ЛУНАПАРКОВА ЛУДНИЦА - КОМАНДНО ТАБЛО

КАЗУС 2

ДОКУМЕНТАЦИЯ

1. ТЕРМИНОЛОГИЯ, КОЯТО СЕ СРЕЩА В ДОКУМЕНТА:

- URL – адреса, чрез който се достъпва дадения ресурс. Винаги започва с localhost:5000 или 127.0.0.1:5000. (localhost:5000/attractions/4)

- Метод – вид на заявката – дали е за създаване на нов запис (POST), редакция на съществуващ запис (PUT), изтриване на съществуващ запис (DELETE), или извличане на съществуващ запис (GET).

- Сървърен отговор – отговора, който сървъра връща, спрямо заявката, която е получила:
    - 200 / OK – заявката е успешно обработена.
    - 201 / Created – заявката е успешна; като резултат е създаден нов ресурс.
    - 404 / Not Found– ресурсът не е намерен.
    - 500 / Internal Server Error – заявка, която не може да се обработи – грешка от сървърна страна.

2. ОПИСАНИЕ НА ENDPOINTS

URL: localhost:5000 +	**МЕТОД**	**ОПИСАНИЕ**	**СЪРВЪРЕН ОТГОВОР**
/attractions	GET	Извежда списък с всички атракциони	200 OK: JSON масив с атракциони *или* 404 Not Found: когато няма никакви записи на атракциони в базата
/attractions/<id> (/attractions/5)	GET	Извежда данни за атракцион, в зависимост от идентификатора му. В примера сървъра ще върне данни за атракцион с идентификатор 5.	200 OK: JSON обект с данни за атракционна *или* 404 Not Found: атракцион с този идентификатор не съществува в базата
/employees	GET	Извежда списък с всички служители	200 OK: JSON масив със служители *или* 404 Not Found: когато няма никакви записи на служители в базата

/employees/<id> (/employees/2)	GET	Извежда данни за служител, в зависимост от идентификатора му. В примера сървъра ще върне данни за служител с идентификатор 2.	200 OK: JSON обект с данни за служителя *или* 404 Not Found: служител с този идентификатор не съществува в базата
/sales	GET	Извежда списък с всички продажби	200 OK: JSON масив със продажби *или* 404 Not Found: когато няма никакви записи на продажби в базата
/sales/<id> (/sales/9	GET	Извежда данни за продажба, в зависимост от идентификатора му. В примера сървъра ще върне данни за служител с идентификатор 9.	200 OK: JSON обект с данни за продажбата *или* 404 Not Found: продажба с този идентификатор не съществува в базата
/tickets	GET	Извежда списък с всички билети	200 OK: JSON масив със билети *или* 404 Not Found: когато няма никакви записи на билети в базата
/tickets/<id> (/tickets/3)	GET	Извежда данни за билет, в зависимост от идентификатора му. В примера сървъра ще върне данни за билет с идентификатор 3.	200 OK: JSON обект с данни за билет *или* 404 Not Found: билет с този идентификатор не съществува в базата

/tickets	POST	Създава нов билет и го запазва в базата. Очакваният формат на входните параметри е JSON и е следния:
		`{`
		`"PRICE": float,`
		`"PROMOTION_ID": int,`
		`"STATUS": "string",`
		`"TYPE": "string",`
		`"VALIDITY_START": "YYYY-MM-DD",`
		`"VALIDITY_END":"YYYY-MM-DD"`
		`}`
			201 Created: Билетът е създаден
			или
			500 Internal Server Error: билетът не е създаден, поради грешка

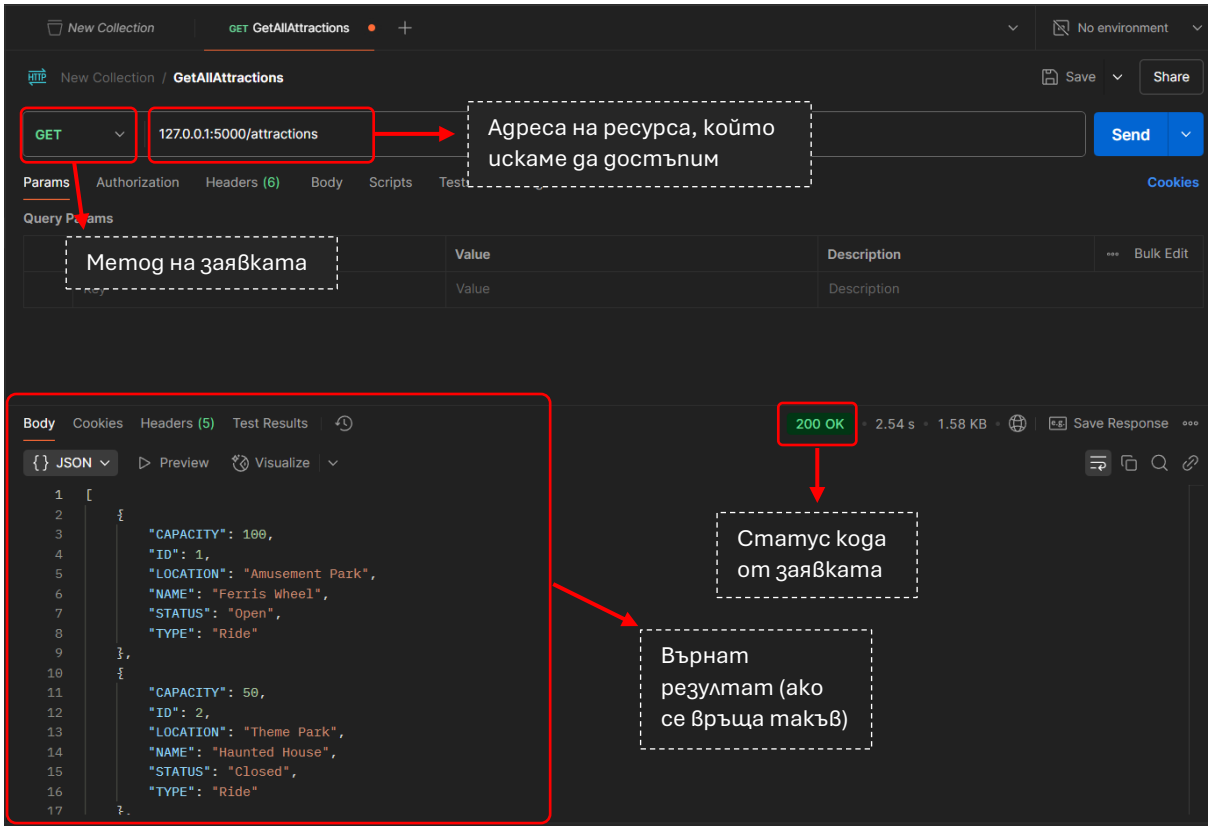
/events	GET	Извежда списък с всички специални събития.	200 OK: JSON масив със събития. *или* 404 Not Found: когато няма никакви записи на събития в базата
/events/<id> (/events/1)	GET	Извежда данни за събитие, в зависимост от идентификатора му. В примера сървъра ще върне данни за събитие с идентификатор 1.	200 OK: JSON обект с данни за събитие *или* 404 Not Found: събитие с този идентификатор не съществува в базата
/visitors	GET	Извежда списък с всички посетители.	200 OK: JSON масив с посетители. *или* 404 Not Found: когато няма никакви записи на посетители в базата
/visitors/<id> (/visitors/4)	GET	Извежда данни за посетител, в зависимост от идентификатора му. В примера сървъра ще върне данни за посетител с идентификатор 4.	200 OK: JSON обект с данни за посетител *или* 404 Not Found: посетител с този идентификатор не съществува в базата

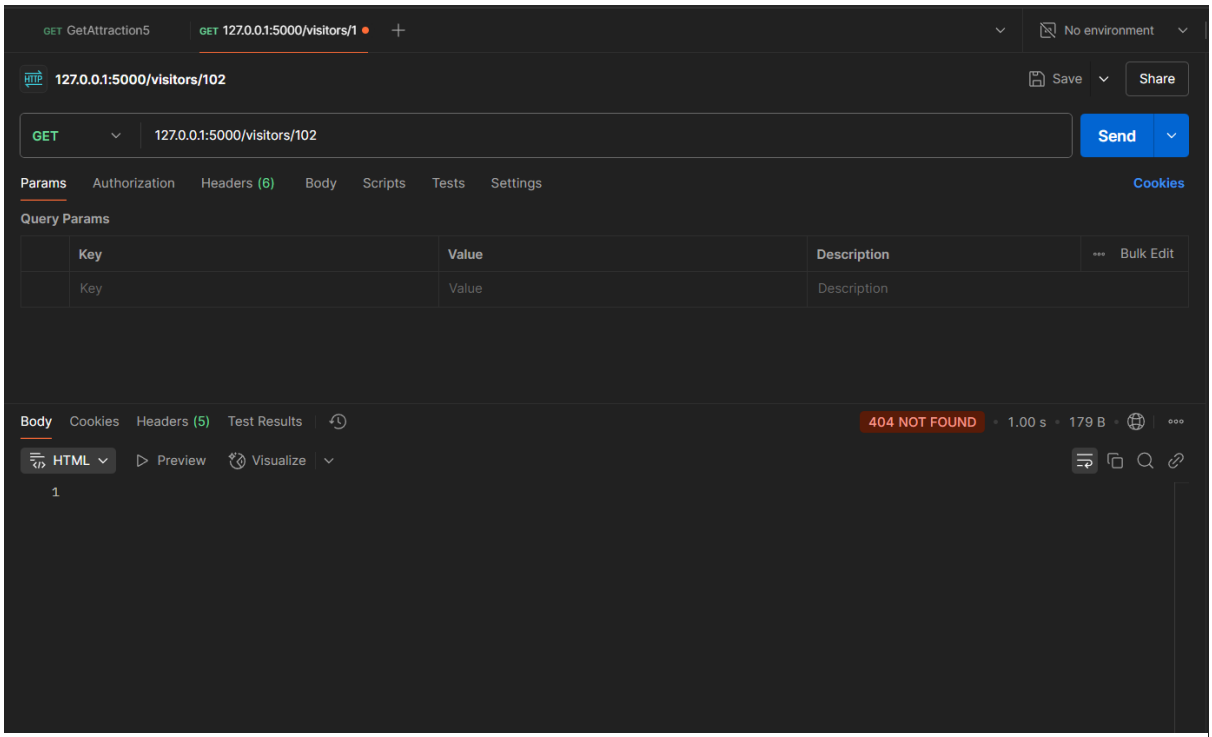
/visitors	POST	Създава нов билет и го запазва в базата. Очакваният формат на входните параметри е JSON и е следния:
		{
		"NAME":"string",
		"EMAIL":"string",
		"PHONE":"string",
		}
			201 Created: Посетителят е създаден
			или
			500 Internal Server Error: посетителят не е създаден, поради грешка

/visitors/<id>
(/visitors/4)
	PUT	Редактира съществуващ запис на посетител. Приема JSON:
		{
		"EMAIL": "sara@williams.com",
		"NAME": "Sarah Williams",
		"PHONE": "2030-4050"
		}
			200 OK: Обектът е създаден.
			или
			404 Not Found: посетител с този идентификатор не съществува в базата
			или
			500 Internal Server Error: заявката не може да се обработи, сървърна грешка

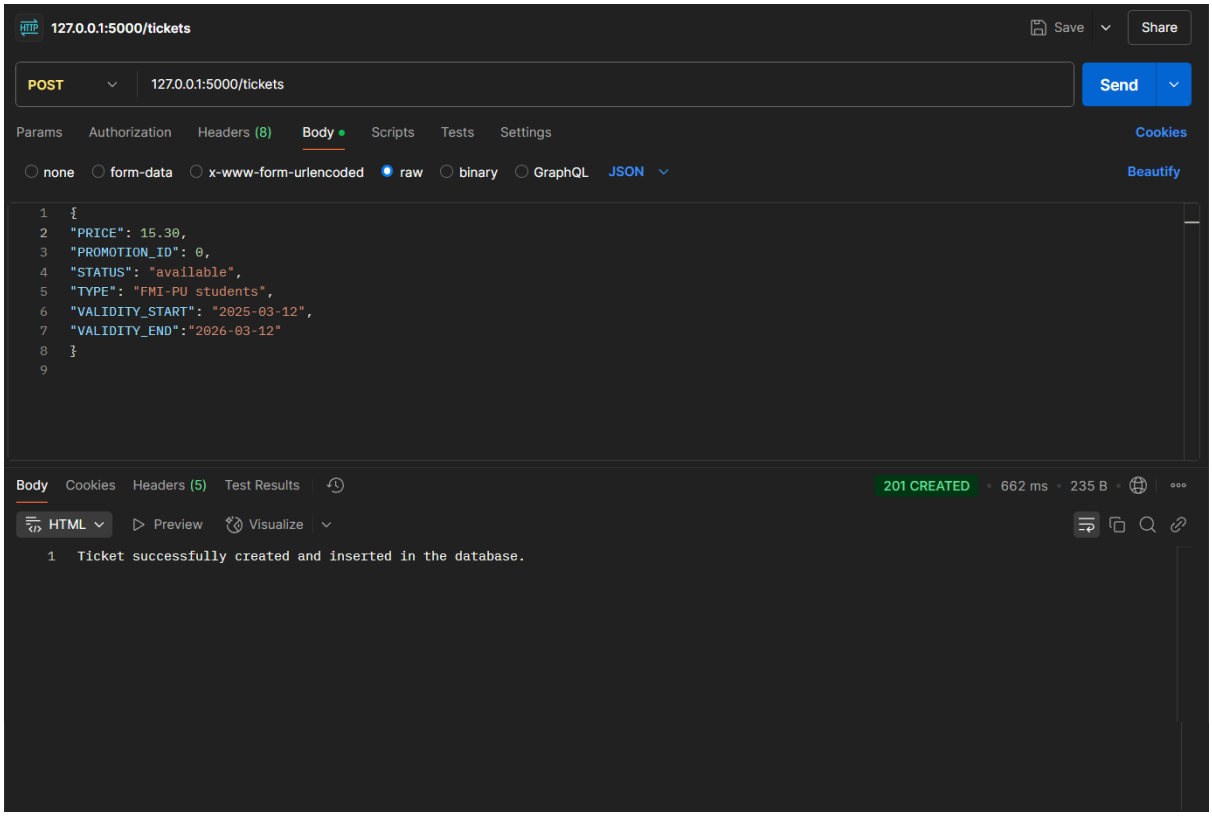
/visitors/<id> (/visitors/101)	DELETE	Изтрива съществуващ запис на посетител. Записът, който ще се изтрива се идентифицира с id.	200 OK: Обектът е изтрит успешно *или* 404 Not Found: посетител с този идентификатор не съществува в базата
/transactions	GET	Извежда списък с всички транзакции.	200 OK: JSON масив с транзакции. *или* 404 Not Found: когато няма никакви записи на посетители в базата
/transactions/<id> (/transactions/6)	GET	Извежда данни за транзакция, в зависимост от идентификатора му. В примера сървъра ще върне данни за транзакция с идентификатор 4.	200 OK: JSON обект с данни за транзакция *или* 404 Not Found: транзакция с този идентификатор не съществува в базата

3. **POSTMAN ЗАЯВКИ** (Тестване на системата)

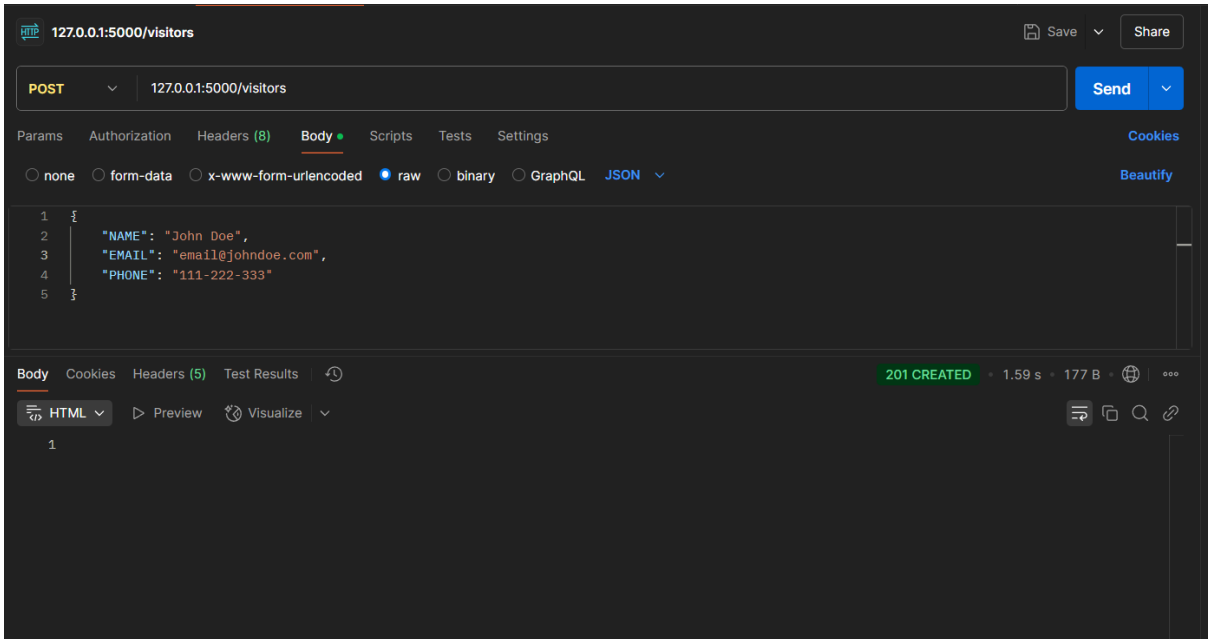




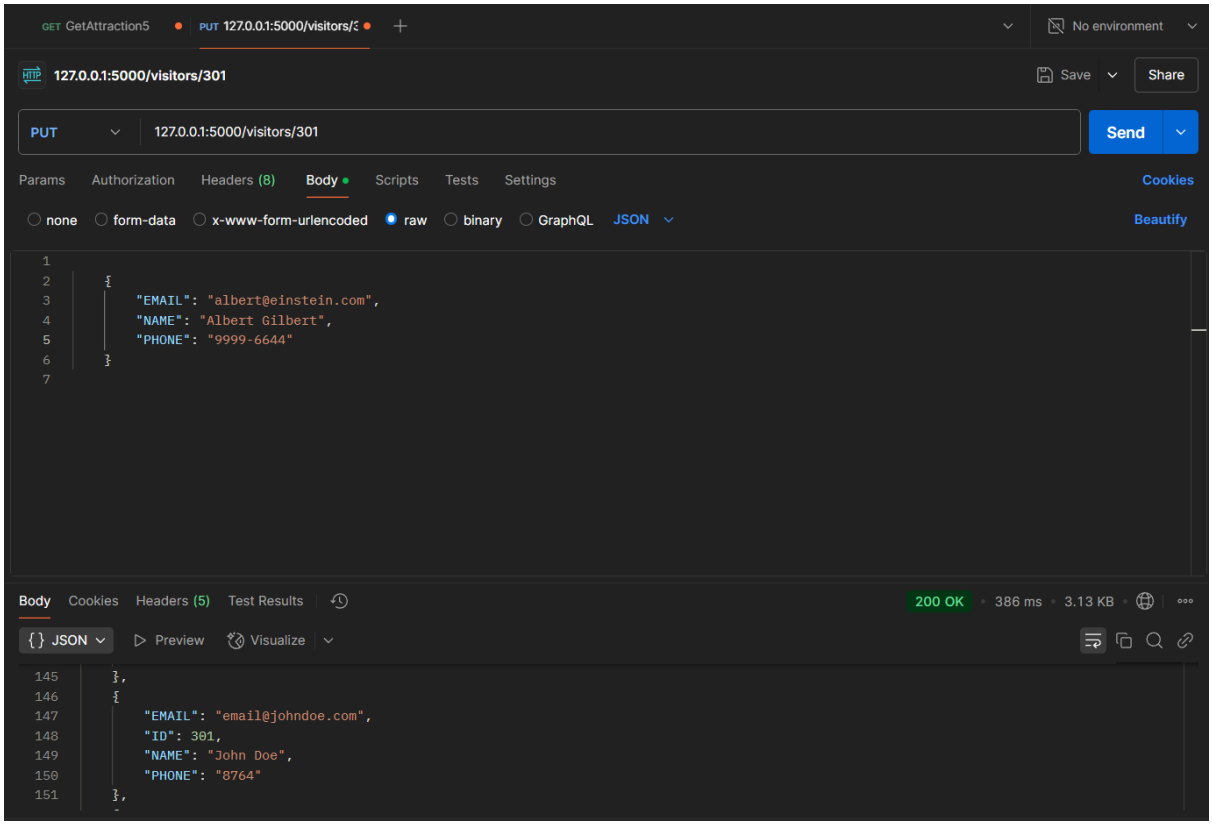
Фигура 1. Правим заявка за данните на посетител с ID 102. Такъв посетител няма! Сървъра връща код 404 – ресурсът не е намерен. Респективно – нямаме данни в JSON формат



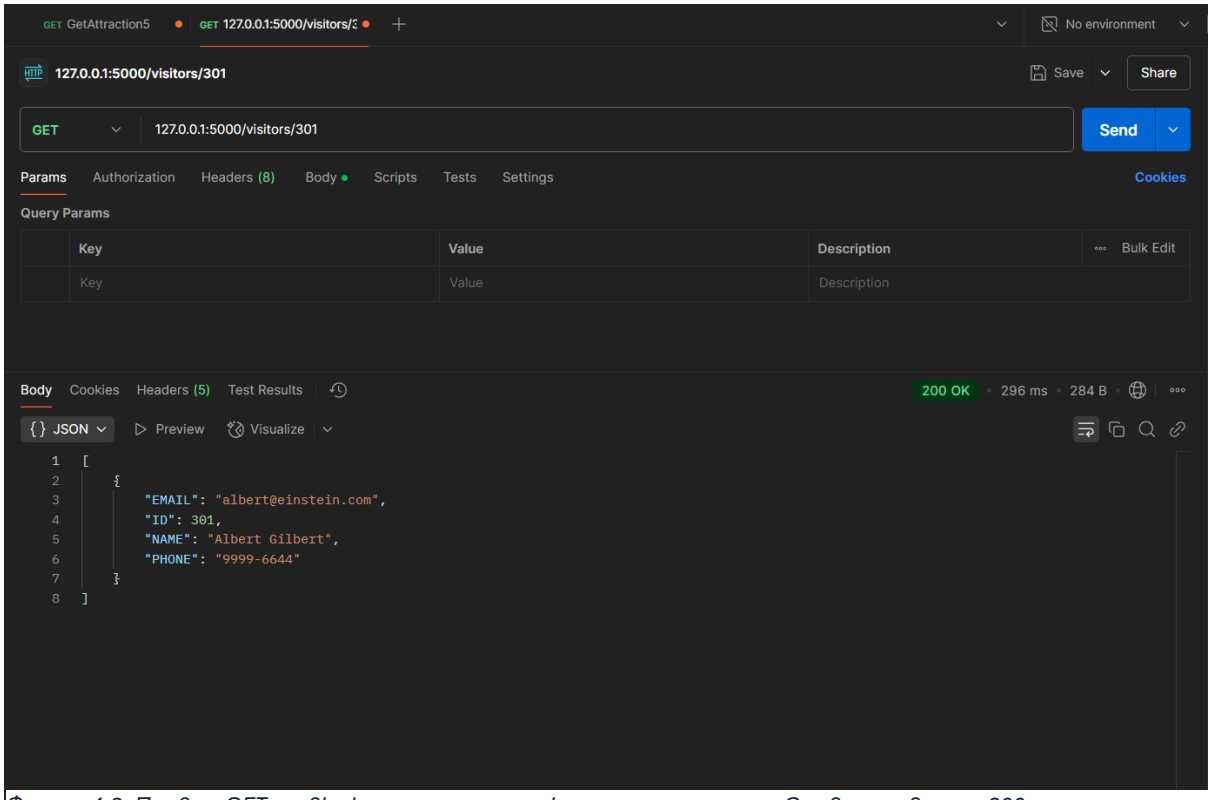
Фигура 3. Правим заявка, която да ни извлече данните само за атракцион с ID 5 Връща се резултат – данните от таблицата; Сървърния отговор е 200 OK.



Фигура 2. POST заявка. Въвеждаме данните в JSON формат в секцията Body(raw, JSON). Сървъра връща код 201 (Успешно създаден) и текстово съобщение за успешно създаване на билет



Фигура 4.1. PUT заявка. Въвеждаме данните в JSON формат в секцията Body(raw, JSON). В Body долу виждаме текущите данни за обекта в базата данни. Пращаме заявката – връща 200 – записът е успешно редактиран.



Фигура 4.2. Правим GET заявка към посетителя, когото променихме. Сървърът връща 200 и данните на посетителя с ИД 301. Вижда се, че данните са променени и новите стойности са записани в базата.