Web Service. Nakon toga potrebno je odabrati najbližu regiju trenutnoj lokaciji kako bi upiti radili najbrže.

Nakon stvaranja klastera potrebno je odabrati na koji način će se izvršiti konekcija. Opcije su sa korisničkim imenom i lozinkom ili certifikatom. Ukoliko je odabrana opcija korisnika s lozinkom potrebno je stvoriti novog korisnika [20].



Slika 49 Odabir načina konekcije

Klikom na gumb „Create User“ uspješno se stvara novi korisnik. Nakon toga potrebno je dodati i IP adresu s koje će biti dozvoljeno spajanje na klaster. Atlas nudi opciju dodavanja trenutne IP adrese koju ćemo iskoristiti.

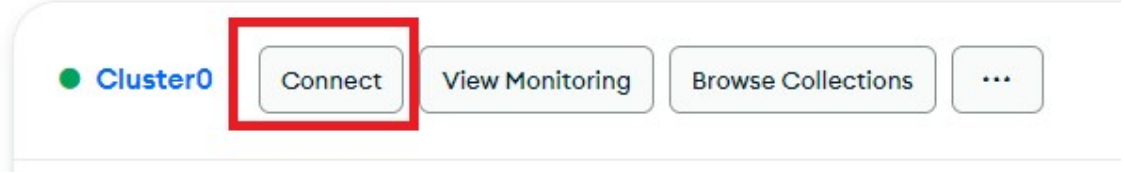


Slika 50 Dodavanje trenutne IP adrese

Klaster je moguće vidjeti klikom na „Database“ u lijevom izborniku.

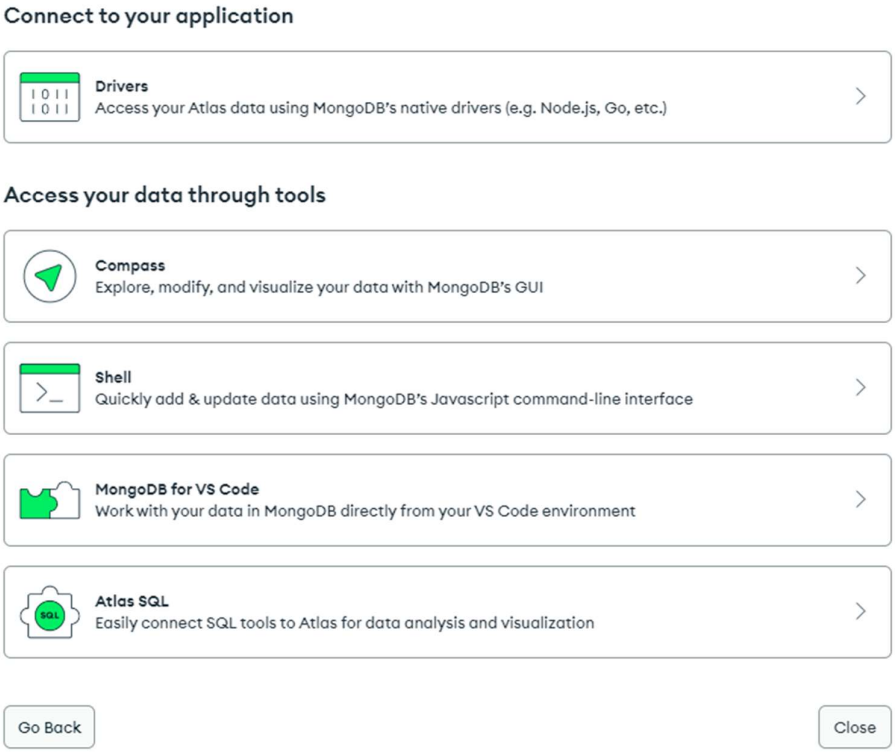
Slika 51 Odabir opcije "Database"

Zatim kliko na gumb „Connect“ moguće je odabrati na koji način se želimo spojiti sa Atlasom.



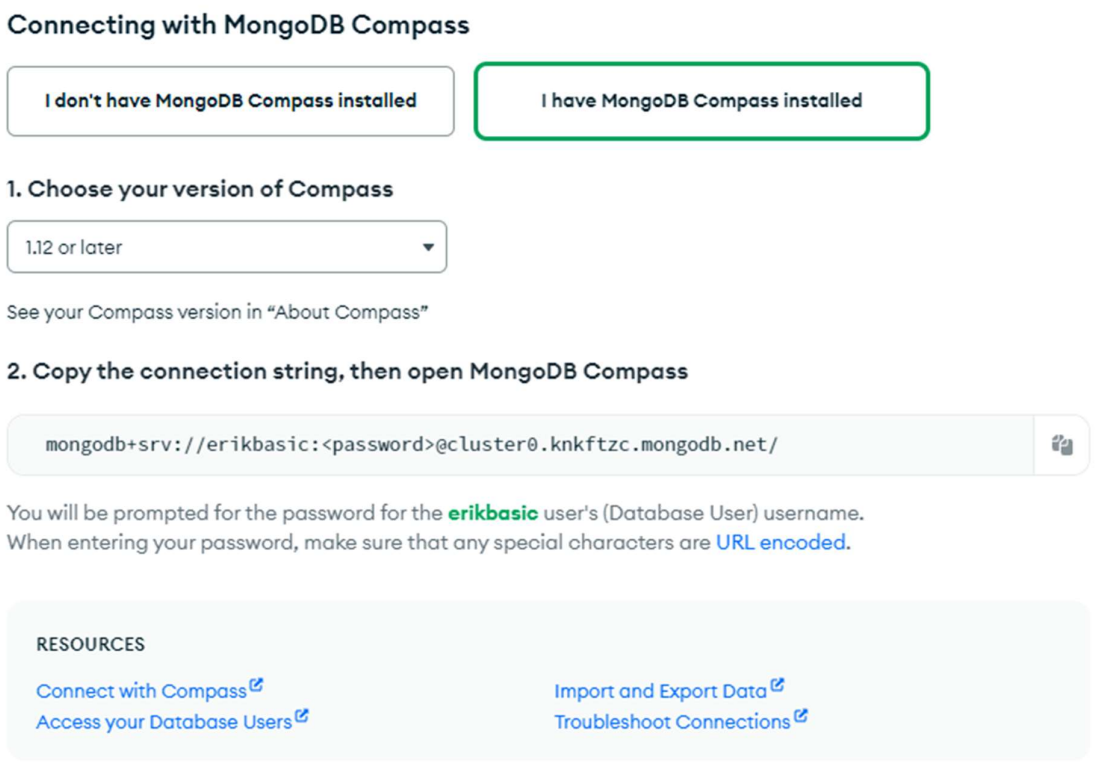
Slika 52 Spajanje na klaster

Na klaster se moguće spojiti putem aplikacije, Compass-a, Shell-a, VSCode-a i Atlas SQL-a [21].



Slika 53 Opcije spajanja

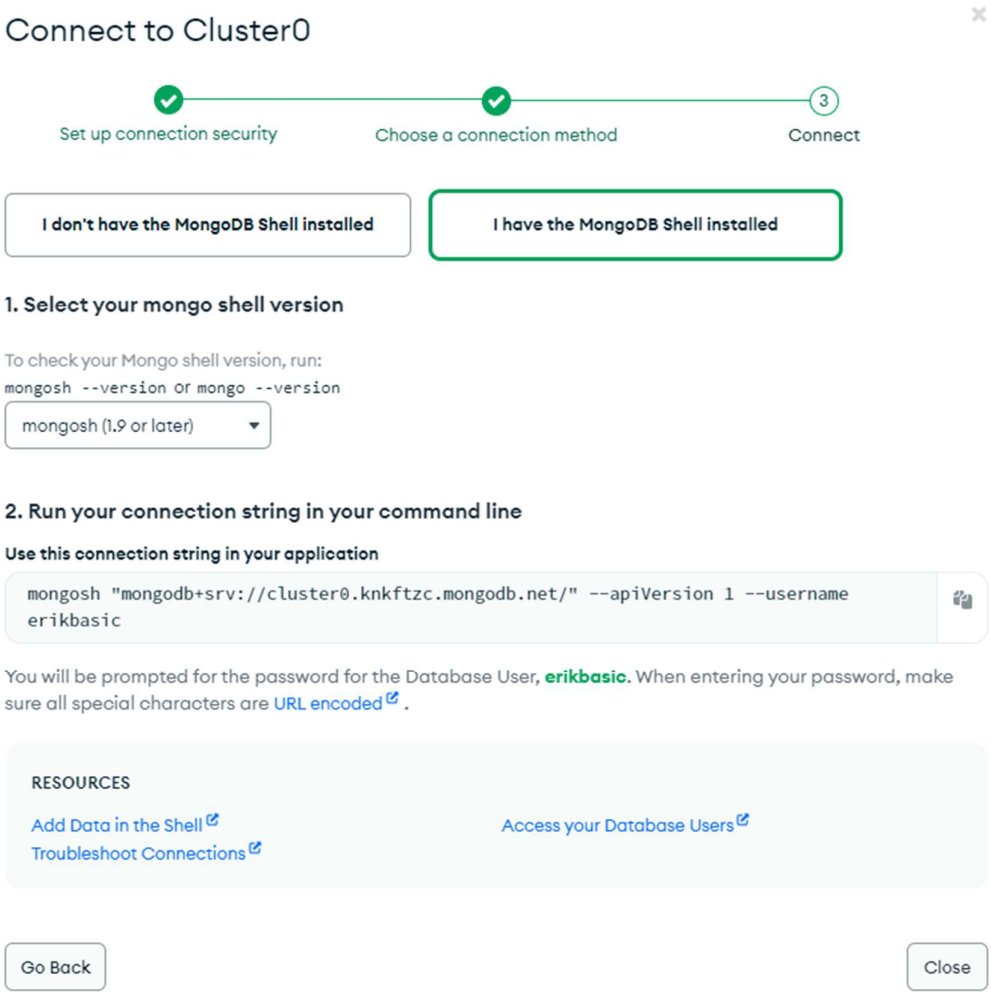
Odabrati ćemo opciju spajanja preko Compass-a. Nakon toga potrebno je odabrati koristimo li verziju Compassa stariju ili noviju od verzije 1.12.



Slika 54 Stvaranje poveznice za spajanje

Pod drugim korakom dobivamo URL preko kojeg se spajamo na klaster. URL je potrebno kopirati prilikom otvaranja Compass-a. Unutar URL-a potrebno je unijeti definiranu korisničku lozinku.

Na sličan način moguće je spojiti se i putem Shell-a. Nakon odabira opcije „Shell" potrebno je odabrati koja verzija Shell-a je instalirana te nakon toga potrebno je kopirati URL u Windows Power Shell.



Slika 55 Stvaranje poveznice za Shell

Nakon kopiranja URL-a potrebno je unijeti definiranu korisničku lozinku.

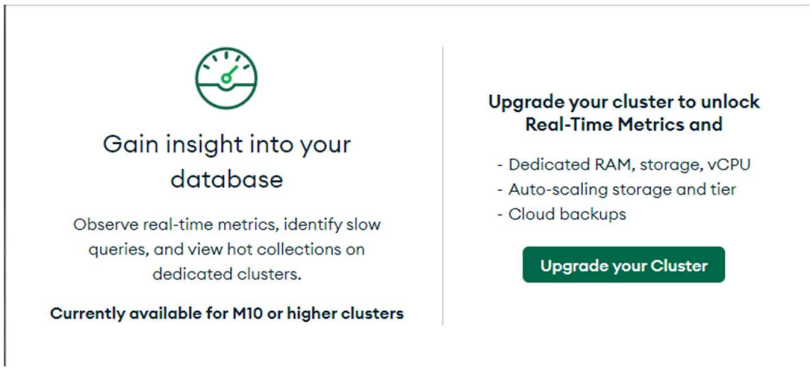
```
PS C:\Users\basic> mongosh "mongodb+srv://cluster0.knkftzc.mongodb.net/" --apiVersion 1 --username erikbasic
Enter password: ********
Current Mongosh Log ID: 64a3ed5d6e13bdb3414181b2
Connecting to:          mongodb+srv://<credentials>@cluster0.knkftzc.mongodb.net/?appName=mongosh+1.9.1
Using MongoDB:          6.0.6 (API Version 1)
Using Mongosh:          1.9.1

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

Atlas atlas-11ut3e-shard-0 [primary] test>
```

Slika 56 Spajanje putem Shell-a

Osim mogućnosti spajanja sa Shellom ili Compass-om, moguće je raditi i direktno u web sučelju.

Web sučelje sadrži mnogo opcija i mogućnosti, no neke od njih su dostupne samo ukoliko se radi sa klasterima koji se plaćaju. Pomoću plaćenih opcija moguće je identificirati sporije upite u bazi te promatrati metrike baze u realnom vremenu.

Npr. opcija „Real Time“ dostupna je samo za klastere klase M10 ili više.

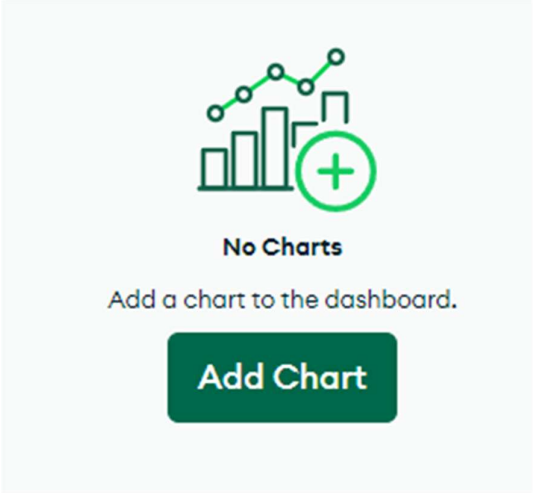


Slika 57 Real time opcija

MongoDB Atlas sadrži opciju stvaranja grafikona. Grafikon je moguće stvoriti klikom na opciju „Charts“ [22].

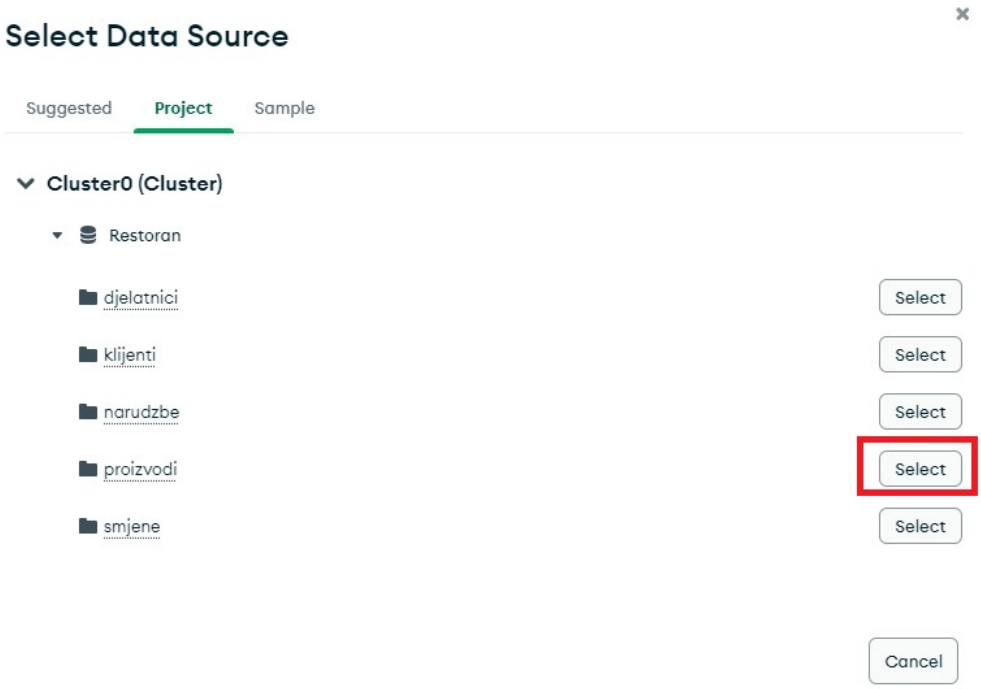
Slika 58 Opcija "Charts"

Nakon toga potrebno je stvoriti novi „Dashboard“. Nakon stvaranja i dodjeljivanja imena moguće je dodati i prvi grafikon.



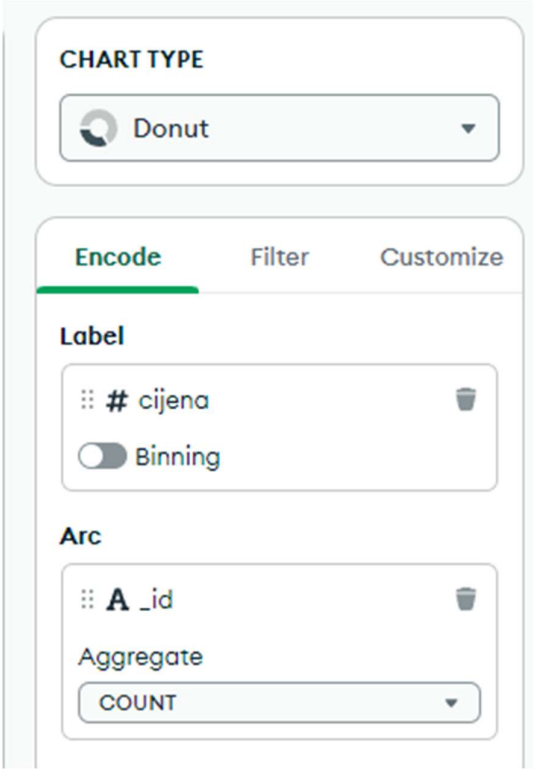
Slika 59 Dodavanje novog grafa

Nakon odabira opcije „Add Chart“ potrebno je odabrati s kojeg klastera će se koristiti podaci. Uz klaster potrebno je odabrati i bazu podataka te kolekciju. Ukoliko bi htjeli stvoriti npr. grafikon cijena proizvoda odabrali bi kolekciju proizvodi.



Slika 60 Odabir kolekcije proizvodi

Nakon odabira kolekcije Atlas će sam preporučiti grafove. No želimo stvoriti tortni grafikon na kojemu ćemo prikazati cijene proizvoda.



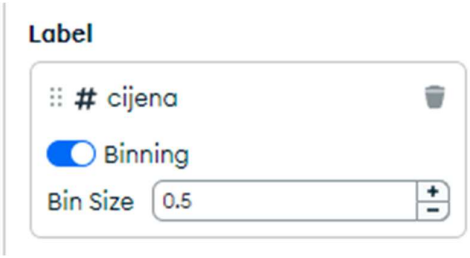
Slika 61 Odabir tortnog grafa

Nakon postavljanja postavki kao na slici „X“ tortni grafikon izgleda ovako:

Slika 62 Tortni grafikon cijena proizvoda

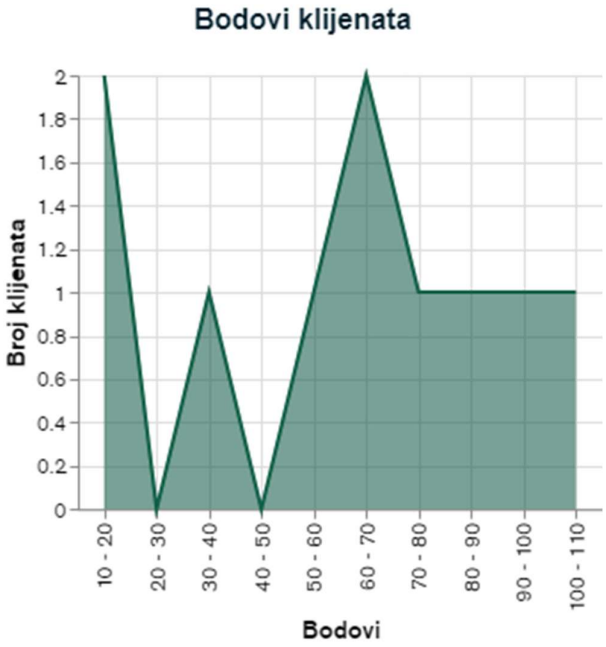
Dobiveni graf je interaktivan što znači da prelaskom kursora dobivamo informaciju koliko se proizvoda nalazi u određenoj grupi (npr. 5 proizvoda za cijenu 2)

Ukoliko umjesto točnih cijena želimo koristiti opseg cijena. Na primjer, ukoliko želimo da graf pokazuje sve proizvode koji su u grupi cijena od 1.5 do 2 eura onda je potrebno uključiti „Binning“ te postaviti vrijednost na 0.5.



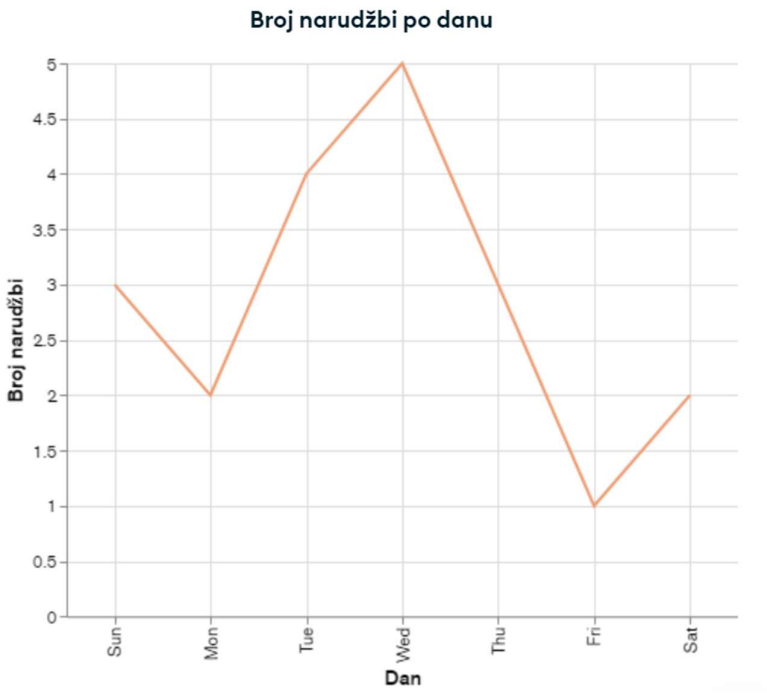
Slika 63 Modificiranje grafa

Sljedeći graf je linijski grafikon koji pokazuje broj klijenata s određenim rasponom bodova.



Slika 64 Broj bodova klijenata

Sljedeći graf prikazati će broj narudžbi po danu. Uz opciju po danu moguće je odabrati opcije po tjednu, mjesecu, satu itd.

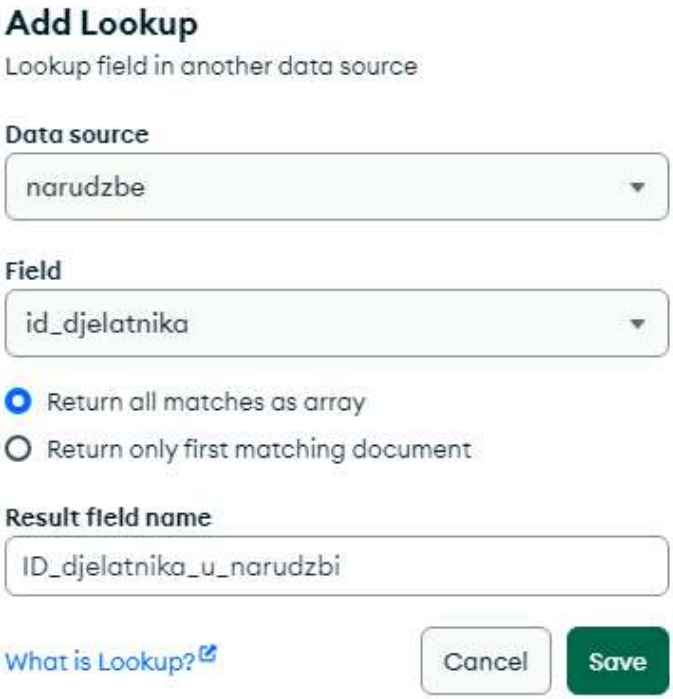


Slika 65 Prikaz narudžbi po danu

MongoDB Charts nudi prikaz podataka tablično. Ukoliko bi htjeli pokazati broj odrađenih narudžbi za svakog djelatnika tada bi tablični prikaz bio najbolji. Prvi korak pri izradi ovakvog prikaza je dodavanje Lookup polja.

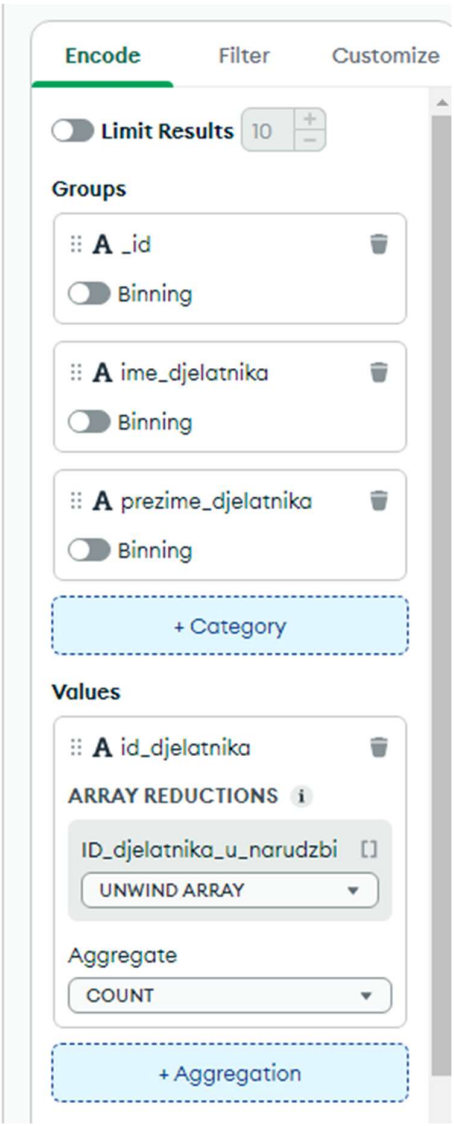
Slika 66 Dodavanje Lookup polja

U padajućem izborniku potrebno je odabrati „Lookup Field“. Nakon toga potrebno je odabrati izvor podataka, polje koje želimo dohvatiti i naziv polja.



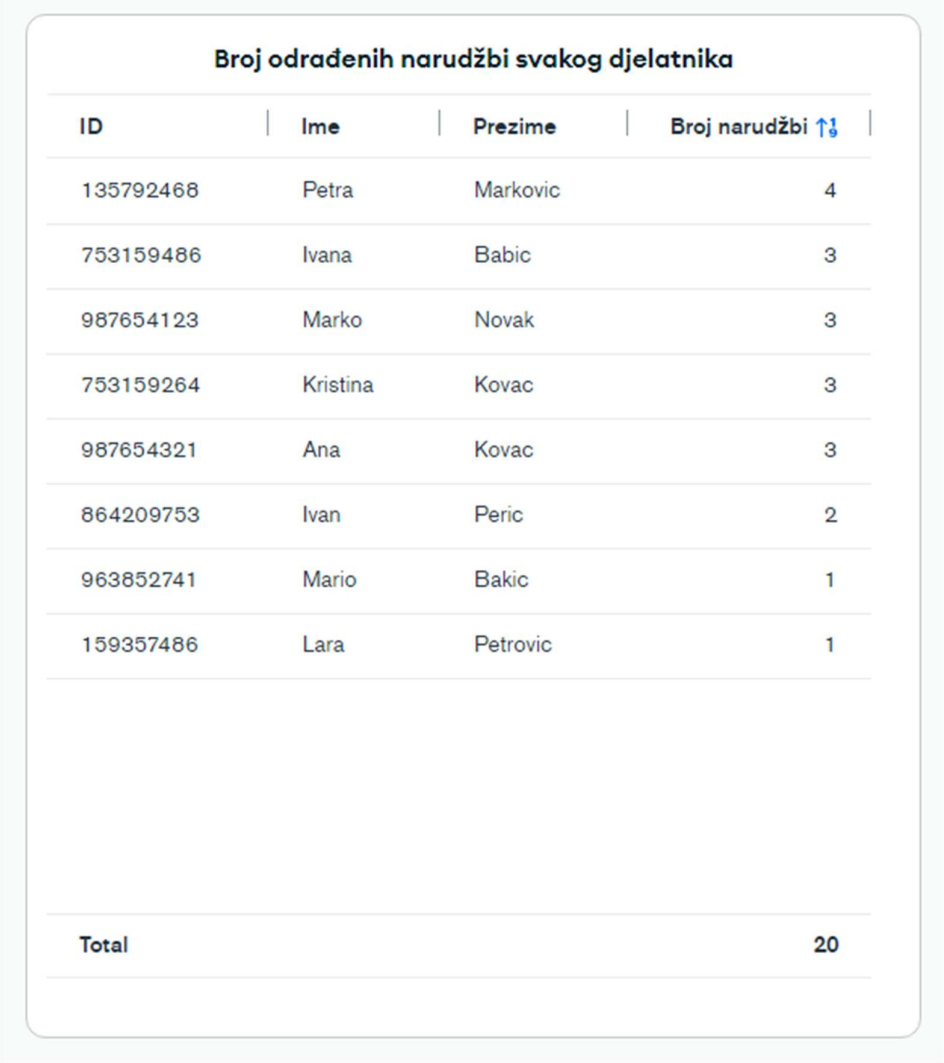
Slika 67 Unos informacija

Prilikom izrade tablice varijable je potrebno postaviti na sljedeći način.



Slika 68 Postavljanje varijabli

Pod „Groups“ je potrebno odabrati sve informacije koje želimo prikazati (u ovom slučaju ID, ime i prezime). Dok pod opcijom „Values“ je potrebno povući polje „id_djelatnika“ ali ono koje se nalazi u Lookup polju. Nakon toga MongoDB Charts sam stvara tablicu na kojoj je vidljivo da točno izračunava broj narudžbi svakog djelatnika.



Broj odrađenih narudžbi svakog djelatnika

ID	Ime	Prezime	Broj narudžbi
135792468	Petra	Markovic	4
753159486	Ivana	Babic	3
987654123	Marko	Novak	3
753159264	Kristina	Kovac	3
987654321	Ana	Kovac	3
864209753	Ivan	Peric	2
963852741	Mario	Bakic	1
159357486	Lara	Petrovic	1
Total			**20**

Slika 69 Tablični prikaz podataka

8. ZAKLJUČAK

U radu su predstavljene nerelacijske baze podataka s naglaskom na MongoDB. Ukratko su objašnjene nerelacijske baze s naglaskom na dokumentne baze. Zatim je prikazan i objašnjen model podataka koji će se koristiti pri izradi projekta u MongoDB bazi. Nakon toga prikazan je način instalacije alata MongoDB Compass i MongoDB Shell. Nakon instalacije slijedi prikaz rada u MongoDB Compass-u. Objašnjen je način spajanja na lokalnu bazu. Zatim je objašnjen način na koji preko grafičkog sučelja možemo stvoriti novu bazu podataka te stvoriti kolekcije. Nakon stvaranja kolekcija objašnjeno je na koji način je moguće unositi jedan ili više dokumenata. Uz stvaranje prikazano je ažuriranje te brisanje dokumenata. Slijedi prikaz rada u MongoDB Shell-u preko kojeg su objašnjene iste stvari koje su objašnjene u Compass-u. Prikazan je niz naredba koje je potrebno unijeti u PowerShell ljusku za stvaranje baza i dodavanje novih kolekcija i dokumenata. Zatim je objašnjeno na koji način je moguće raditi upite u MongoDB-u. Prikazan je način izvođenja upita u Compass-u i Shell-u. Prikazano je mnogo upita počevši od jednostavnijih do kompliciranijih u kojemu se koriste operatori za logičke i aritmetičke operacije. Nakon toga prikazan je način izvođenja upita nad više kolekcija. Prikazan je upit nad kojim se vrši spajanje kolekcija djelatnik i smjene te upit koji spaja narudžbu i proizvode. Nakon kompleksnijih upita, prikazan je način ažuriranja i brisanja dokumenata putem Shell-a. U zadnjem poglavlju prikazan je rad alatom MongoDB Atlas. Prikazan je način registracije na ovaj servis u oblaku. Nakon toga prikazano je na koji način je moguće spojiti bazu podataka u oblaku sa alatima koji su objašnjeni od prije. Nakon toga detaljnije je objašnjena funkcionalnost grafova koju nudi ovaj servis. Prikazan je način stvaranja 2 grafa te na koji način se oni mogu modificirati.

Kao baza podataka, MongoDB je pružila solidnu osnovu za implementaciju projekta. S obzirom na korišteni model, baza je omogućila fleksibilno pohranjivanje podataka. Cijeli MongoDB sustav sadrži raznolik skup alata za upravljanje i korištenje podataka. Ovisno o znanju, potrebama i preferencijama moguće je koristiti grafičko sučelje ili rad u ljusci kako bi se ostvarili isti zadaci. Jezik upita nad ovom bazom omogućuje fleksibilno pretraživanje strukturiranih i nestrukturiranih podataka. Raditi u bazi MongoDB bilo je vrlo zanimljivo. Dokumentacija je vrlo dobra te je objasnila sve stvari vrlo jednostavno i temeljito.

MongoDB primjer je dokumentne baze. Takve baze koriste se pri aplikacijama s velikim obujmom podataka gdje su visoka skalabilnost i brza izvedba prioriteti. Ove baze sve češće se koriste u web i mobilnim aplikacijama, e-trgovinama, videoigrama itd. MongoDB jedna je od najpopularnijih i najpoznatijih dokumentnih baza podataka na tržištu. Prema mnogim stranicama nalazi se među najboljih 5 nerelacijskih baza podataka.

Nerelacijske baze nastavljaju razvijati svoje performanse, skalabilnost i sigurnost. Tvrtke sve češće uočavaju prednosti ovakvih baza te se očekuje da bi popularnost nerelacijskih baza podataka mogla sve više rasti.

LITERATURA:

1. C. Strauch, W. Kriha „NoSQL Databases“, 2011.
2. NoSQL, Wikipedija, Preuzeto 11.7.2023. https://en.wikipedia.org/wiki/NoSQL
3. G. Vaish „Getting Started With NoSQL“, 2013.
4. Materijali s kolegija „Infrastruktura za podatke velikog obujma“
5. Install MongoDB on Windows, Preuzeto 28.6.2023. https://www.mongodb.com/docs/manual/tutorial/install-mongodb-on-windows/
6. Install mongosh, Preuzeto 28.6.2023. https://www.mongodb.com/docs/mongodb-shell/install/
7. The Local Database, Preuzeto 5.7.2023. https://www.mongodb.com/docs/manual/reference/local-database/
8. Databases, Preuzeto 6.7.2023. https://www.mongodb.com/docs/compass/current/databases/
9. Insert, Preuzeto 6.7.2023. https://www.mongodb.com/docs/compass/current/documents/insert/
10. Create-database, Preuzeto 6.7.2023. https://www.mongodb.com/basics/create-database
11. Query Filter, Preuzeto 11.7.2023. https://www.mongodb.com/docs/compass/current/query/filter/
12. Query Documents, Preuzeto 11.7.2023. https://www.mongodb.com/docs/mongodb-shell/crud/read/
13. Query Operators, Preuzeto 11.7.2023. https://www.mongodb.com/docs/manual/reference/operator/query/
14. Database References, Preuzeto 11.7.2023. https://www.mongodb.com/docs/manual/reference/database-references/
15. DB Collection Aggregate, Preuzeto 11.7.2023. https://www.mongodb.com/docs/manual/reference/method/db.collection.aggregate/
16. Delete, Preuzeto 11.7.2023. https://www.mongodb.com/docs/mongodb-shell/crud/delete/
17. Update, Preuzeto 11.7.2023. https://www.mongodb.com/docs/manual/reference/method/db.collection.update/
18. Pull, Preuzeto 11.7.2023. https://www.mongodb.com/docs/manual/reference/operator/update/pull/
19. Push, Preuzeto 11.7.2023. https://www.mongodb.com/docs/manual/reference/operator/update/push/
20. Create Atlas Account, Preuzeto 11.7.2023. https://www.mongodb.com/docs/atlas/tutorial/create-atlas-account/
21. Connect to Your Cluster, Preuzeto 11.7.2023. https://www.mongodb.com/docs/atlas/tutorial/connect-to-your-cluster-v2/
22. Charts, Preuzeto 11.7.2023. https://www.mongodb.com/docs/charts/

POPIS TABLICA:

POPIS SLIKA:

PRILOZI:

1. Poveznica za preuzimanje MongoDB-a lokalno
https://www.mongodb.com/try/download/community

2. Poveznica za MongoDB Atlas https://www.mongodb.com/atlas/database