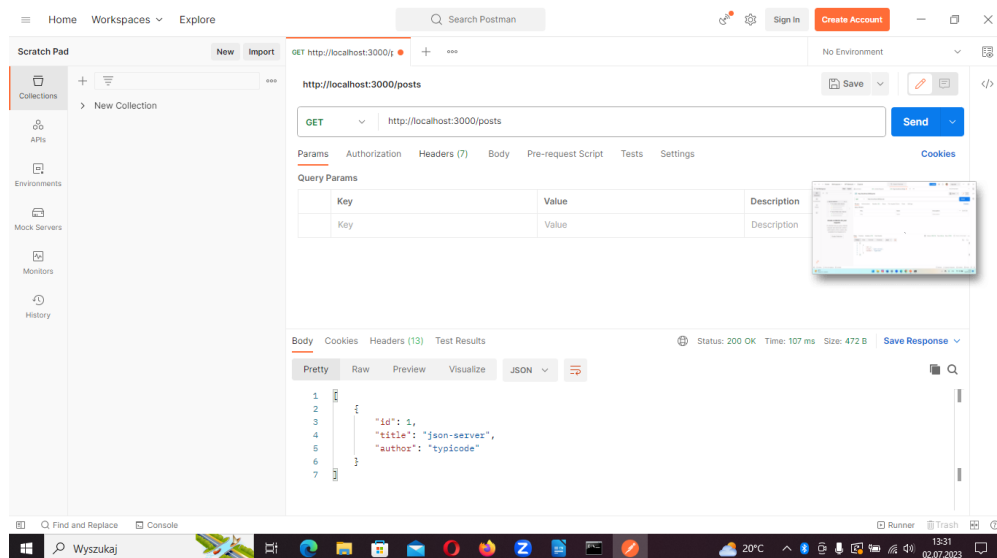


# Przykładowe zadania z użyciem POSTMAN

**1. Pobieranie wszystkich postów** – ustawiam metodę get w polu ENTER URL or paste wpisuję adres mojego localhosta którego testuję <http://localhost:3000/>

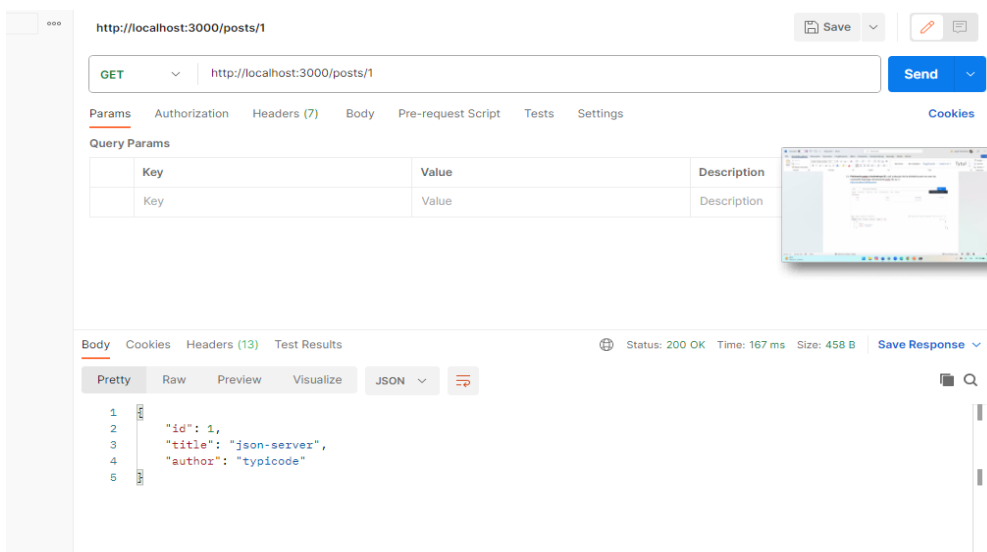
dodaje końcówkę post bo chcę zobaczyć posty <http://localhost:3000/posts>

Po uzupełnieniu adresu i metody naciskam SEND a na dole otrzymuje odpowiedz mojego serwera który ma wyświetlić wszystkie posty

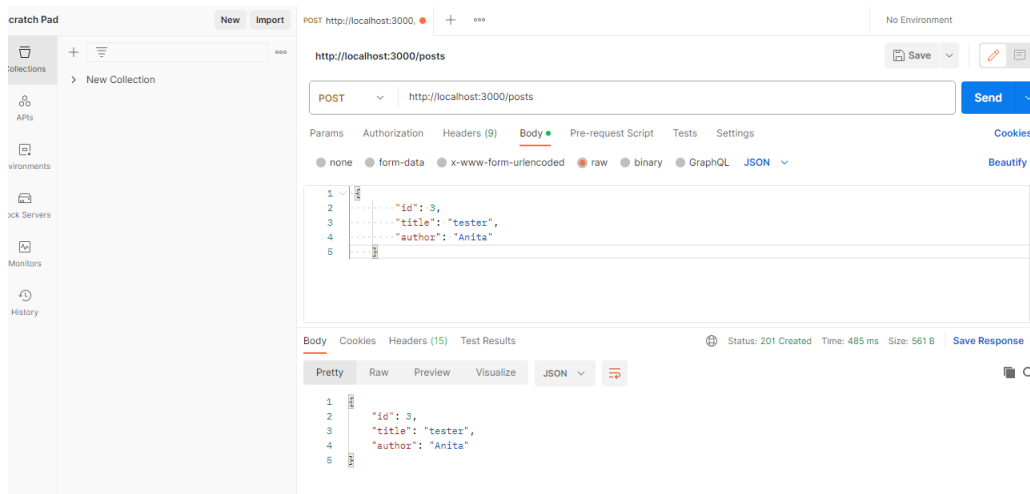


**2. Pobieranie posta z konkretnym ID** – czyli wskazuję który konkretnie post ma nam się wyświetlić dopisując do końcówki posts ID np.1

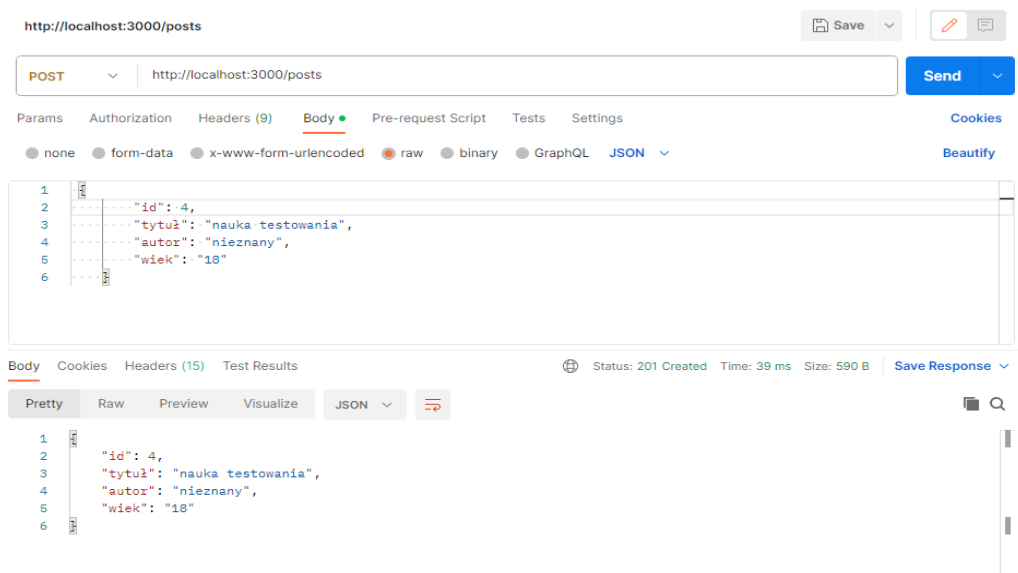
<http://localhost:3000/posts/1>



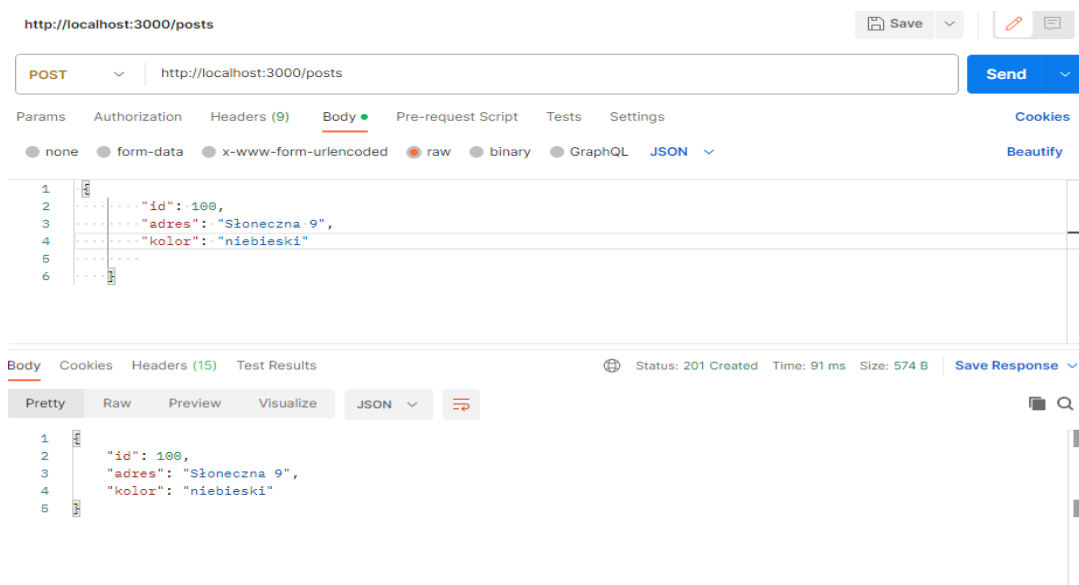
## Tworzenie nowego posta – metodą post



Tworzymy nowego posta ID 4 tytuł „nauka testowania” autor nie znany , wiek 18

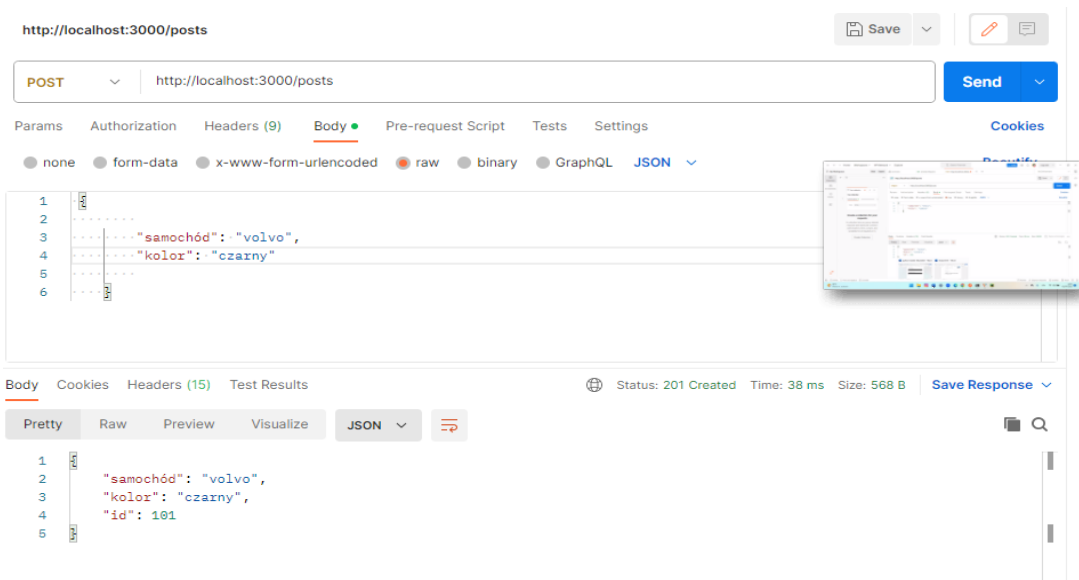


Tworzymy nowego posta adres Słoneczna 9, kolor niebieski ID 100



Uwaga

Co się stanie jeżeli stworzę nowego posta bez adresu ID o następującej zawartości samochód volvo, kolor czarny

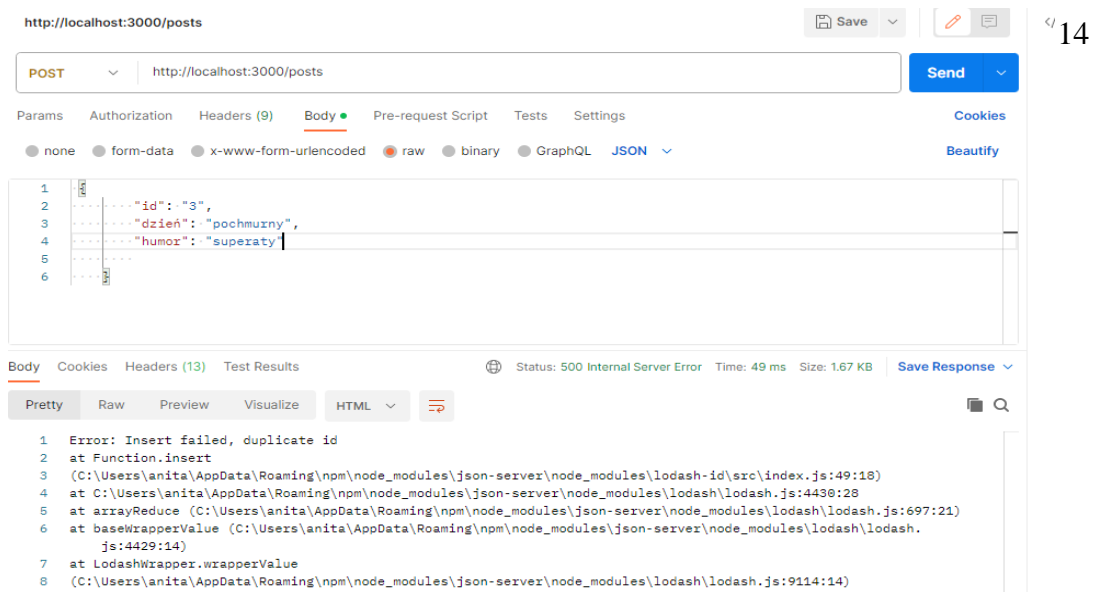


## Tworzy nowy numer ID na podstawie ostatniego utworzonego id posta

### Uwaga

Co się stanie jeżeli będę próbować stworzyć nowego posta o id który już istnieje w serwerze np. ID 3 i treści dzień pochmurny, humor superaty

Otrzymuje status 500 informujący że taki post o takim ID już istnieje a metoda post tylko tworzy a nie uaktualnia czy modyfikuje posty, do tego służą inne metody



14

```
POST http://localhost:3000/posts

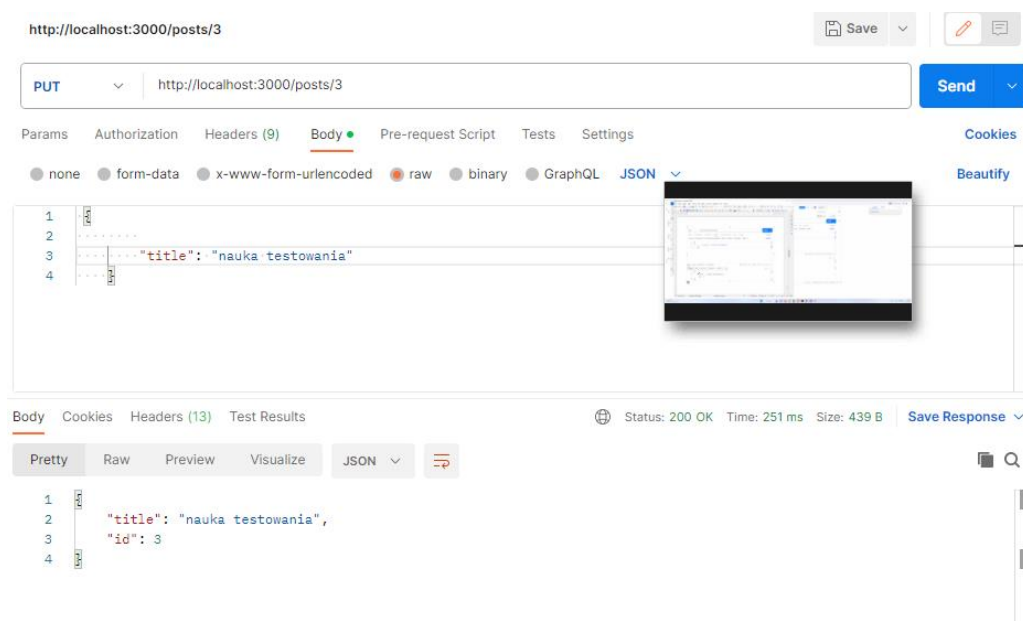
{
  "id": "3",
  "dzień": "pochmurny",
  "humor": "superaty"
}
```

Status: 500 Internal Server Error Time: 49 ms Size: 1.67 KB

```
1 Error: Insert failed, duplicate id
2 at Function.insert
3 (C:\Users\anita\AppData\Roaming\npm\node_modules\json-server\node_modules\lodash-id\src\index.js:49:18)
4 at C:\Users\anita\AppData\Roaming\npm\node_modules\json-server\node_modules\lodash\lodash.js:4438:28
5 at arrayReduce (C:\Users\anita\AppData\Roaming\npm\node_modules\json-server\node_modules\lodash\lodash.js:697:21)
6 at baseWrapperValue (C:\Users\anita\AppData\Roaming\npm\node_modules\json-server\node_modules\lodash\lodash.js:4429:14)
7 at LodashWrapper.prototype.value
8 (C:\Users\anita\AppData\Roaming\npm\node_modules\json-server\node_modules\lodash\lodash.js:9114:14)
```

## Aktualizacja postów metodą PUT

W ID aktualizuje tytuł na nauka testowania



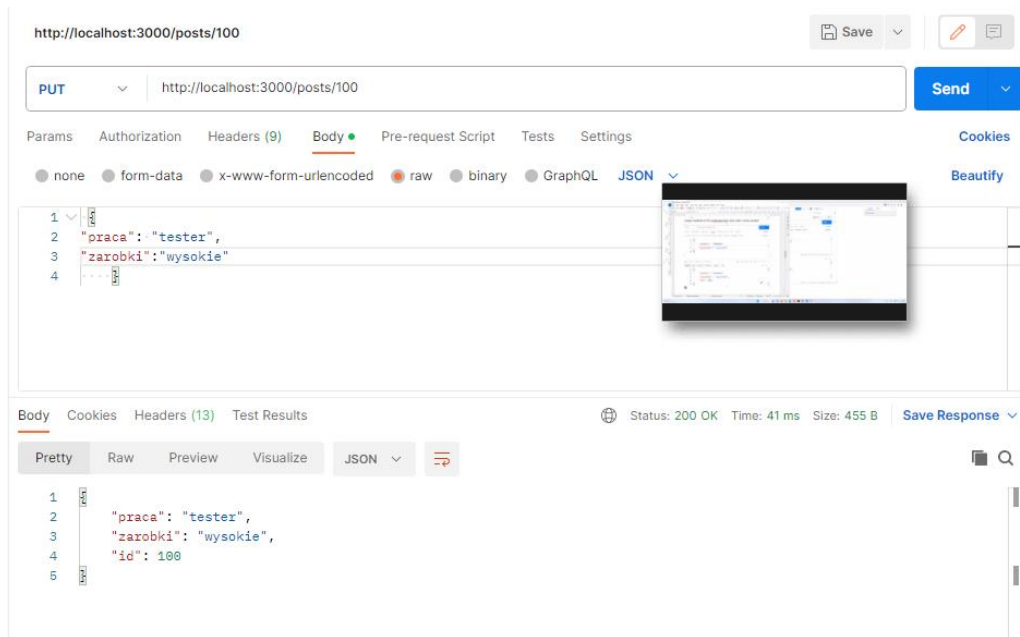
```
PUT http://localhost:3000/posts/3

{
  "title": "nauka testowania",
  "id": 3
}
```

Status: 200 OK Time: 251 ms Size: 439 B

```
1 {
2   "title": "nauka testowania",
3   "id": 3
4 }
```

Modyfikuje ID 100 o następująca treść: praca tester, zarobki wysokie



http://localhost:3000/posts/100

PUT http://localhost:3000/posts/100

Params Authorization Headers (9) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "praca": "tester",
3   "zarobki": "wysokie"
4 }
```

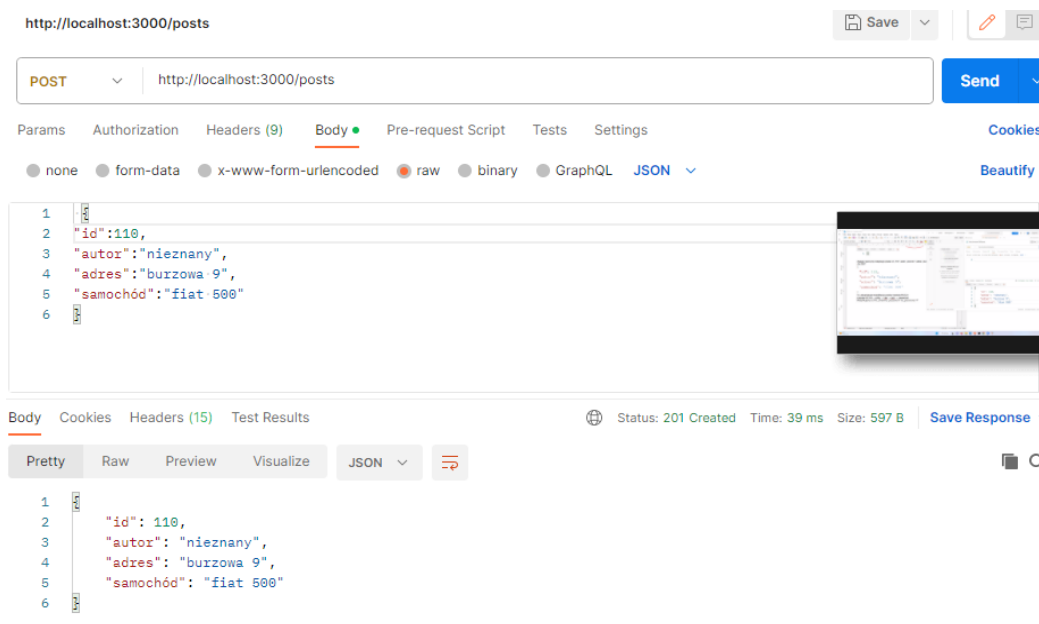
Body Cookies Headers (13) Test Results

Status: 200 OK Time: 41 ms Size: 455 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "praca": "tester",
3   "zarobki": "wysokie",
4   "id": 100
5 }
```

Tworzymy nowego posta ID 110, autor anonim, adres burzowa 9, samochód fiat 500



http://localhost:3000/posts

POST http://localhost:3000/posts

Params Authorization Headers (9) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "id": 110,
3   "autor": "nieznany",
4   "adres": "burzowa 9",
5   "samochód": "fiat 500"
6 }
```

Body Cookies Headers (15) Test Results

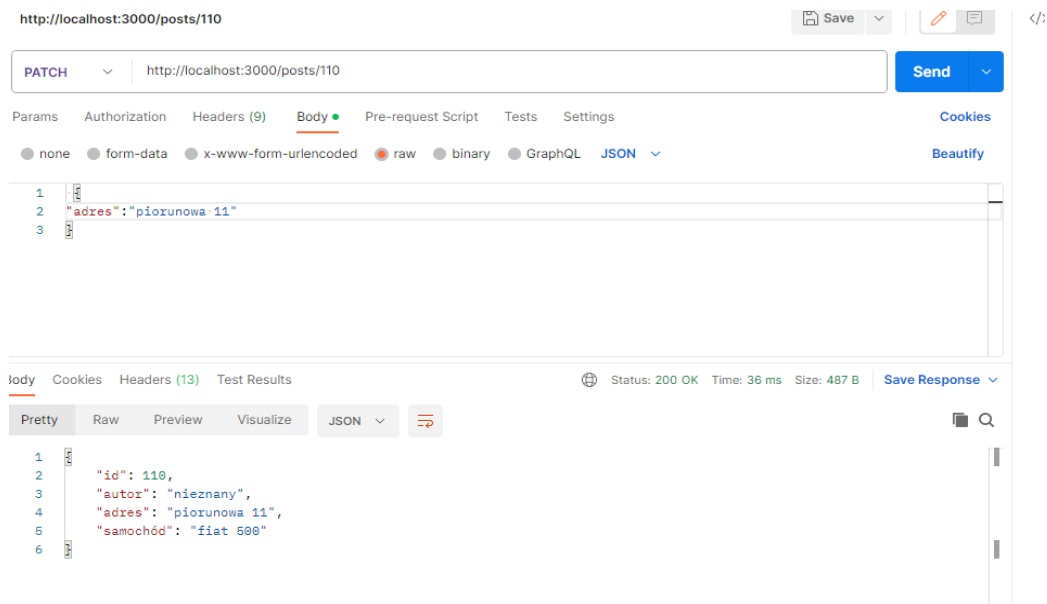
Status: 201 Created Time: 39 ms Size: 597 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 110,
3   "autor": "nieznany",
4   "adres": "burzowa 9",
5   "samochód": "fiat 500"
6 }
```

## Aktualizacja – modyfikacja metoda patch

Modyfikujemy id 110 zmieniamy burzowa 9 na piorunowa 11



http://localhost:3000/posts/110

PATCH http://localhost:3000/posts/110

Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

1  
2 "adres": "piorunowa 11"  
3

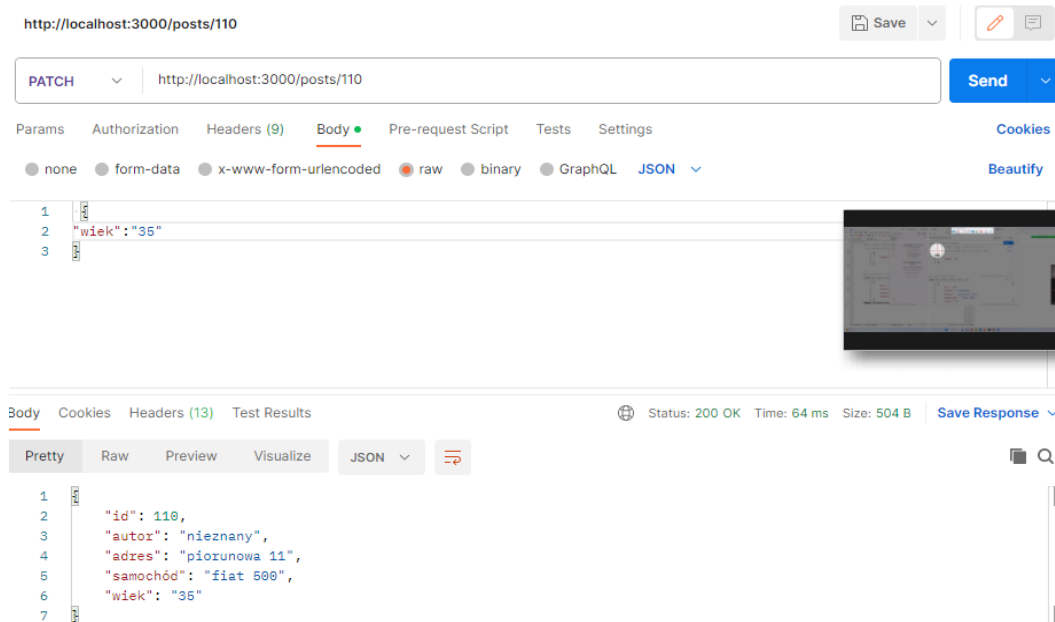
Body Cookies Headers (13) Test Results

Status: 200 OK Time: 36 ms Size: 487 B Save Response

Pretty Raw Preview Visualize JSON

1  
2 "id": 110,  
3 "autor": "nieznany",  
4 "adres": "piorunowa 11",  
5 "samochód": "fiat 500"  
6

Modyfikuje metoda PATCH ID 110 o następujące wartości wiek 35



http://localhost:3000/posts/110

PATCH http://localhost:3000/posts/110

Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

1  
2 "wiek": "35"  
3

Body Cookies Headers (13) Test Results

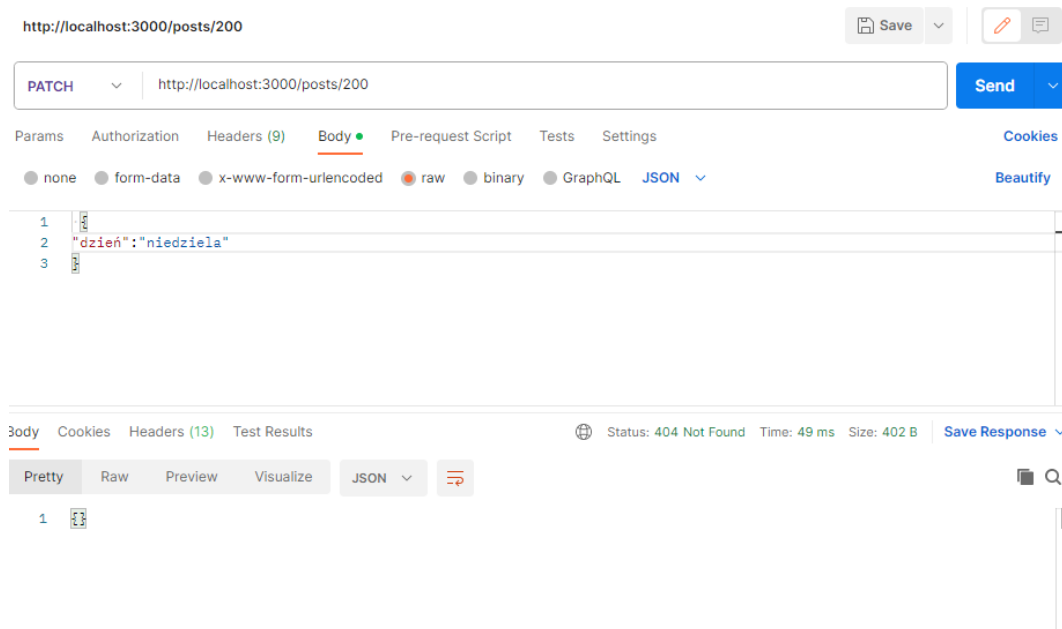
Status: 200 OK Time: 64 ms Size: 504 B Save Response

Pretty Raw Preview Visualize JSON

1  
2 "id": 110,  
3 "autor": "nieznany",  
4 "adres": "piorunowa 11",  
5 "samochód": "fiat 500",  
6 "wiek": "35"  
7

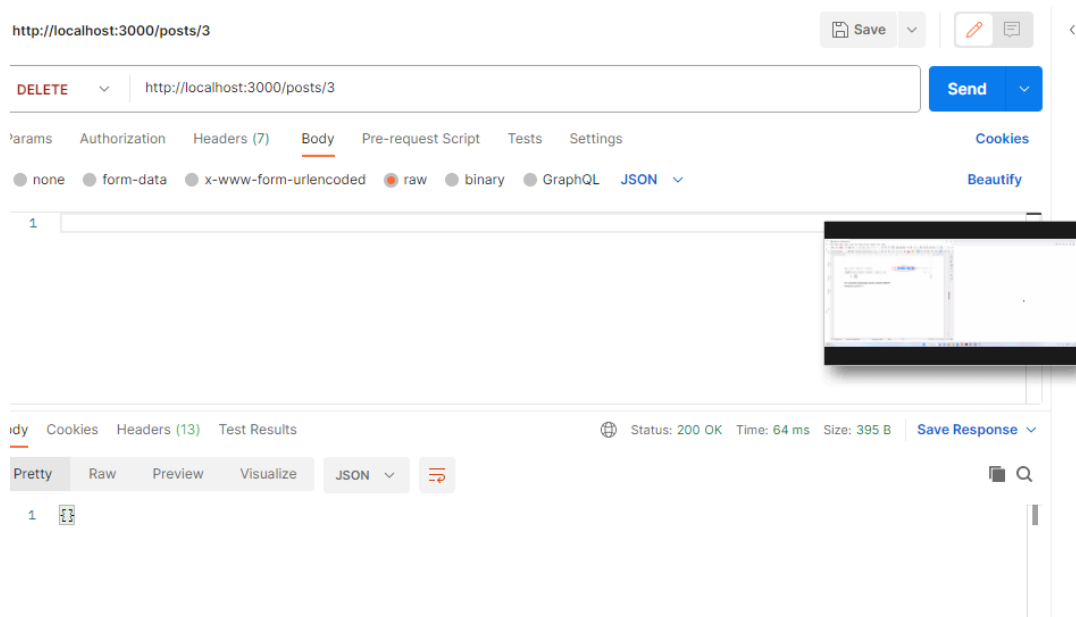
Aktualizuje metoda PATCH posta ID 200 który nie istnieje o następujące dane  
dzień niedziela

Status : 404Not Found



## Usuwanie danego posta metoda DELETE

Usuwamy ID 3



Filtrowanie wybranych elementów z konkretnego serwera i z konkretnego posta

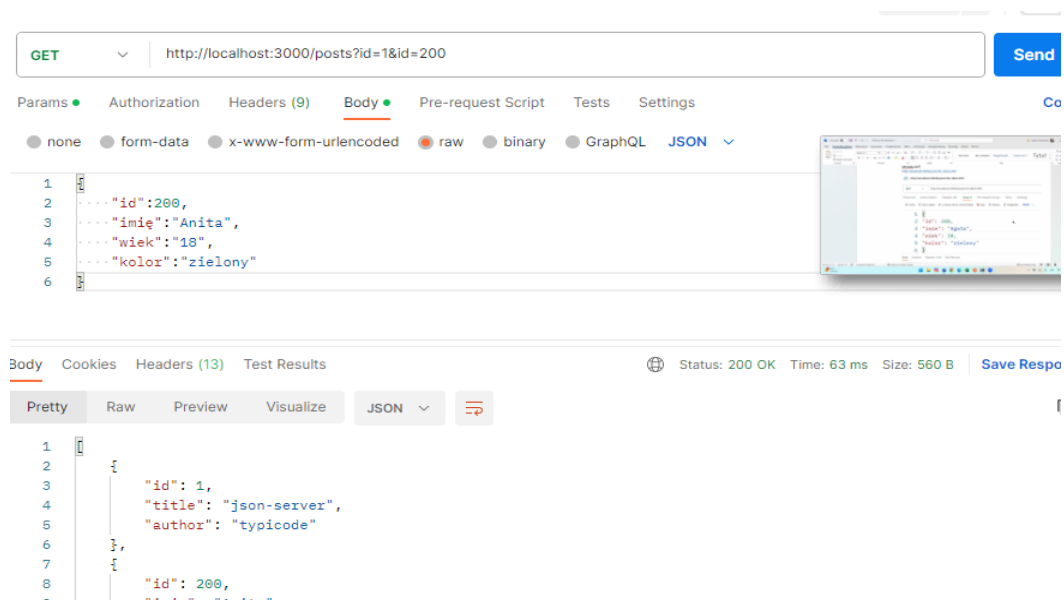
-Filtrowanie po poście i konkretnym ID np. ID 4

Get <http://localhost:3000/posts?id=4>



Filtrowanie dwóch postów

id 1 i id 200





Filtrowanie po tytule np. kolor, czyli chce żeby wyfiltrował wszystkie posty które zawierają kolor zielony

The screenshot shows a REST client interface with a GET request to `http://localhost:3000/posts?kolor=zielony`. The request body is a JSON object: `{ "id": 200, "imie": "Anita", "wiek": "18", "kolor": "zielony" }`. The response status is 200 OK, and the response body is the same JSON object, displayed in a pretty-printed format.

```
GET http://localhost:3000/posts?kolor=zielony
```

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "id": 200,
3   "imie": "Anita",
4   "wiek": "18",
5   "kolor": "zielony"
6 }
```

body Cookies Headers (13) Test Results Status: 200 OK Time: 111 ms Size: 484 B Save Response

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "id": 200,
4     "imie": "Anita",
5     "wiek": "18",
6     "kolor": "zielony"
7   }
8 ]
```