



Курсовая работа на тему «Серверная часть веб-приложения "Платформа онлайн образования"»

Дисциплина: Разработка серверных частей интернет-ресурсов

Работу выполнил: студент группы ИКБО-24-20 Постнов Н. С.

Руководитель работы: старш. преп. Сеницын А. В.



Цель данной курсовой работы

Создание серверной части веб-приложения по продаже автомобилей

Какие задачи должно выполнять приложение?

Приложение должно предоставлять пользователям следующий функционал:

Функция
Получить каталог автомобилей
Получить детальную информацию о каждом автомобиле из каталога
Поиск требуемого автомобиля
Получить список автомобилей определенного бренда
Получить изображение автомобиля из каталога
Войти в свой аккаунт
Зарегистрироваться
Выйти из своего аккаунта

Какие задачи должно выполнять приложение?

Приложение должно предоставлять администраторам следующий функционал:

Функция
Получить список всех автомобилей
Получить список всех пользователей
Добавить информацию об автомобиле
Добавить информацию о пользователе
Изменить информацию об автомобиле
Изменить информацию о пользователе
Получить информацию о автомобиле
Получить информацию о пользователе
Удалить информацию об автомобиле
Удалить информацию о пользователе



Тактико-техническое задание

- Коммуницирование с сервером осуществляется с помощью запросов к REST API сервера,
- Взаимодействие пользователей с сервером осуществляется через программу Postman,
- Сервер автоматизирует и оптимизирует работу предприятия и повышает удобство использования услуг предприятия клиентами.

Серверная часть использует:

- язык Java 11й версии вместе со своей платформой JVM,
- фреймворк Spring для создания REST-full инфраструктуры сервера, для коммуникации между клиентом и сервером,
- фреймворк Hibernate для создания и управления СУБД и возможности использовать ООП при работе с базой данных,
- PostgreSQL в качестве СУБД для управления базой данных,
- библиотека Project Lombok – для оптимизации разработки,
- система контроля версий Git,
- онлайн платформа для хранения исходных кодов проекта GitHub.com,
- Интегрированная среда разработки IntelliJ IDEA Ultimate для поддержки фреймворков и библиотек, а также для повышения удобства разработки.

Схема общей архитектуры веб-приложения

- Используется клиент-серверная архитектура
- Сервер спроектирован по паттерну MVC
- Контроллер реализует REST API
- Используется разделение элементов на логические уровни
- Уровни относительно независимы
- Позволяет относительно легко и быстро масштабировать приложение

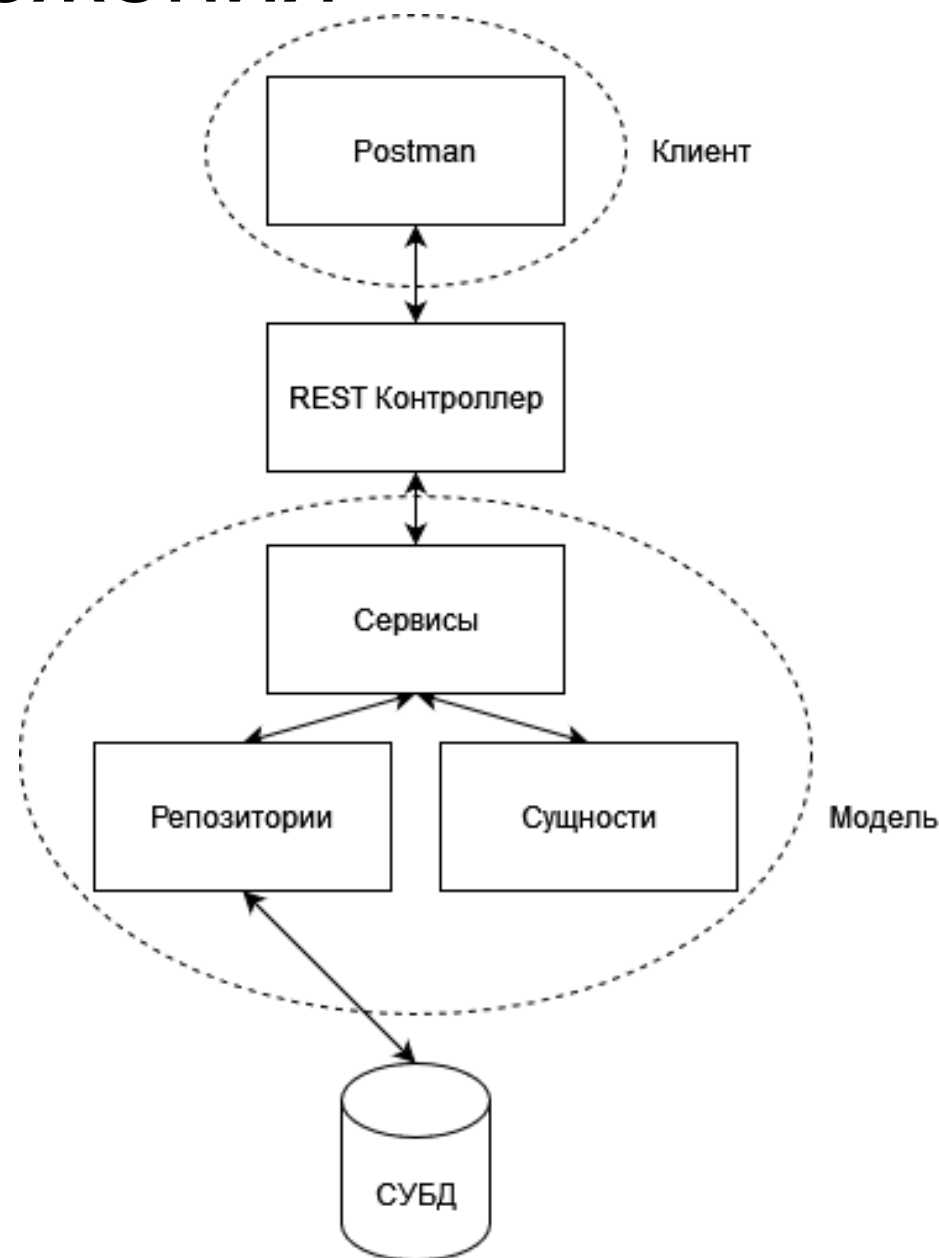
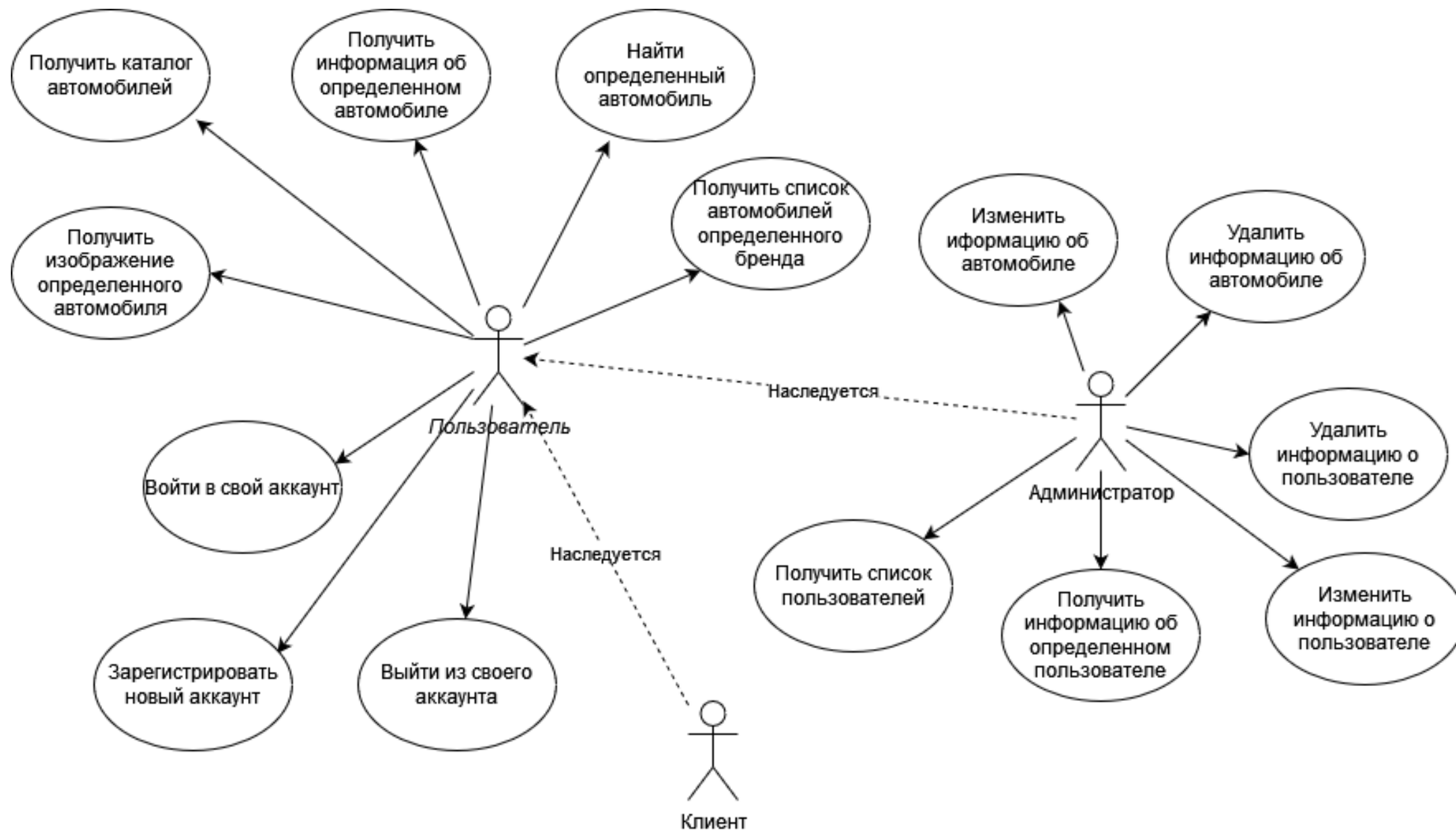
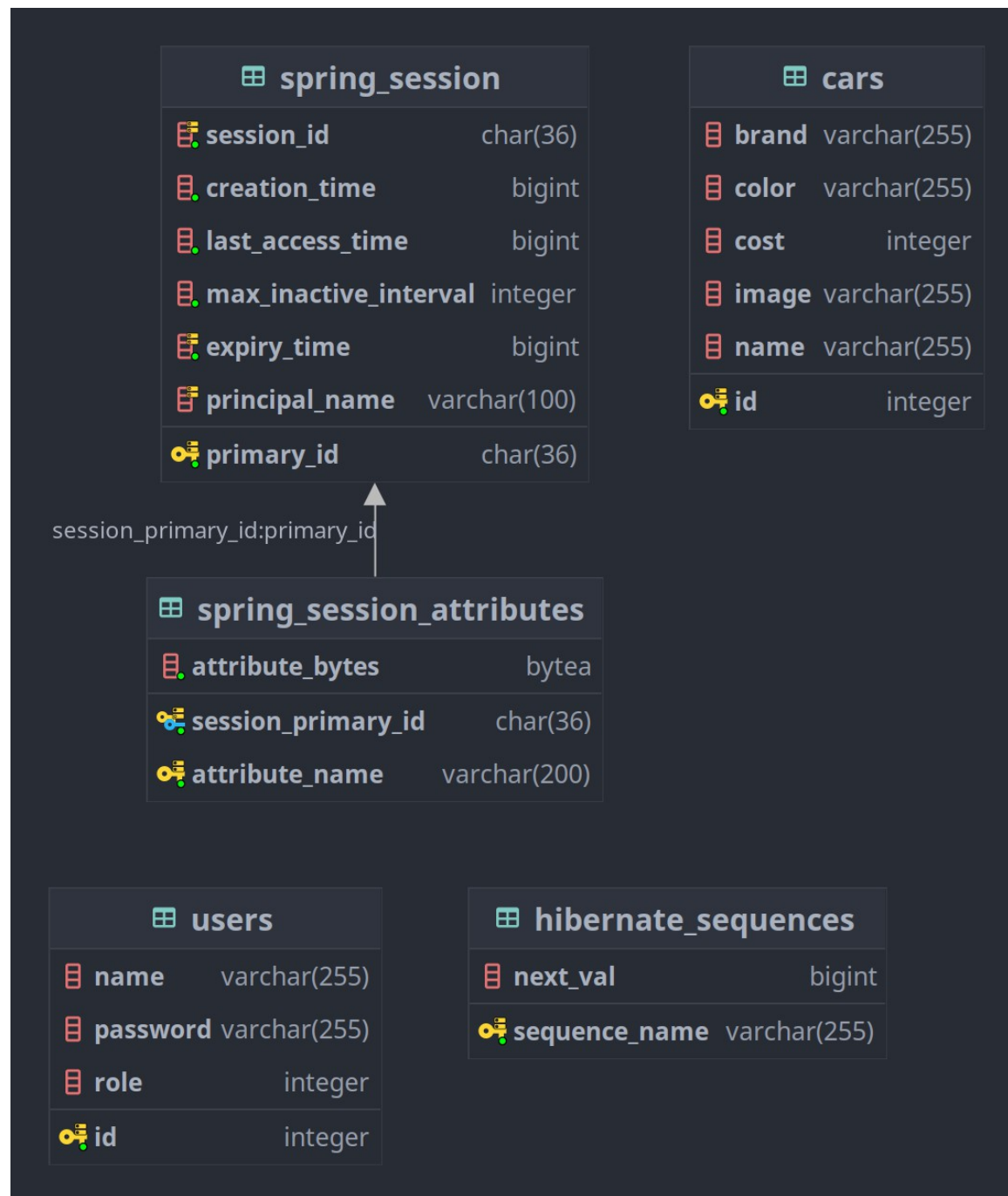


Диаграмма взаимодействия с приложением



Структура базы данных



Тестирование запросов

Postman

File Edit View Help

Home Workspaces ▾ API Network ▾ Explore

Search Postman

Invite Upgrade

Overview POST localhost:8080/login + ... No Environment

localhost:8080/login Save

POST localhost:8080/login Send

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

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

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/>	name	user			
<input checked="" type="checkbox"/>	password	user			
	Key	Text ▾ Value	Description		

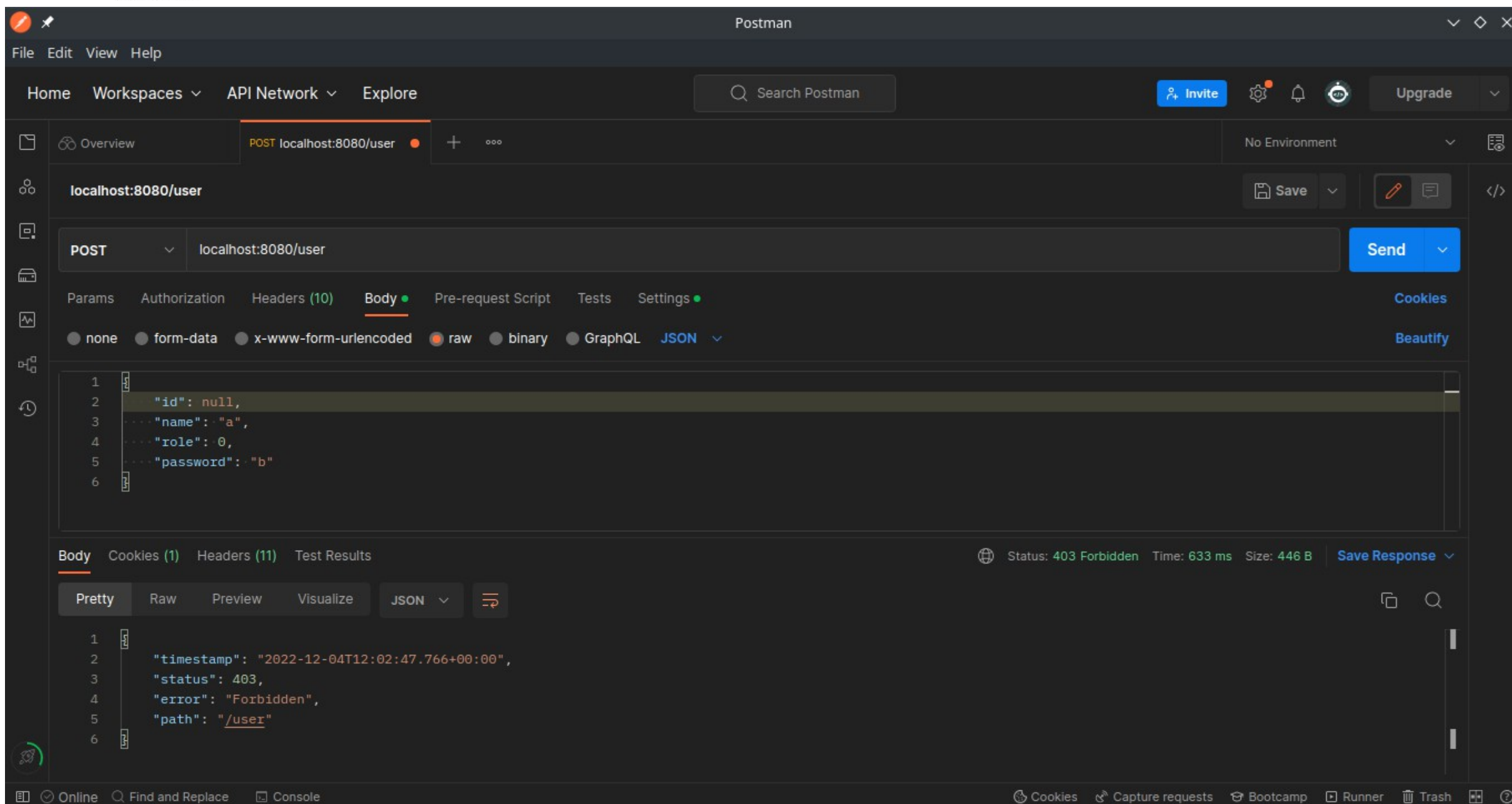
Body Cookies (1) Headers (12) Test Results

Status: 302 Found Time: 504 ms Size: 444 B Save Response ▾

Name	Value	Domain	Path	Expires	HttpOnly	Secure
SESSION	ZWJhMTVko' ...	localhost	/	Session	true	false

Online Find and Replace Console Cookies Capture requests Bootcamp Runner Trash

Тестирование запросов



The screenshot displays the Postman application interface. The top navigation bar includes 'File', 'Edit', 'View', and 'Help'. The main interface is divided into several sections:

- Overview:** Shows the selected request 'POST localhost:8080/user'.
- Request Details:** The method is 'POST' and the URL is 'localhost:8080/user'. The 'Body' tab is active, showing a JSON payload:

```
{  "id": null,  "name": "a",  "role": 0,  "password": "b"}
```
- Response:** The status is '403 Forbidden', the time is '633 ms', and the size is '446 B'. The response body is displayed in the 'Body' tab, showing a JSON error object:

```
{  "timestamp": "2022-12-04T12:02:47.766+00:00",  "status": 403,  "error": "Forbidden",  "path": "/user"}
```

The bottom status bar indicates the application is 'Online' and provides options for 'Find and Replace', 'Console', 'Cookies', 'Capture requests', 'Bootcamp', 'Runner', 'Trash', and a help icon.

Тестирование запросов

Postman

File Edit View Help

Home Workspaces ▾ API Network ▾ Explore

Search Postman

Invite Upgrade

Overview POST localhost:8080/user + ... No Environment

localhost:8080/user Save

POST localhost:8080/user Send

Params Authorization Headers (11) Body Pre-request Script Tests Settings Cookies

Headers Hide auto-generated headers

	KEY	VALUE	DESCRIPTION	...	Bulk Edit	Presets ▾
<input checked="" type="checkbox"/>	Authorization	Basic Og==				
<input checked="" type="checkbox"/>	Cookie	SESSION=OGY5YTdiZWQtYzMtYS00ZWNIITg2YjltZDVIYzc2MmQy...				Go to cookies
<input checked="" type="checkbox"/>	Postman-Token	<calculated when request is sent>				
<input checked="" type="checkbox"/>	Content-Type	application/json				

Body Cookies (1) Headers (10) Test Results

Status: 200 OK Time: 428 ms Size: 305 B Save Response ▾

Pretty Raw Preview Visualize Text ▾

1

Online Find and Replace Console Cookies Capture requests Bootcamp Runner Trash

Тестирование запросов

Postman

File Edit View Help

Home Workspaces ▾ API Network ▾ Explore

Search Postman

Invite Upgrade

Overview GET localhost:8080/all/car + ... No Environment ▾

cursov_p / localhost:8080/all/car Save ▾ ...

GET ▾ localhost:8080/all/car Send ▾

Params Authorization • Headers (8) Body Pre-request Script Tests Settings • Cookies </>

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (11) Test Results

Status: 200 OK Time: 882 ms Size: 1.92 KB Save Response ▾

Pretty Raw Preview Visualize JSON ▾

```
1 {
2   "image": "MITSUBISHI LANCER RED",
3   "cost": 750000,
4   "color": "red",
5   "name": "LANCER",
6   "id": 1,
7   "brand": "MITSUBISHI"
8 }
9 },
10 {
11   "image": "CHEVROLET AVEO BLACK",
12   "cost": 430000,
13   "color": "black",
14   "name": "AVEO",
15   "id": 2,
```

Online Find and Replace Console Cookies Capture requests Bootcamp Runner Trash ?

Тестирование запросов

Postman

File Edit View Help

Home Workspaces ▾ API Network ▾ Explore

Search Postman

Invite Upgrade

Overview GET localhost:8080/all/car + ... No Environment ▾

cursov_p / localhost:8080/all/car Save ...

GET localhost:8080/car?id=3 Send ▾

Params • Authorization • Headers (8) Body Pre-request Script Tests Settings • Cookies </>

Query Params

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/>	id	3			
	Key	Value	Description		

Body Cookies Headers (11) Test Results

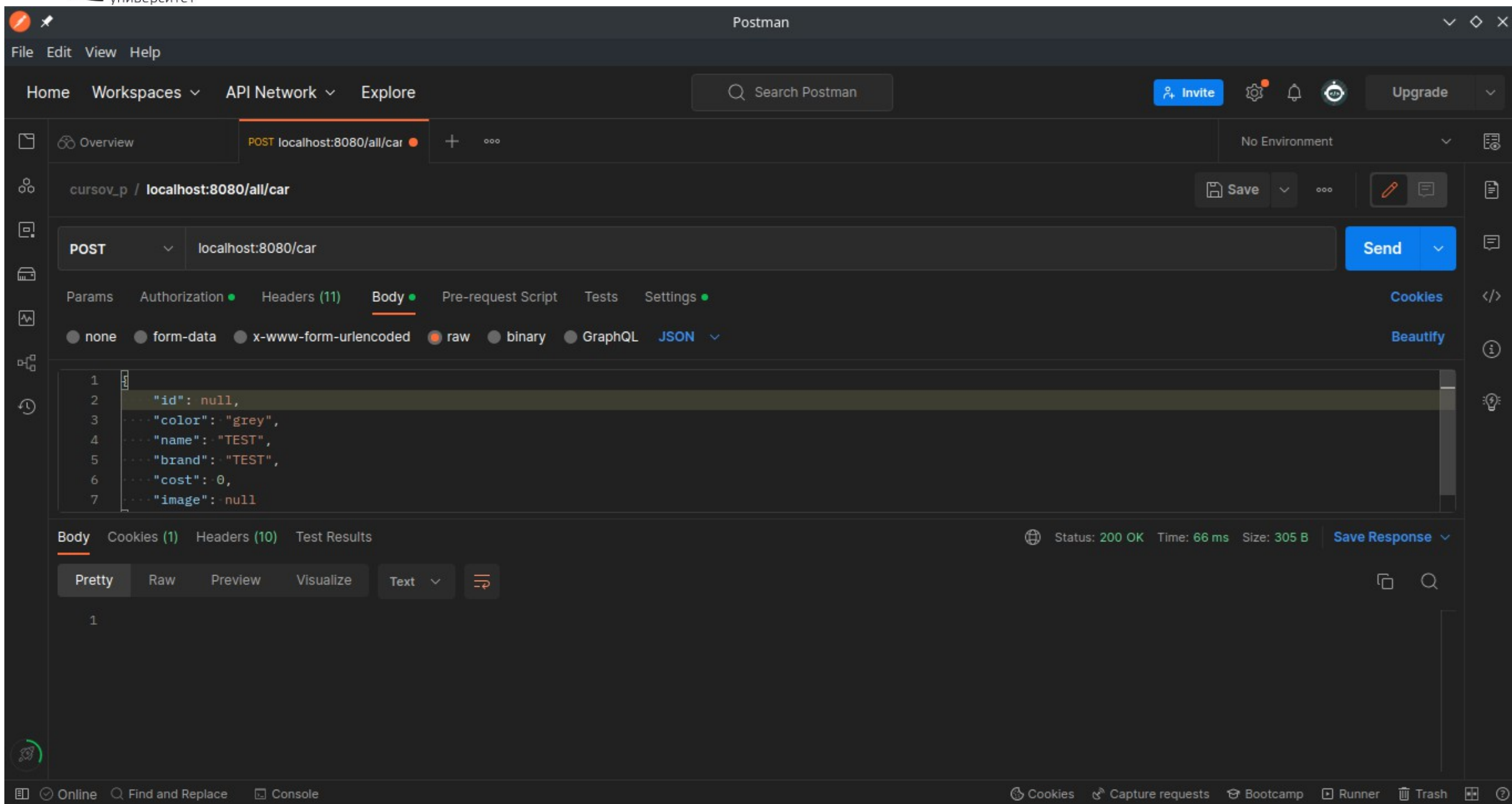
Status: 200 OK Time: 30 ms Size: 443 B Save Response ▾

Pretty Raw Preview Visualize JSON ▾

```
1 {  
2   "image": "FORD EXPLORER RED",  
3   "cost": 870000,  
4   "color": "red",  
5   "name": "EXPLORER",  
6   "id": 3,  
7   "brand": "FORD"  
8 }
```

Online Find and Replace Console Cookies Capture requests Bootcamp Runner Trash

Тестирование запросов

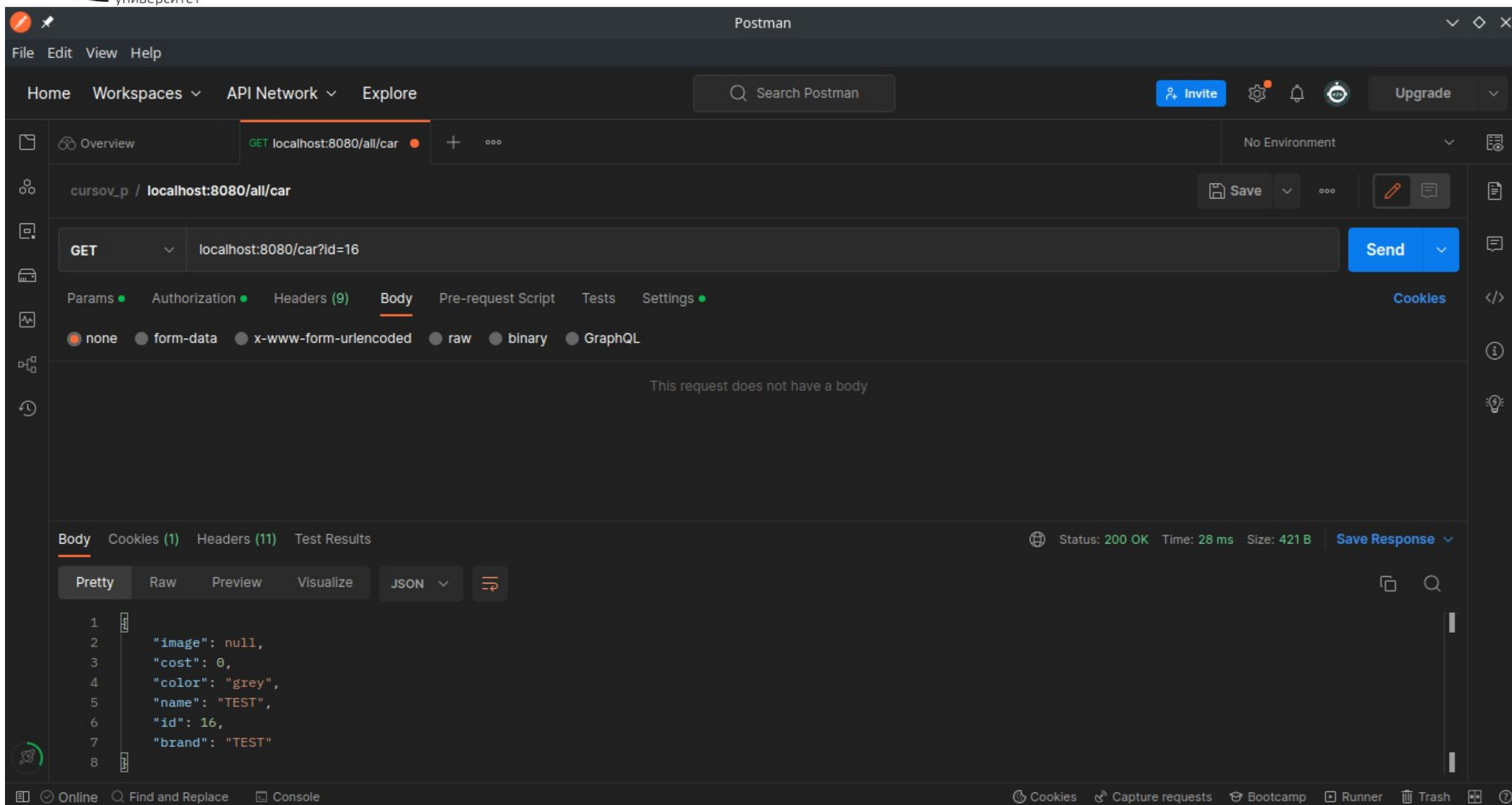


The screenshot displays the Postman application interface. The top navigation bar includes 'File', 'Edit', 'View', and 'Help'. Below it, the 'Home' tab is active, showing 'Workspaces' and 'API Network'. A search bar labeled 'Search Postman' is present. The main workspace shows a collection named 'cursov_p' with a folder 'localhost:8080/all/car'. A POST request is selected, and the 'Body' tab is active. The request body is a JSON object:

```
1 {
2   "id": null,
3   "color": "grey",
4   "name": "TEST",
5   "brand": "TEST",
6   "cost": 0,
7   "image": null
8 }
```

The status bar at the bottom indicates 'Status: 200 OK', 'Time: 66 ms', and 'Size: 305 B'. The 'Save Response' button is visible. The bottom of the interface shows a status bar with 'Online', 'Find and Replace', 'Console', 'Cookies', 'Capture requests', 'Bootcamp', 'Runner', 'Trash', and a help icon.

Тестирование запросов



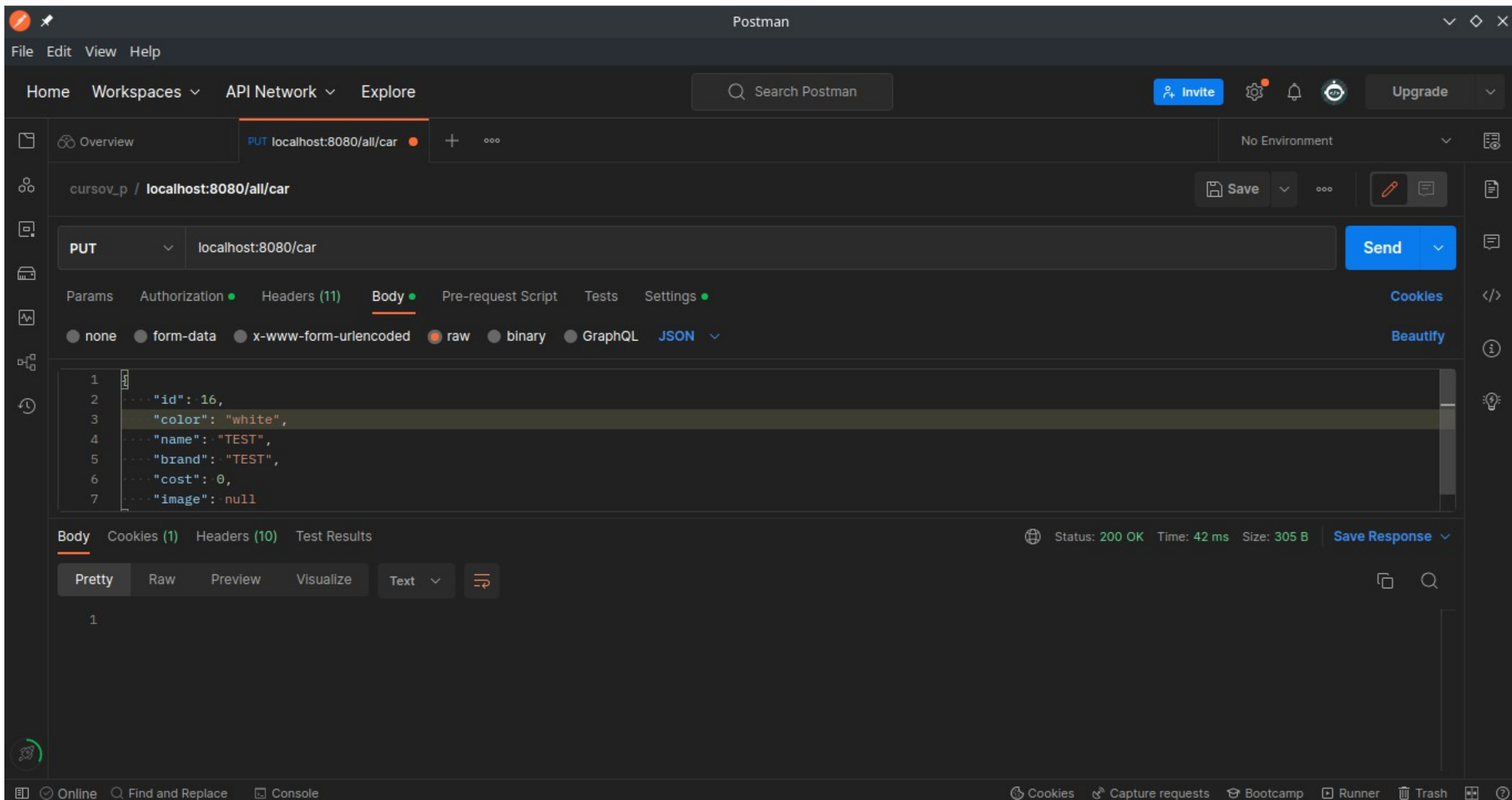
The screenshot displays the Postman application interface. At the top, the menu bar includes File, Edit, View, and Help. Below it, the navigation bar shows Home, Workspaces, API Network, and Explore. A search bar labeled "Search Postman" is present. The main workspace is divided into several sections:

- Overview:** Shows a list of requests, including a GET request to `localhost:8080/all/car`.
- Request Editor:** The selected request is a GET request to `localhost:8080/car?id=16`. The "Body" tab is active, showing "none" as the body type. The "Send" button is visible.
- Response View:** The "Body" tab is selected, showing the response in "Pretty" JSON format. The status is "200 OK", the time is "28 ms", and the size is "421 B". The response data is:

```
{  "image": null,  "cost": 0,  "color": "grey",  "name": "TEST",  "id": 16,  "brand": "TEST"}
```

The bottom status bar includes icons for Online, Find and Replace, Console, Cookies, Capture requests, Bootcamp, Runner, Trash, and a help icon.

Тестирование запросов



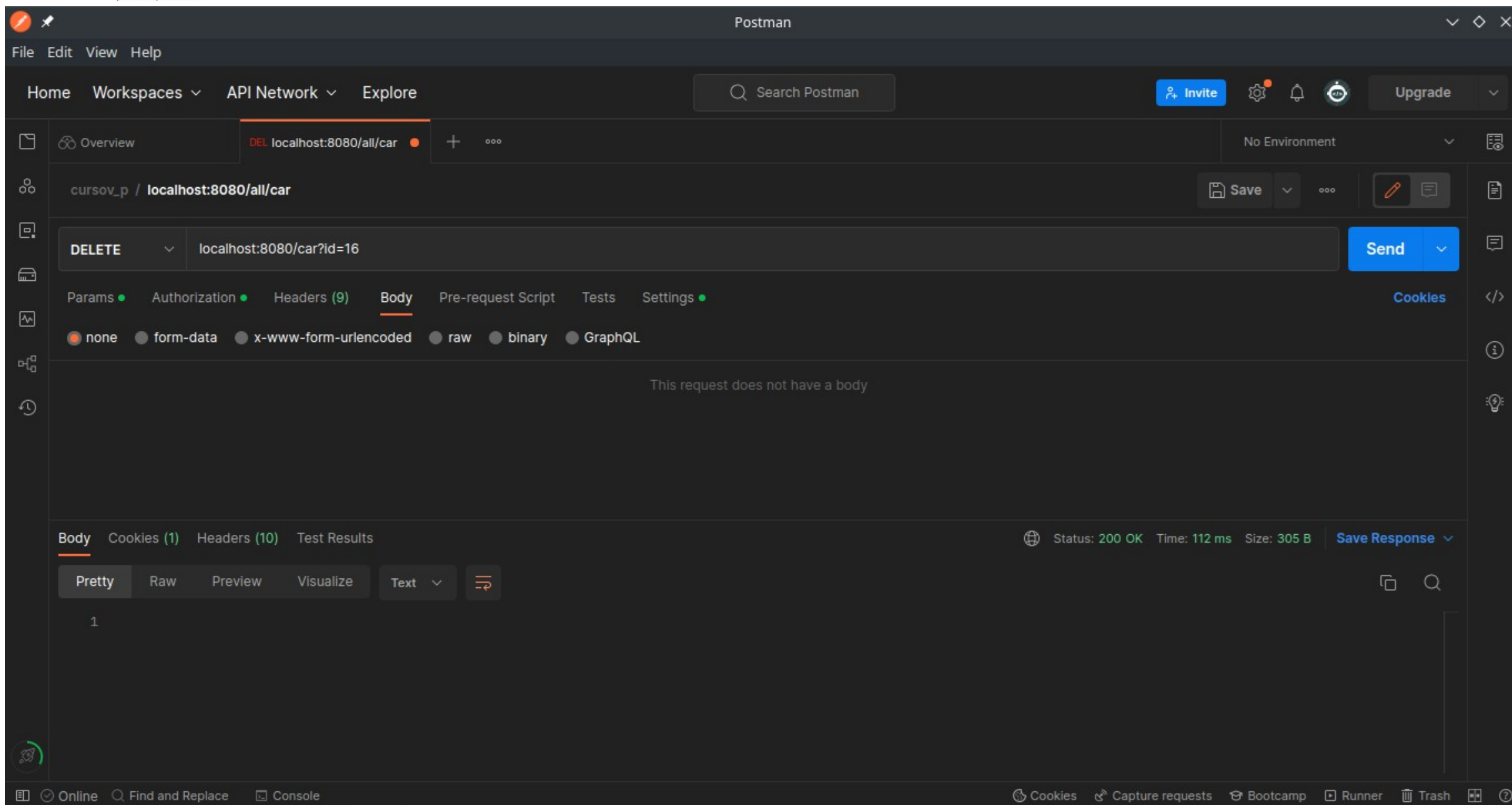
The screenshot displays the Postman application interface. The top navigation bar includes 'File', 'Edit', 'View', and 'Help'. Below it, the main navigation bar shows 'Home', 'Workspaces', 'API Network', and 'Explore'. A search bar labeled 'Search Postman' is present. The right side of the top bar features an 'Invite' button, a settings icon, a notification bell, a status icon, and an 'Upgrade' button.

The main workspace is divided into several sections. On the left, there's a sidebar with icons for Overview, Collections, Environments, Recent, History, and Sync. The central area shows a PUT request to 'localhost:8080/all/car'. The request is saved under the collection 'cursov_p'. The 'Body' tab is selected, showing a JSON payload:

```
1 {
2   "id": 16,
3   "color": "white",
4   "name": "TEST",
5   "brand": "TEST",
6   "cost": 0,
7   "image": null
8 }
```

Below the request body, the 'Body' tab is selected in the response section, showing the response in 'Pretty' format. The status bar at the bottom indicates 'Status: 200 OK', 'Time: 42 ms', and 'Size: 305 B'. The bottom of the interface includes a status bar with 'Online', 'Find and Replace', 'Console', 'Cookies', 'Capture requests', 'Bootcamp', 'Runner', 'Trash', and a help icon.

Тестирование запросов



The screenshot displays the Postman application interface. At the top, the title bar reads "Postman". Below it is a menu bar with "File", "Edit", "View", and "Help". The main interface is divided into several sections:

- Top Bar:** Includes tabs for "Home", "Workspaces", "API Network", and "Explore". A search bar labeled "Search Postman" is on the right, along with buttons for "Invite", "Upgrade", and a "No Environment" dropdown.
- Left Sidebar:** Contains icons for "Overview", "cursov_p / localhost:8080/all/car", and other API items.
- Request Editor:** Shows a "DELETE" request to "localhost:8080/car?id=16". The "Body" tab is selected, showing "This request does not have a body". Other tabs include "Params", "Authorization", "Headers (9)", "Pre-request Script", "Tests", and "Settings".
- Response Section:** At the bottom, it shows the response status "Status: 200 OK", time "Time: 112 ms", and size "Size: 305 B". A "Save Response" button is visible.
- Bottom Bar:** Includes a status bar with "Online", "Find and Replace", "Console", and other utility icons.

Тестирование запросов

Postman

File Edit View Help

Home Workspaces ▾ API Network ▾ Explore

Search Postman

Invite Upgrade

Overview GET localhost:8080/all/car + ...

No Environment

cursov_p / localhost:8080/all/car

Save ...

GET http://localhost:8080/static/CHEVROLET AVEO BLACK.jpg Send ▾


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

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

This request does not have a body

Body Cookies (1) Headers (16) Test Results

Status: 200 OK Time: 28 ms Size: 171.62 KB Save Response ▾



Online Find and Replace Console

Cookies Capture requests Bootcamp Runner Trash



Выводы

По итогам выполнения данной работы были получены навыки проектирования и разработки серверных частей интернет-ресурсов и баз данных.

Разработка велась с учетом спроектированной архитектуры на основе паттерна MVC. Использовалась архитектура REST для создания прикладного интерфейса для взаимодействия с сервером.

Созданный программный продукт отвечает всем поставленным техническим заданием целям. Он был полностью протестирован.



Спасибо за внимание!

Исходные коды приложения хранятся на удаленном репозитории на платформе GitHub.com, расположенном по адресу
<https://github.com/NikitaPostnov101/RSCHIR/tree/main/Course%20work>