

Osnove HTTP-a

POSTMAN

POWERED BY:



Sadržaj

- REST API
- POSTMAN
- POSTMAN Interface

REST

- REST Representational state transfer.
- REST zapravo predstavlja stil web arhitekture i upravlja ponašanjem klijenta i servera.

To znači da će server pozivom na RESTful API vratiti klijentu reprezentaciju stanja zahtjevanog resursa.

Reprezentacija stanja može biti u nekom od formata:

- JSON
- XML
- HTML
- ...

API

- API - **A**pplication **P**rogramming **I**nterface
- Posrednik koji omogućuje komunikaciju između dvije aplikacije.
- Putem API-a , front-end (klijent) i back-end (server) aplikacija zapravo komuniciraju.
- Komunikacija može biti između više različitih back-end aplikacija.

REST API

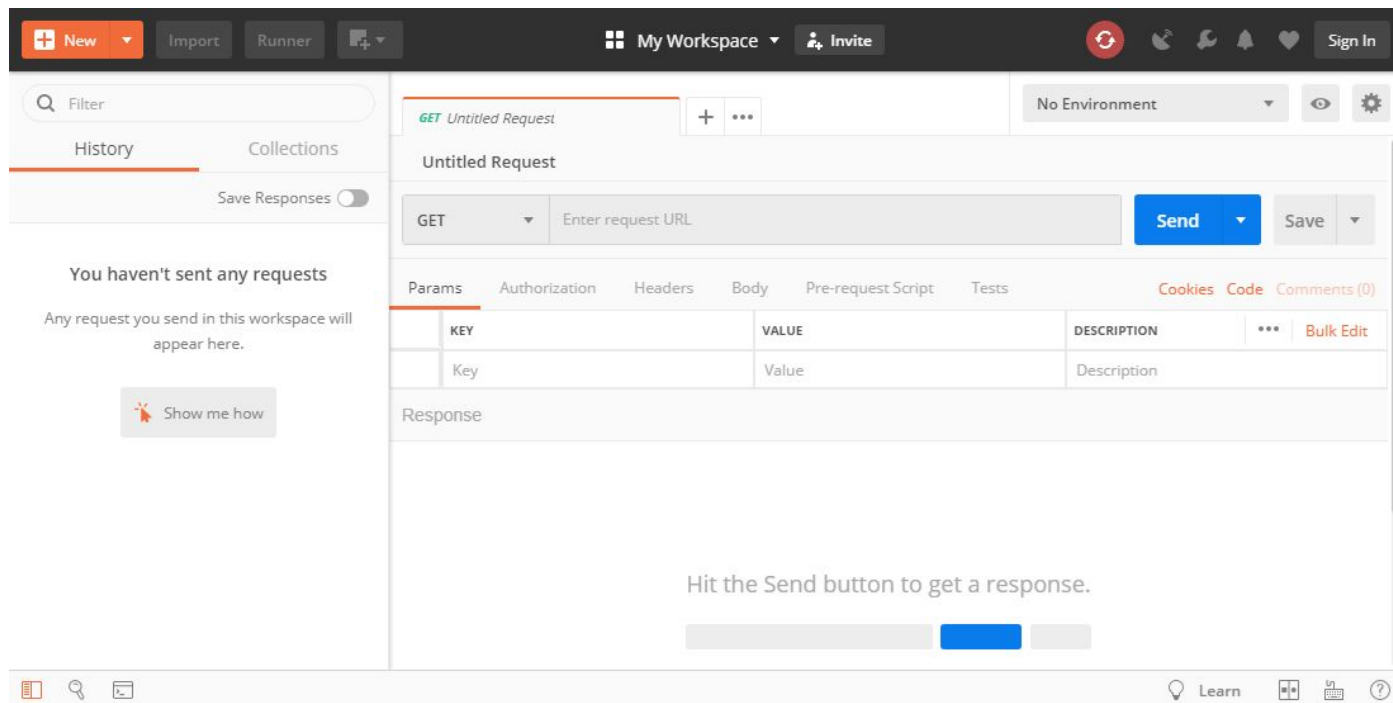
- REST API komunikacija se zasniva na HTTP ili HTTPS protokolu.
- Klijent i Server su neovisni jedan od drugog, što znači da aplikacije mogu biti pisane u različitim programskim jezicima.
- REST API Karakteristike :
 - ❑ **Client-Server** - klijent se brine samo za front-end, back-end se brine samo za server dio, oboje može biti zamijenjeno neovisno jedno od drugog.
 - ❑ **Stateless** - Klijentski podaci nisu spremljeni na serveru između requestova, a sesija se zapravo nalazi na klijentskoj strani.
 - ❑ **Cache** - Klijent može cache-irati response.

POSTMAN

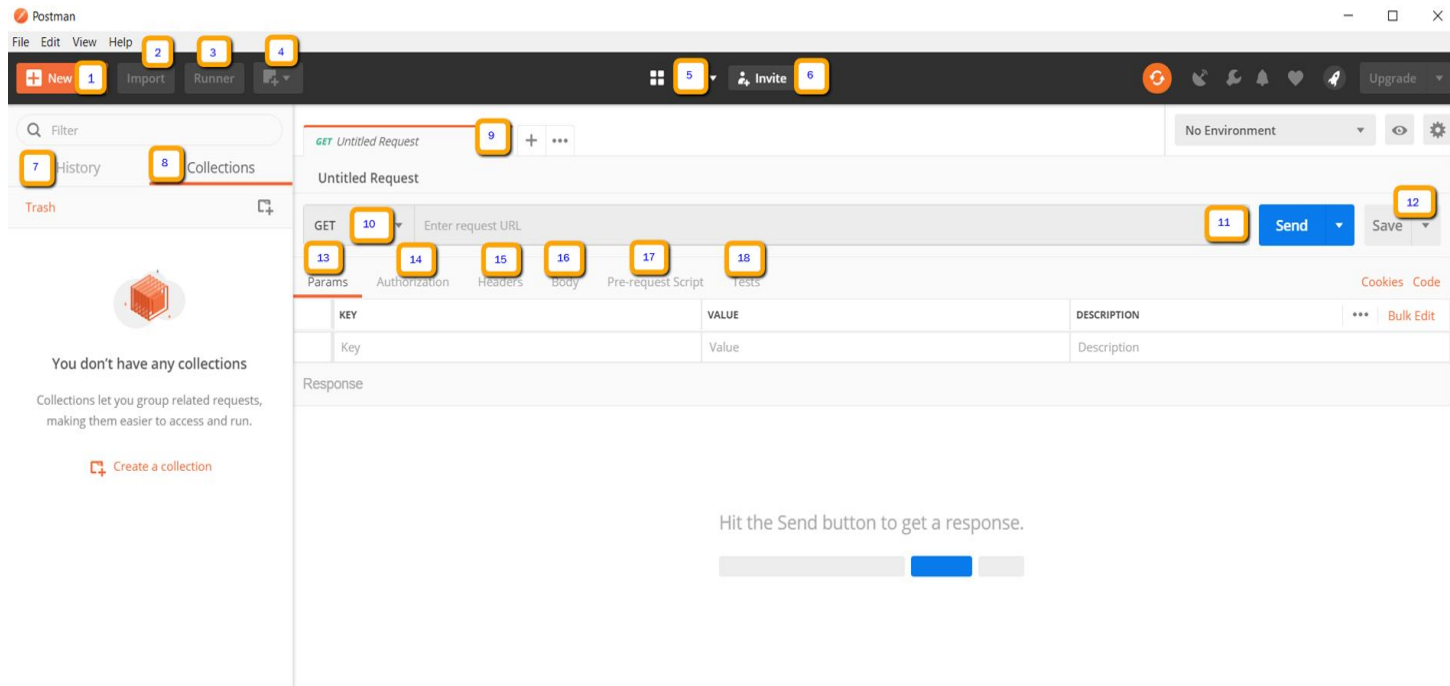
- POSTMAN je alat kreiran za testiranje REST API-a
- Programerima omogućuje da prije same implementacije neke REST API rute , vide kakve točno podatke neka REST API ruta traži , te što će zapravo dobiti ukoliko je response 2xx , a što ako je u pitanju neki drugi response.

[POSTMAN Official Web](#)

POSTMAN Interface



POSTMAN Interface



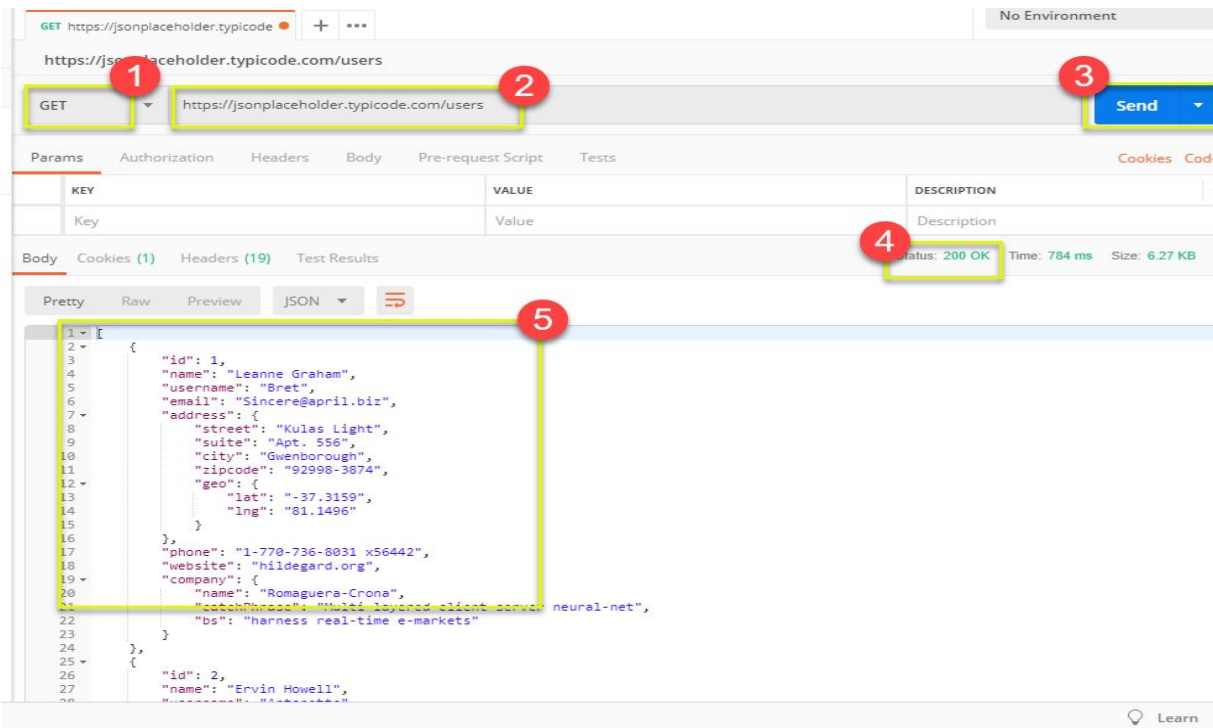
POSTMAN Interface

1. New - Kreira novi request/kolekciju/environment varijablu itd
2. Import - Importa već kreirane requestove
3. Runner - Automatski testovi se mogu izvršiti preko Collection Runner-a
4. Open New - Otvora novi tab
5. My Workspace - Defaultni workspace, a možete kreirati novi također
6. Invite - Pozovite nekog unutar vaseg workspace-a
7. History - Povijest zadnjih requestova
8. Collections - Vaši kreirani requestovi podijeljeni u odgovarajuće kolekcije
9. Request tab - Ime vašeg requesta

POSTMAN Interface

10. HTTP Request - Različite HTTP metode
11. Request URL - API Endpoint putanja
12. Save - Sprema request
13. Params - Query params
14. Authorization - U slučaju da je API zaštićen Auth tokenom , ovdje ga postavljamo.
15. Headers - Postavljamo odgovarajuće headere.
16. Body - U slučaju POST ili PUT metode , u body ubacujemo request podatke.
17. Pre-request Script - Skripte izvršene prije request.
18. Tests - Skripte izvršene za vrijeme request-a

POSTMAN Interface - Rad s GET-om



The screenshot displays the Postman interface for a GET request to `https://jsonplaceholder.typicode.com/users`. The interface is annotated with five red circles and yellow boxes highlighting key elements:

- 1**: The `GET` method dropdown.
- 2**: The URL input field containing `https://jsonplaceholder.typicode.com/users`.
- 3**: The `Send` button.
- 4**: The status bar showing `Status: 200 OK`, `Time: 784 ms`, and `Size: 6.27 KB`.
- 5**: The JSON response body, which is a list of two user objects. The first object is expanded, showing details like `"id": 1`, `"name": "Leanne Graham"`, `"username": "Bret"`, `"email": "Sincere@april.biz"`, and `"address": { "street": "Kulas Light", "suite": "Apt. 556", "city": "Gwenborough", "zipcode": "92998-3874", "geo": { "lat": "-37.3159", "lng": "81.1496" } }`.

```
1 [
2   {
3     "id": 1,
4     "name": "Leanne Graham",
5     "username": "Bret",
6     "email": "Sincere@april.biz",
7     "address": {
8       "street": "Kulas Light",
9       "suite": "Apt. 556",
10      "city": "Gwenborough",
11      "zipcode": "92998-3874",
12      "geo": {
13        "lat": "-37.3159",
14        "lng": "81.1496"
15      }
16    },
17    "phone": "1-770-736-8031 x56442",
18    "website": "hildegard.org",
19    "company": {
20      "name": "Romaguera-Crona",
21      "catchPhrase": "Multi-layered client-server neural-net",
22      "bs": "harness real-time e-markets"
23    }
24  },
25  {
26    "id": 2,
27    "name": "Ervin Howell",
28    "username": "Carter",
29    "email": "Sincere@april.biz",
30    "address": {
31      "street": "P.O. Box 6459, Reno, NV 89506",
32      "suite": "Apt. 950",
33      "city": "Reno",
34      "zipcode": "89506-9129",
35      "geo": {
36        "lat": "39.5354",
37        "lng": "-119.8429"
38      }
39    },
40    "phone": "(760) 291-7067",
41    "website": "eltonjohnsophisticated.com",
42    "company": {
43      "name": "Casper Group",
44      "catchPhrase": "Innovative two-tiered extranet",
45      "bs": "e-enable e-business"
46    }
47  }
48 ]
```

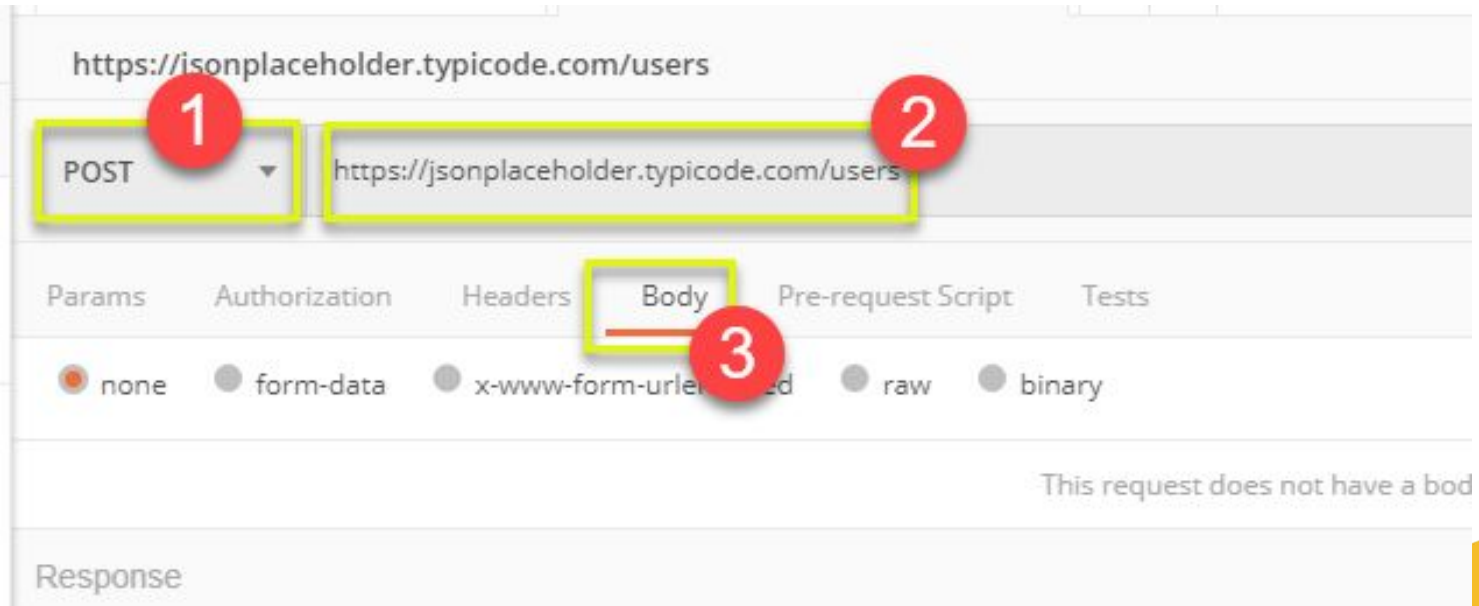
POSTMAN Interface - Rad s GET-om

- 1. Postavite HTTP metodu na GET
- 2. U request URL ubacite vas endpoint.
- 3. Kliknite pošalji
- 4. Trebali bi vidjeti da je sve u redu

HINT: za request URL mozete koristiti API:

- <https://school-api.spark.ba/v1/pizzas>

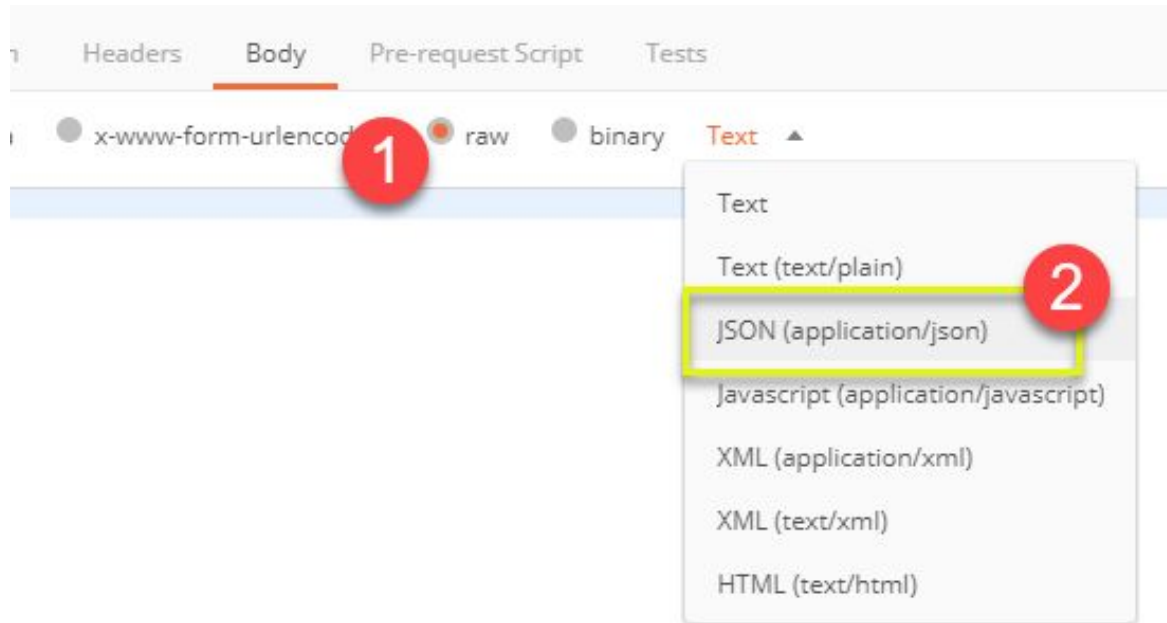
POSTMAN Interface - Rad s POST-om



POSTMAN Interface - Rad s POST-om

- 1. Postavite HTTP metodu na POST.
- 2. Postavite request URL .
- 3. Prebacite se na Body tab.

POSTMAN Interface - Rad s POST-om



POSTMAN Interface - Rad s POST-om

- 1. Kliknite na "raw"
- 2. Select JSON
- 3. Te unutar radne površine prozora dodajte vaše podatke.
- 4. Kliknite SEND za slanje requesta.

Zadatak 1.0

Kreirajte novi korisnicki racun putem sljedećeg API endpoint-a:

→ <https://school-api.spark.ba/v1/register>

→ Metoda POST

→ Payload aka request body

```
{  
    "username": "franjo",  
    "password": "123",  
    "first_name": "franjo",  
    "last_name": "dankic",  
    "age": 12  
}
```

Zadatak 1.1

Logirajte se putem sljedećeg API endpoint-a

- <https://school-api.spark.ba/v1/login>
- Metoda POST
- Payload aka request body

```
{  
    "username": "franjo",  
    "password": "123"  
}
```

HINT : Kad dobijete response kopirajte vrijednost propertya “token”

Zadatak 1.2

Zatražite pregled svih članaka , putem sljedećeg API endpoint-a

- <https://school-api.spark.ba/v1/posts/all>
- Metoda **GET**
- Postavite novi header unutar **HEADERS** tab-a

Authorization : spremljeni token

Zadatak 1.3

Kreirajte novi članak putem sljedećeg API endpoint-a

→ <https://school-api.spark.ba/v1/post>

→ Metoda **POST**

→ **Payload** aka Request body

```
{  
    "title": "Test",  
    "content": "Lorem ipsum"  
}
```

→ Postavite novi header unutar **HEADERS** tab-a
Authorization : *spremljeni token*

Zadatak 1.4

Dohvatite samo svoje članke preko sljedećeg API endpoint-a

→ <https://school-api.spark.ba/v1/posts/me>

→ Metoda **GET**

Postavite novi header unutar **HEADERS** tab-a

Authorization : *spremljeni token*

Zadatak 1.5

Ažurirate postojeći članak , putem sljedećeg API endpoint-a

- <https://school-api.spark.ba/v1/post/{ID}>
- Metoda **PUT** ({ID} označava dinamički dodijeljen ID prilikom kreiranja samog članka, zamijenite {ID} s odgovarajućim post_id-em).
- Payload aka Request Body

```
{  
    "title": "Test update",  
    "content": "PUT na rutu"  
}
```

Postavite novi header unutar **HEADERS** tab-a

Authorization : *spremljeni token*

Zadatak

Nakon što ste odradili i zadnju PUT metodu , ponovo pogodite API za dohvat svih Vasih članaka.

Ono sto bi trebao biti rezultat akcije , jest novo stanje ukoliko ste ga ispravno promijenili.

POWERED BY:



Pitanja ?

POWERED BY:



THANK YOU
for attention