



МИНОБРНАУКИ РОССИИ

*Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА*

Институт Информационных технологий (ИТ)

Кафедра инструментального и прикладного программного обеспечения
(ИиППО)

**ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 4
по дисциплине**

«Проектирование и разработка серверных частей интернет-ресурсов»

Выполнил студент группы ИКБО-20-23

Комисарик М.А.

Принял Ассистент кафедры ИиППО

Благирев М.М.

Практическая работа выполнена

«__» 202__ г.

(подпись студента)

«Зачтено»

«__» 202__ г.

(подпись руководителя)

Москва 2025

ХОД РАБОТЫ

```
location /api/ {
    rewrite ^/api/(.*)$ /api.php/$1 break;
    proxy_pass http://apache_backend;
    proxy_buffering off;
    proxy_http_version 1.1;
    proxy_set_header Host $host;
}
```

Рисунок 1 – Новый блок location в файле nginx.conf

```
1 <?php
2 header( header: 'Content-Type: application/json; charset=utf-8');
3 require_once '../includes/db_connect.php';
4
5 $method = $_SERVER['REQUEST_METHOD'];
6 if (!key_exists( key: 'PATH_INFO', $_SERVER))
7 {
8     echo "Enter the query";
9     return;
10 }
11 $request = explode( separator: '/', trim($_SERVER['PATH_INFO']), characters: '/');
12 $entity = $request[0] ?? '';
13 $id = $request[1] ?? null;
14
15 if (!isset($_SERVER['PHP_AUTH_USER']))
16 {
17     header( header: 'WWW-Authenticate: Basic realm="Autoservice API"');
18     http_response_code( response_code: 401);
19     echo json_encode(['error' => 'Требуется аутентификация.']);
20     exit;
21 }
22 if ($_SERVER['PHP_AUTH_USER'] != 'admin' || $_SERVER['PHP_AUTH_PW'] != 'admin123')
23 {
24     http_response_code( response_code: 403);
25     echo json_encode(['error' => 'Неверные учетные данные.']);
26     exit;
27 }
```

Рисунок 2 –api.php, часть 1

```

29     try
30     {
31         switch ($method)
32         {
33             case 'GET':
34                 if ($entity == 'services')
35                 {
36                     if ($id)
37                     {
38                         $statement = $pdo->prepare(query: "SELECT * FROM services WHERE id = ?");
39                         $statement->execute([$id]);
40                         $service = $statement->fetch(mode: PDO::FETCH_ASSOC);
41                         if ($service)
42                         {
43                             echo json_encode($service);
44                         } else
45                         {
46                             http_response_code(response_code: 404);
47                             echo json_encode(['error' => 'Услуга не найдена.']);
48                         }
49                     } else
50                     {
51                         $statement = $pdo->query(query: "SELECT * FROM services");
52                         $services = $statement->fetchAll(mode: PDO::FETCH_ASSOC);
53                         echo json_encode($services);
54                     }
55                 }
56             break;

```

Рисунок 3 – api.php, часть 2

```

57         case 'POST':
58             $input = json_decode(file_get_contents(filename: 'php://input'), associative: true);
59             if ($entity == 'services')
60             {
61                 $statement = $pdo->prepare(query: "INSERT INTO services (name, description, price, duration, category) VALUES (?, ?, ?, ?, ?)");
62                 $statement->execute([$input['name'], $input['description'], $input['price'], $input['duration'], $input['category']]);
63                 http_response_code(response_code: 201);
64                 echo json_encode(['message' => 'Услуга успешно создана.', 'id' => $pdo->lastInsertId()]);
65             }
66             break;
67         case 'PUT':
68             $input = json_decode(file_get_contents(filename: 'php://input'), associative: true);
69             if ($entity == 'services' && $id)
70             {
71                 $statement = $pdo->prepare(query: "UPDATE services SET name=?, description=?, price=?, duration=?, category=? WHERE id=?");
72                 $statement->execute([$input['name'], $input['description'], $input['price'], $input['duration'], $input['category'], $id]);
73                 echo json_encode(['message' => 'Услуга успешно обновлена.']);
74             }
75             break;
76         case 'DELETE':
77             if ($entity == 'services' && $id)
78             {
79                 $statement = $pdo->prepare(query: "DELETE FROM services WHERE id=?");
80                 $statement->execute([$id]);
81                 echo json_encode(['message' => 'Услуга успешно удалена.']);
82             }
83             break;

```

Рисунок 4 – api.php, часть 3

```

84             default:
85                 http_response_code(response_code: 405);
86                 echo json_encode(['error' => 'Метод не разрешен.']);
87                 break;
88             }
89         }
90     catch (Exception $e)
91     {
92         http_response_code(response_code: 500);
93         echo json_encode(['error' => 'Внутренняя ошибка сервера: ' . $e->getMessage()]);
94     }

```

Рисунок 5 – api.php, часть 4

```

GET http://localhost/api/services
200 OK
25 ms 3.57 KB
[{"id": 1, "name": "Замена моторного масла", "description": "Полная замена моторного масла и масляного фильтра", "price": "50.00", "duration": 45, "category": "Техническое обслуживание"}, {"id": 2, "name": "Замена воздушного фильтра", "description": "Замена воздушного фильтра двигателя", "price": "25.00", "duration": 20, "category": "Техническое обслуживание"}, {"id": 3, "name": "Замена тормозной жидкости", "description": "Полная замена тормозной жидкости с прокачкой", "price": "60.00", "duration": 60, "category": "Тормозная система"}, {"id": 4, "name": "Замена масла в АКПП", "description": "Замена масла в автоматической коробке передач", "price": "100.00", "duration": 90, "category": "Техническое обслуживание"}]

```

Рисунок 6 – GET запрос на получение всех услуг

```

GET http://localhost/api/services/1
200 OK
23 ms 800 B
{
  "id": 1,
  "name": "Замена моторного масла",
  "description": "Полная замена моторного масла и масляного фильтра",
  "price": "50.00",
  "duration": 45,
  "category": "Техническое обслуживание"
}

```

Рисунок 7 – GET запрос на получение услуги с id 1

```

POST http://localhost/api/services
201 Created
42 ms 347 B
{
  "message": "Услуга успешно создана.",
  "id": 9
}

```

Рисунок 8 – POST запрос на создание услуги

```

GET http://localhost/api/services/9
{
  "id": 9,
  "name": "новый сервис",
  "description": "",
  "price": "10.00",
  "duration": 20,
  "category": ""
}
  
```

Рисунок 9 – Получение добавленной услуги

```

PUT http://localhost/api/services/9
{
  "description": "тестовое описание",
  "category": "тестовая категория"
}

{
  "message": "Услуга успешно обновлена."
}
  
```

Рисунок 10 – PUT запрос на изменение новой услуги

```

GET http://localhost/api/services/9
{
  "id": 9,
  "name": "новый сервис",
  "description": "тестовое описание",
  "price": "10.00",
  "duration": 20,
  "category": "тестовая категория"
}
  
```

Рисунок 11 – Получение измененной услуги

```

DELETE http://localhost/api/services/9
{
  "message": "Услуга успешно удалена."
}
  
```

Рисунок 12 – Удаление новой услуги

The screenshot shows the Postman application interface. At the top, there is a header with 'GET' and the URL 'http://localhost/api/services/9'. On the right side of the header is a blue 'Send' button. Below the header, there are tabs for 'Params', 'Authorization', 'Headers (9)', 'Body', 'Scripts', and 'Settings'. The 'Params' tab is currently selected. Under 'Params', there is a section titled 'Query Params' with a table. The table has columns for 'Key', 'Value', and 'Description'. There is one row with the key 'Body' and the value '{ } JSON'. Below the table, there are buttons for 'Preview' and 'Debug with AI'. To the right of the table, there is a status bar showing '404 Not Found', '24 ms', '308 B', and other icons. At the bottom of the interface, there are several small icons.

Рисунок 13 – Проверка успешности удаления новой услуги