

# ARHITEKTURA I DIZAJN VEB PLATFORME ZA UPRAVLJANJE KORISNIČKIM ŽALBAMA I ZAHTEVIMA

Aleksa Kojadinović

Matematički fakultet, Univerzitet u Beogradu  
2024

# UVOD

- Platforma za upravljanje korisničkih žalbama i zahtevima
- Zašto veb?
  - Univerzalnost veb pregledača
  - Ažuriranje aplikacije
  - Integracija sa eksternim servisima

# KARTICA

- Jedna asinhrona korisnička žalba ili zahtev
- Može imati dodeljenje agente
- Može biti označena različitim oznakama
- Prolazi kroz statuse
- ...

The screenshot displays a Zendesk support ticket interface. On the left, a sidebar contains fields for 'Assignee\*' (Support/Lisa Kelly), 'Followers' (Jbantz), 'Sharing' (z3nlondon), 'Tags' (password, sign\_in), 'Type' (Problem), 'Priority' (High), and 'Product type' (Saas Cloud). The main area shows the ticket title 'Trouble signing in', the time '14 minutes ago', and the user 'Venkat Kumar'. Below this, a 'Public reply' section shows a message from Venkat Kumar: 'Hello Support, Can you help? I'm having trouble with my password. Venkat'. The interface also includes options for 'Internal note' and 'Call', and a 'Conversations' tab with 'All' and 'Public' sub-tabs.

# STS FUNKCIONALNOSTI

## Brzina interneta je iznenada opala

29/04/2024 u 11:32 AM

by Benjamin Turner

Novo

DODELI NEKOME

Dodaj oznaku

PROMENI STATUS

Dobar dan,

Koristim usluge Vašeg interneta već nekoliko meseci bez problema. Međutim, od skoro sam primetio značajan pad u brzini.

Da li biste mogli da proverite šta se dešava?

Hvala unapred.

# STS FUNKCIONALNOSTI

Ullamco laboris eu consequat pariatur

19/08/2023 u 05:54 PM

by Olivia Reynolds

Otvoreno

AA Administrator Administrator ✕

DODELI NEKOME

Hitnost

Veoma hitno ✕

Odeljenje

Tehnička podrška ✕

Dodaj oznaku

PROMENI STATUS

# STS FUNKCIONALNOSTI

## Odeljenje

Koje odeljenje treba da adresira ovu karticu?

Ime

Jezik	Value
en	Department
sr	Odeljenje

Opis


Jezik	Value
en	Which department should handle this?
sr	Koje odeljenje treba da adresira ovu karticu?

# STS FUNKCIONALNOSTI

## Dozvole:

Ko može dodavati ove oznake?

administrator 

agent 


Ko može ukloniti ove oznake?


administrator 

agent 

Ko može videti ove oznake?

administrator 

agent 

customer 

SAČUVAJ

# STS FUNKCIONALNOSTI

ES Emma Smith premešteno iz **Novo** na **Otvoreno** na 29/04/2024 u 11:37 AM

ES Emma Smith premešteno iz **Otvoreno** na **U toku** na 29/04/2024 u 11:37 AM

ES Emma Smith premešteno iz **U toku** na **Rešeno** na 29/04/2024 u 11:37 AM

Ne možete komentarisati ovaj tiket



# STS FUNKCIONALNOSTI

NAPRAVI NOVOG KORISNIKA

Pretraži	Uloge	Status		
Puno ime	Uloga	Status	Pristup AI	Akcije
Administrator Administrator	Administrator	Aktivan	Da	<div><div>PROMENI STATUS</div><div>PROMENI ULOGU</div><div>POVUCI PRISTUP AI</div></div>
Emma Smith	Agent			<div><div>PROMENI STATUS</div><div>PROMENI ULOGU</div><div>POVUCI PRISTUP AI</div></div>
Oliver Johnson	Agent			<div><div>PROMENI STATUS</div><div>PROMENI ULOGU</div></div>

# STS FUNKCIONALNOSTI

## Kako da promenim svoju metodu plaćanja?

30/04/2024 u 10:36 AM

od strane Benjamin Turner

U toku

ES Emma Smith

DODELI NEKOME

Dodaj oznaku

PROMENI STATUS

PRIKAŽI SAŽETAK AI

Zdravo,

Vaše usluge do sada sam plaćao direktno karticom. Sada plaćam putem PayPal-a. Nisam siguran da li je to ispravno.

Hvala unapred.

### AI summary

Benjamin wants to switch from direct debit card payment to PayPal. Emma advised to go to Settings -> Payment methods but the button for change was greyed out. Logging out and back in did not resolve the issue.

ZATVORI

# BEKEND

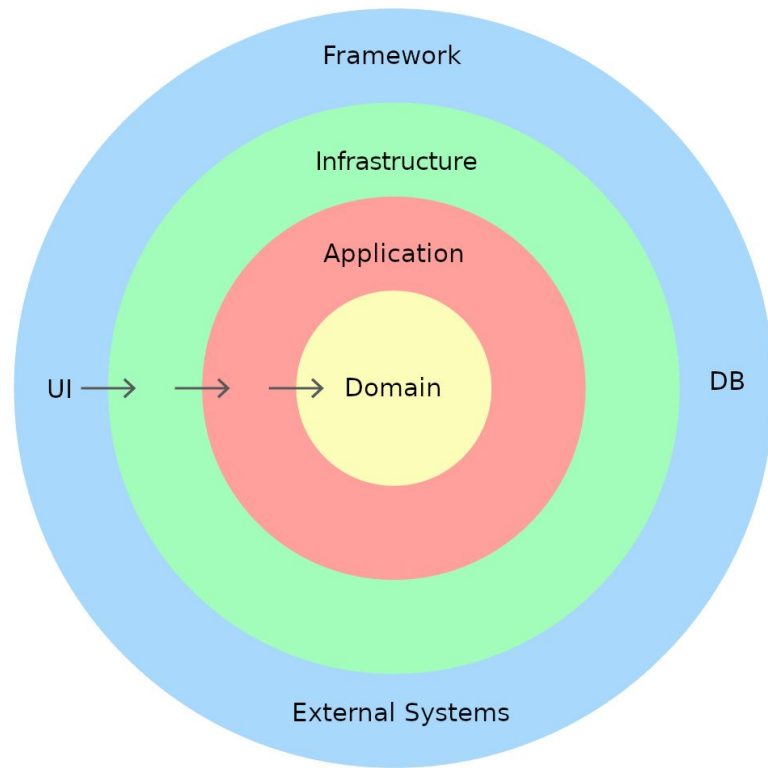


# DDD

Uloga softvera je rešenje problema iz realnog sveta, i sve komponente rade za taj cilj. Kompleksnost neminovno dolazi iz modela i ne treba je izbegavati već ispravno kontrolisati.

Dva glavna aspekta:

- zajednički/svepristuni jezik
- slojevita arhitektura



# DDD – DOMENSKI JEZIK

Upotreba domenskog jezika u kodu

- održava spregu između modela i koda
- čitljivost
- samodokumentovanost



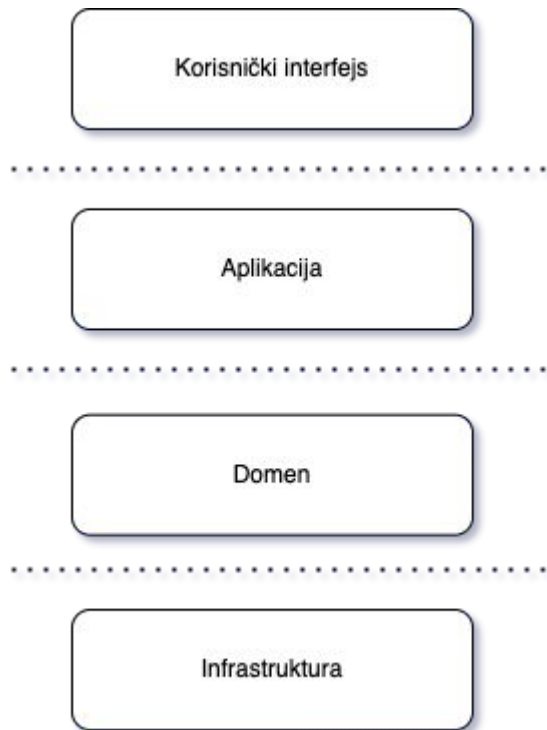
```
1 if (ticket.status === CLOSED || ticket.status === RESOLVED) {  
2   throw new BadRequestException('Ticket closed.');
```

```
3 }
```



```
1 if (ticket.isFinalStatus()) {  
2   throw new CannotChangeCommentsForTicketStatusError(ticket.status);  
3 }
```

# DDD – SLOJEVITA ARHITEKTURA



# NESTJS

Moderno okruženje jezika JavaScript za izradu backend aplikacija.

- bazirano na biblioteci *expressjs*
- modularna arhitektura
- podržava umetanje zavisnosti (Dependency Injection)



# NESTJS – DI

Tehnika upravljanja zavisnostima koja oslobađa klase odgovornosti kreiranja objekata od kojih zavise.

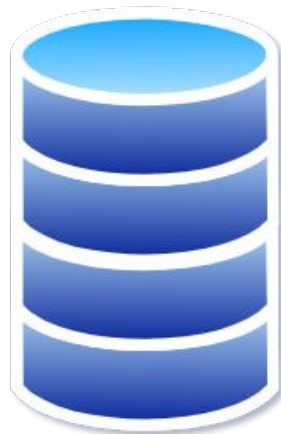
- zahteva podršku okruženja
- deklarativno se podešava.

```
1  @Module({
2    imports: [
3      mongooseModule.forFeature([ { name: TicketDb.name, schema: TicketSchema } ]),
4      UsersModule,
5      TicketTagSystemModule,
6      NotificationsModule,
7    ],
8    exports: [TicketService],
9    controllers: [TicketsController],
10   providers: [
11     TicketService,
12     TicketCommentService,
13     TicketRedactionService,
14     TicketTagUpdateService,
15     TicketAssigneesService,
16     TicketInfoService,
17     TicketsRepository,
18   ],
19 })
20 export class TicketsModule {}
21
```



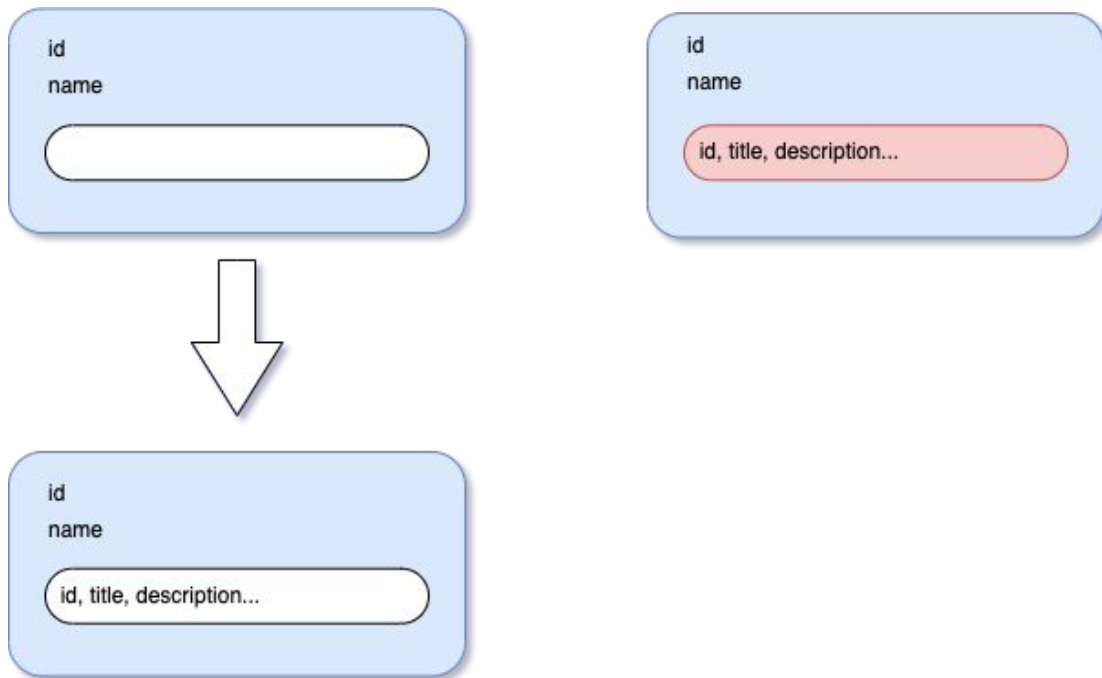
# SLOJ INFRASTRUKTURE

- Ne inicira nikakve domenske akcije.
- Implementira trajnost podataka (eng. Persistence).
- Najčešći šablon je repozitorijum (eng. *Repository*).

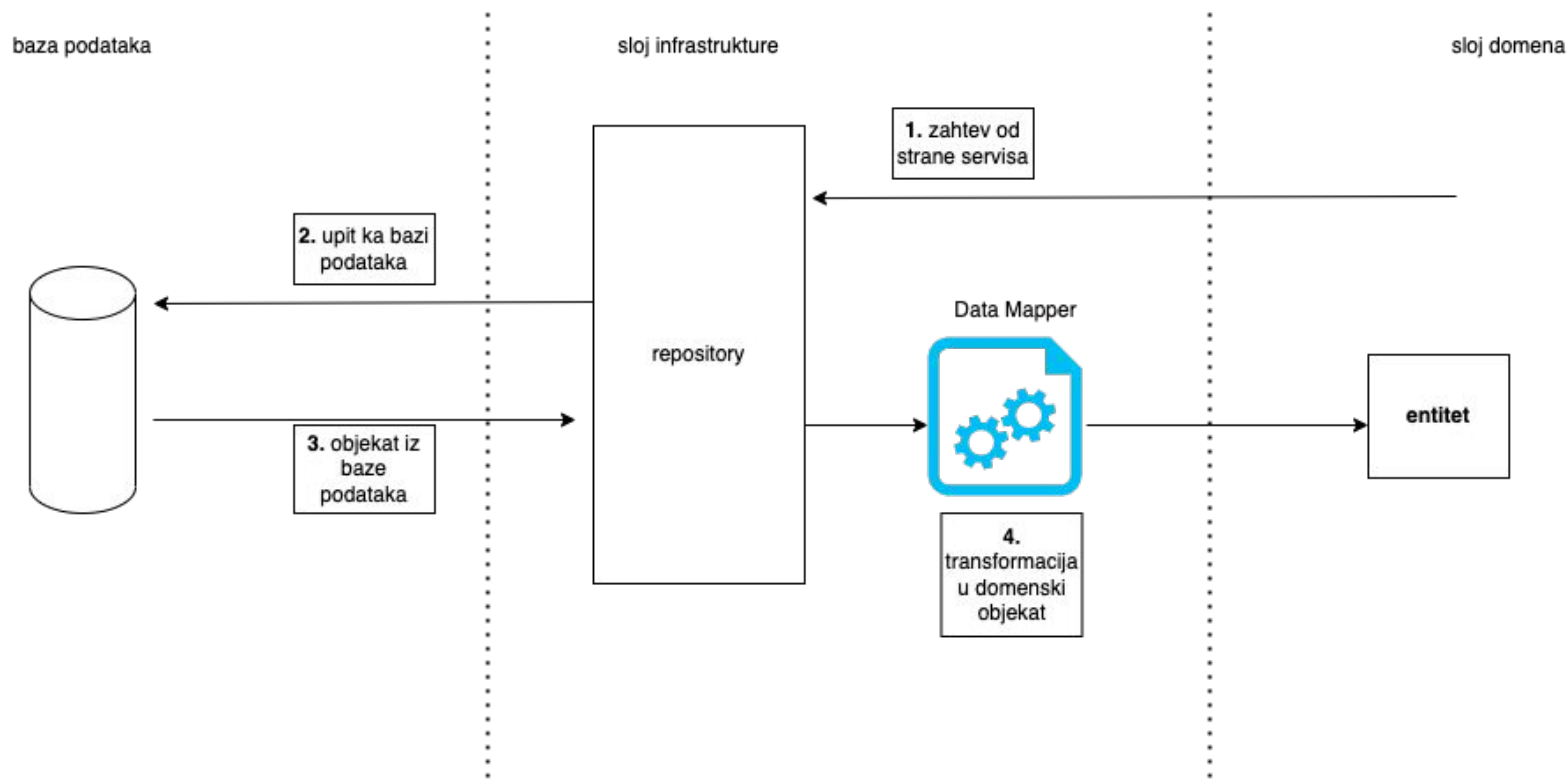


# SLOJ INFRASTRUKTURE

Izabrano je pohlepno učitavanje za sve entitete.



# SLOJ INFRASTRUKTURE



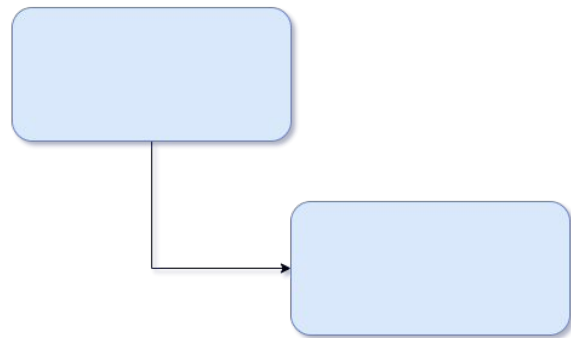
# SLOJ INFRASTRUKTURE

Preslikavanje podataka vrši se bibliotekom *automapper*, kao i na svim slojevima

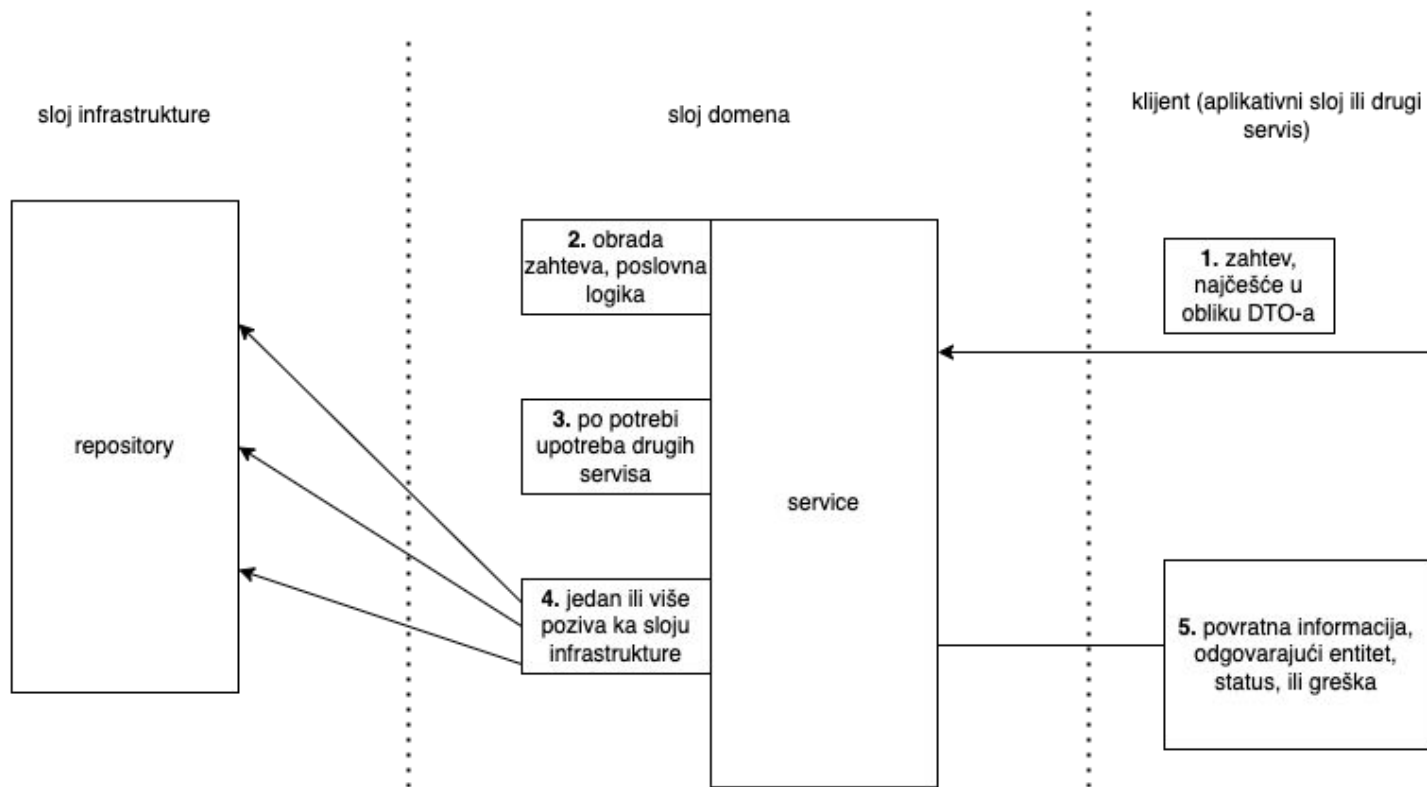
```
1  forMember(  
2      (destination) => destination.comments,  
3      mapFrom((source) => {  
4          const commentItems = source.history.filter(  
5              (item) => item.type === TicketHistoryEntryType.COMMENT_ADDED,  
6              );  
7  
8          if (commentItems.length === 0) {  
9              return [];  
10         }  
11  
12         return commentItems  
13             .map((item) => {  
14                 const payload = item.payload as TicketHistoryEntryCommentAdded;  
15  
16                 const deletes = source.history.filter(  
17                     (deleteItem) =>  
18                         deleteItem.type ===  
19                             TicketHistoryEntryType.COMMENT_DELETED &&  
20                             (deleteItem.payload as TicketHistoryEntryCommentChanged)  
21                                 .commentId === payload.commentId,  
22                     );  
23  
24                 const wasDeleted = deletes.length > 0;  
25  
26                 if (wasDeleted) {  
27                     return null;  
28                 }  
29             })  
30     })  
31 )
```

# SLOJ DOMENA

- Sadrži poslovnu logiku i operacije koje direktno oslikavaju probleme realnog sveta koji se modeluju.



# SLOJ DOMENA

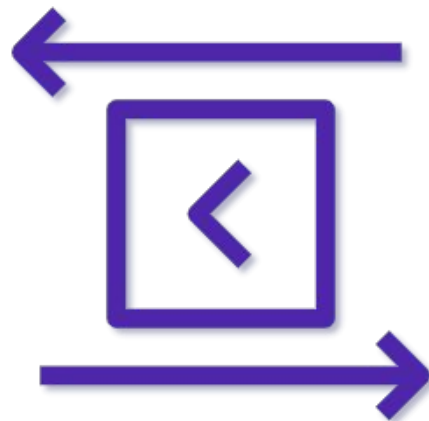


# SLOJ APLIKACIJE

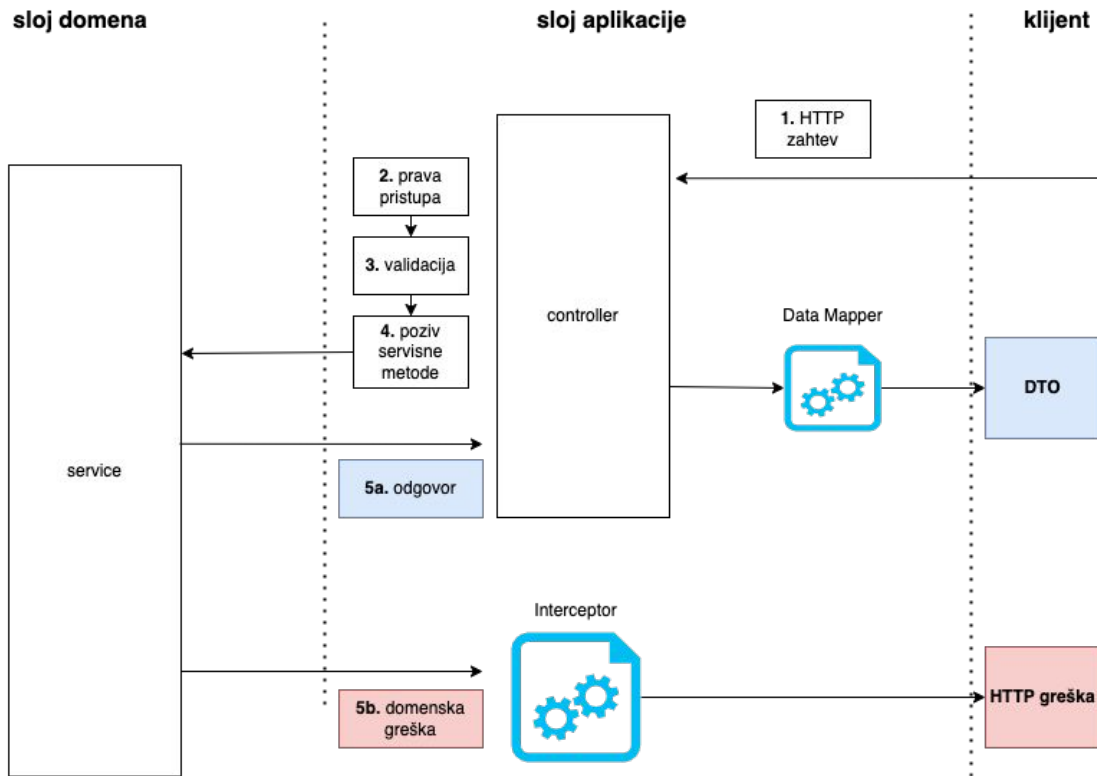
Tanak sloj nesvestan poslovne logike koji služi za ispoljavanje funkcionalnosti spoljašnjem svetu.

Izbor transportnog sloja

- REST
- GraphQL
- gRPC

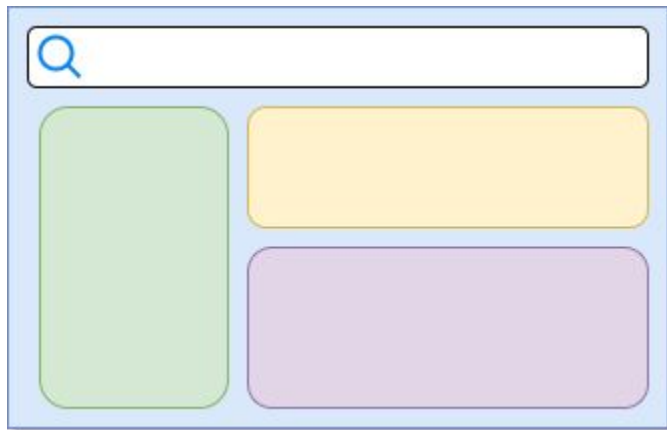


# SLOJ APLIKACIJE

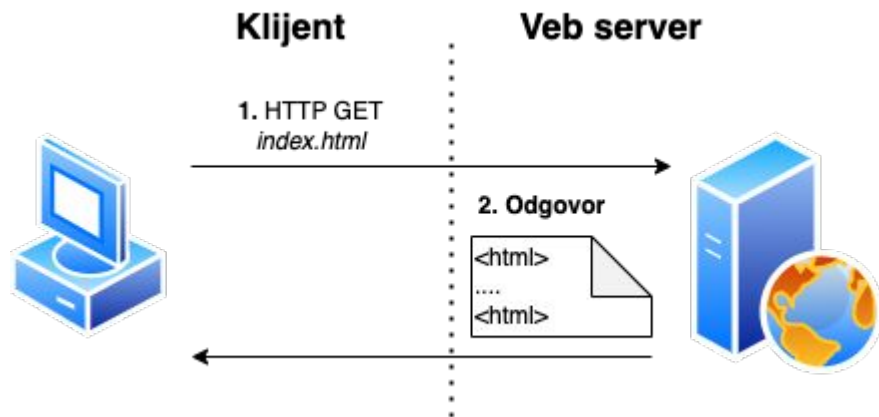




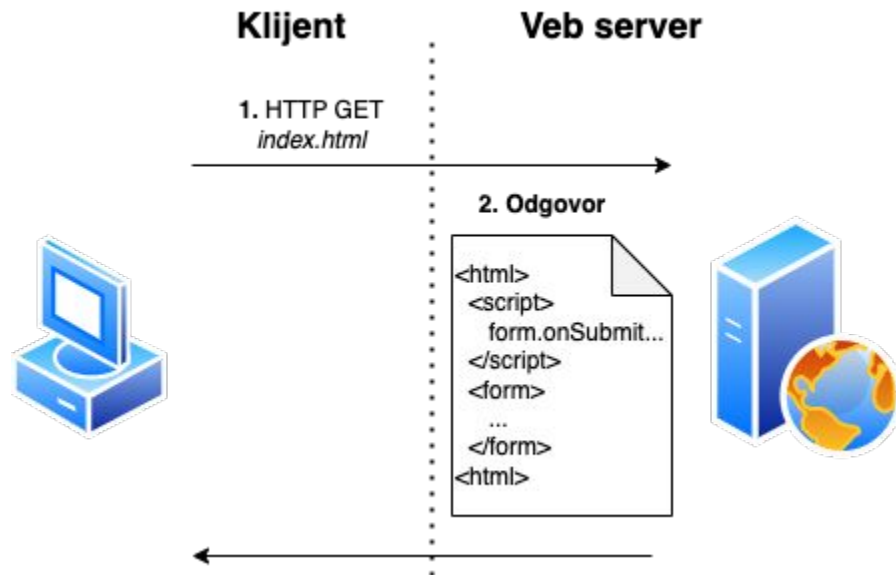
# FRONTEND



# EVOLUCIJA – STATIČKI SAJTOVI



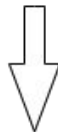
# EVOLUCIJA – SKRIPTE



# EVOLUCIJA – SPA



1. prvobitni prikaz - minimalan HTML



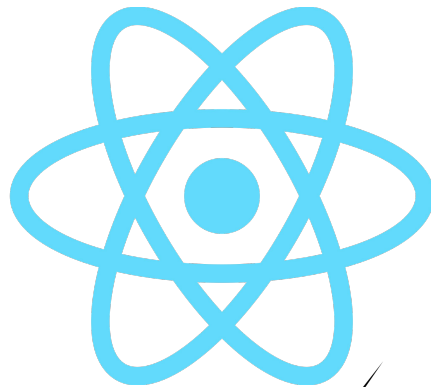
2. skripta prikazuje sadržaj

# REACT I NEXT.JS

React je jedna od najpopularnijih biblioteka za razvoj korisničkih interfejsa.

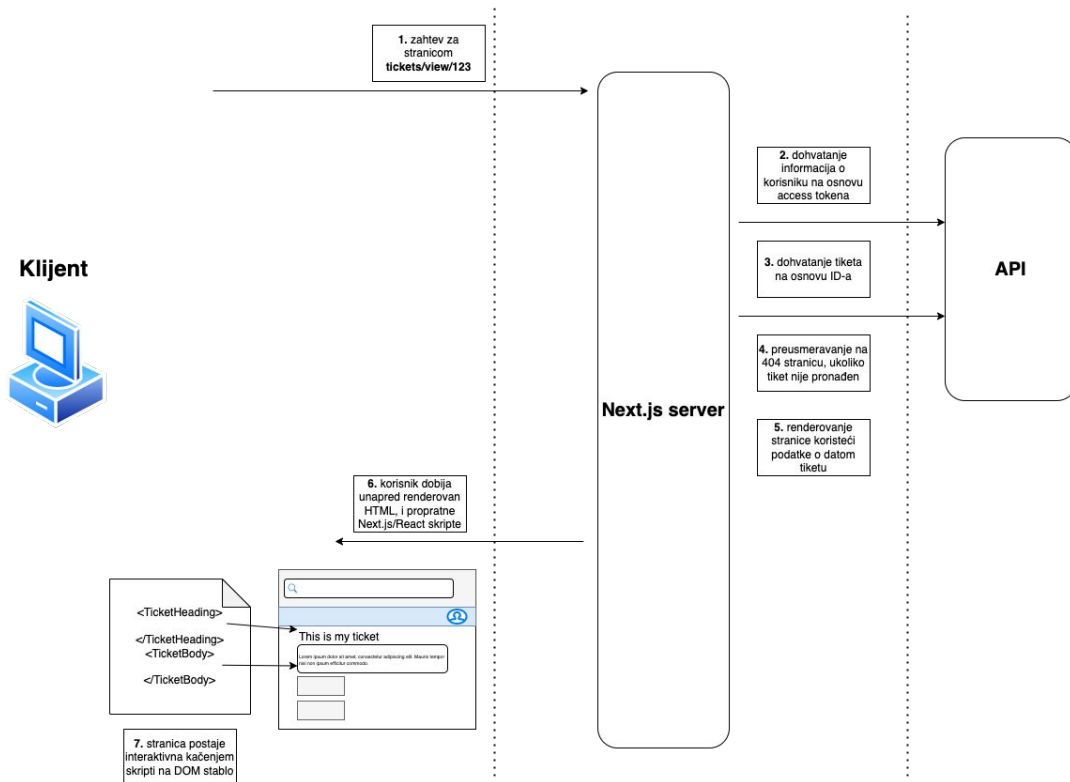
Next.js uvodi hibridne režime renderovanja za React

- CSR
- SSR
- SSG
- ISR
- Streaming SSR



NEXT.JS

# TOK ZAHTEVA



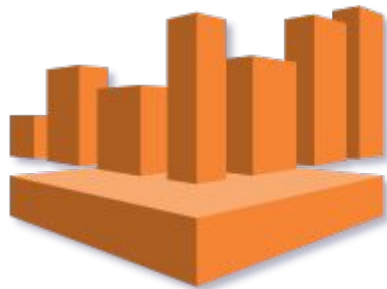
# DOHVATANJE PODATAKA

- Centralni problem u modernom frontend razvoju, pogotovu SPA.
- Redux - biblioteka za upravljanje stanjem aplikacije.
- RTKQ - koristi redux za deklarativno dohvaćanje podataka.



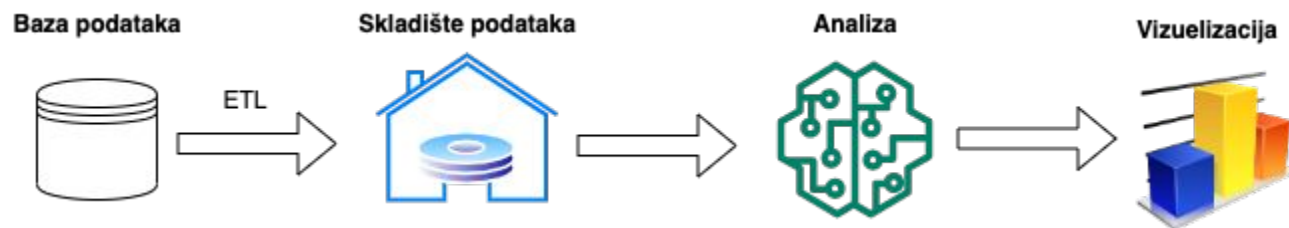
```
1 export const ticketsSlice = api.injectEndpoints({
2   endpoints: (builder) => ({
3     getTicket: builder.query({
4       query: ({ id, ...params }) => ({
5         url: `/tickets/${id}`,
6         params,
7       }),
8       providesTags: (res) => {
9         return res ? [{ type: 'getTicket', id: res.id }] : [];
10      },
11    }),
12    createTicket: builder.mutation({
13      query: (body) => ({
14        url: '/tickets',
15        method: 'POST',
16        body,
17      }),
18    }),
19    updateTicket: builder.mutation({
20      query: ({ id, ...update }) => ({
21        url: `/tickets/${id}`,
22        method: 'PATCH',
23        body: update,
24      }),
25      invalidatesTags: ({ id }) => [{ type: 'getTicket', id }],
26    }),
27  })
28 })
```

# ANALITIKA





# PROCES



# ETL



```
1 query = {}
2 if latest_timestamp:
3     query["timestamp"] = {"$gte": latest_timestamp}
4
5 unprocessed_tickets_cursor = src_tickets_collection.find(query)
6
7 new_tickets = []
8
9 for ticket in unprocessed_tickets_cursor:
10     transformed_ticket = transform_data(ticket)
11     new_tickets.append(transformed_ticket)
12
13 if new_tickets:
14     dest_tickets_collection.insert_many(new_tickets)
```

# STREAMLIT

## STS metrics dashboard

Tickets Agents Customers

From

2023/08/01

To

2024/01/03

Average resolution time

7.41 h

Average pickup time

1.74 h

Average first response time

2.31 h

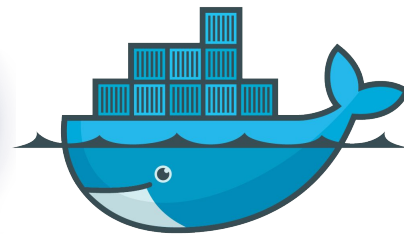
# DEVOPS



# DOCKER I KONTEJNERIZACIJA

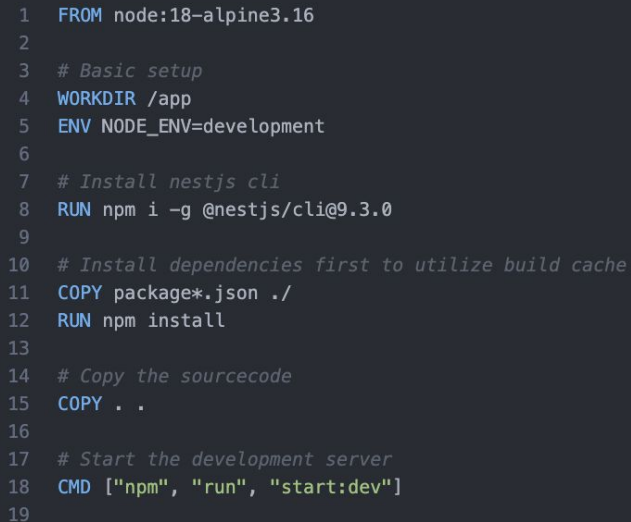
Prirodna potreba za izolacijom okruženja.  
Istorjiski je bilo više faza.

- Zasebni fizički serveri
- Vrituelne mašine
- Kontejneri



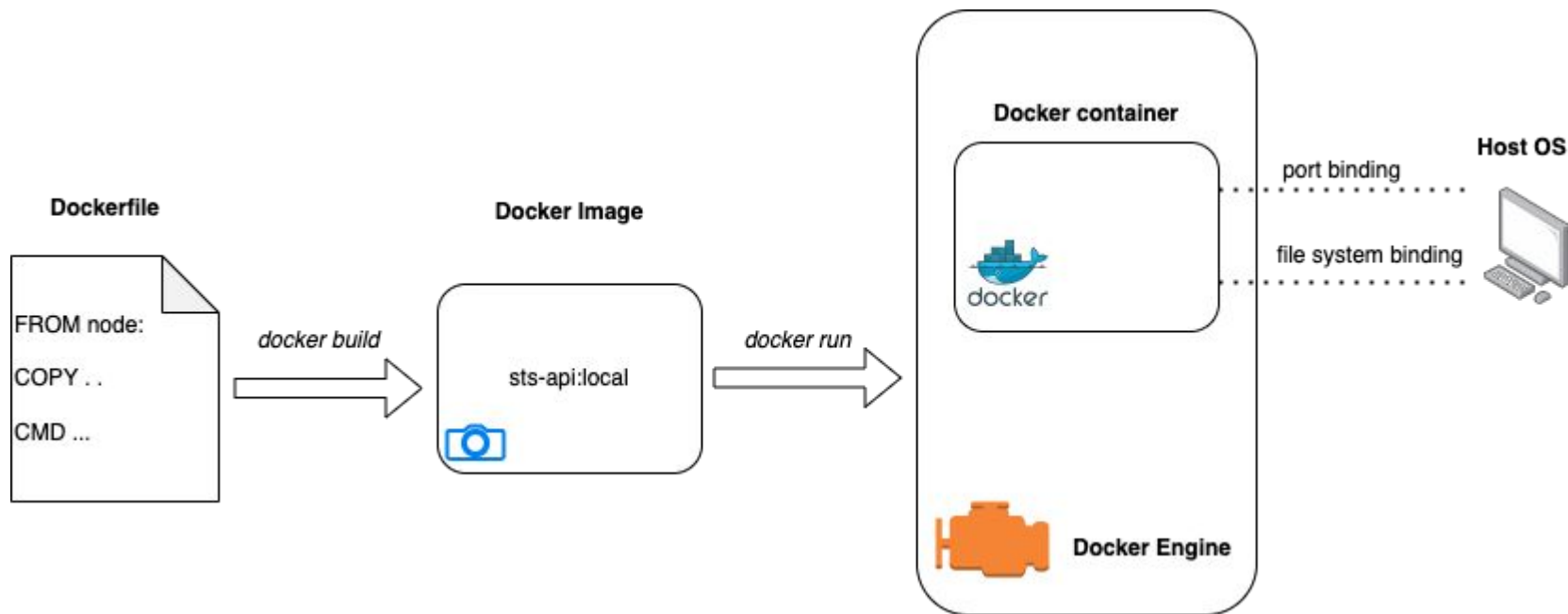
# DOCKER SLIKE

- Deklarativna izgradnja idealne mašine za datu potrebu.
- Koristis sopstvenu sintaksu u takozvanim *Dockerfile*-ovima



```
1 FROM node:18-alpine3.16
2
3 # Basic setup
4 WORKDIR /app
5 ENV NODE_ENV=development
6
7 # Install nestjs cli
8 RUN npm i -g @nestjs/cli@9.3.0
9
10 # Install dependencies first to utilize build cache
11 COPY package*.json ./
12 RUN npm install
13
14 # Copy the sourcecode
15 COPY . .
16
17 # Start the development server
18 CMD ["npm", "run", "start:dev"]
19
```

# DOCKER KONTEJNERI, ORKESTRACIJA



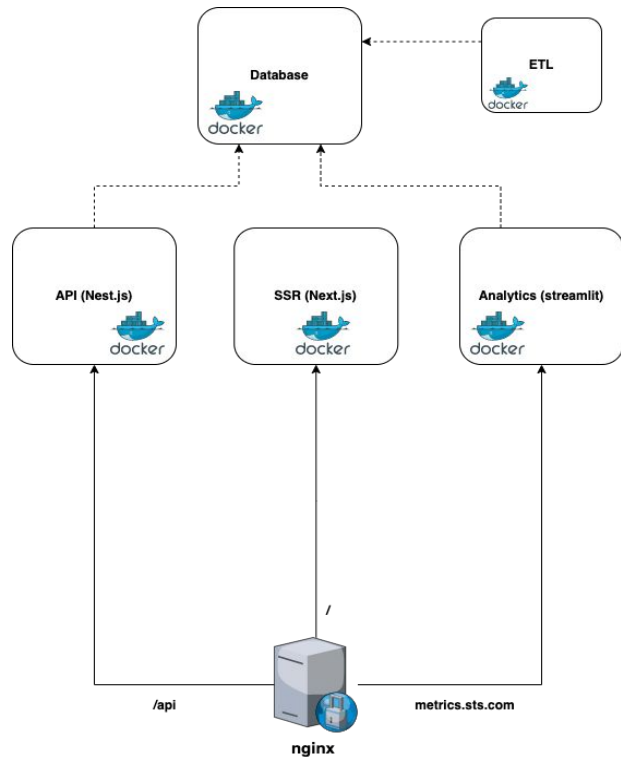
# DOCKER-COMPOSE

```
1 maindb:
2   container_name: main-db-prod
3   image: mongo:latest
4   environment:
5     MONGO_INITDB_ROOT_USERNAME: ${MAIN_DB_USERNAME}
6     MONGO_INITDB_ROOT_PASSWORD: ${MAIN_DB_PWD}
7   ports:
8     - 27017:27017
9   volumes:
10    - main-db-data:/data/db
11   extra_hosts:
12     - "host.docker.internal:host-gateway"
13
14 api_main:
15   container_name: api-main-prod
16   image: aleksakojadinovic/sts:api-latest
17   environment:
18     MAIN_DB_USERNAME: ${MAIN_DB_USERNAME}
19     MAIN_DB_PWD: ${MAIN_DB_PWD}
20     JWT_SECRET: ${JWT_SECRET}
21     FIREBASE_KEY_PATH: "/app/certificates/sts-firebase-key.json"
22   ports:
23     - "3001:3000"
24   extra_hosts:
25     - "host.docker.internal:host-gateway"
```



# NGINX

- Veb server
- Reverzni proksi
- Serviranje statičkog sadržaja
- Load balancing



# PRODUKCIJA, DOCKERHUB

Sve potrebno za pokretanje softvera upakovano je u slike i otpremljeno na dockerhub.

Produkcionsa mašina ne mora klonirati izvorni kod uopšte.



# ZAKLJUČAK, NEDOSTACI

- Prikaz kompletnog razvoja softverskog sistema primenom ustanovljenih praksi, a koristeći moderne tehnologije iz Javascript ekosistema.
- Produkciono okruženje, univerzalnost i moć docker-a.
- Prostor za poboljšanje
  - Testovi
  - Eager loading vs Lazy loading
  - Čist javascript na frontendu
  - Minimalna analitika