

Продолжим разработку проекта информационной системы «МояГибдд»

1. Мы сделали классы объектов, научились записывать данные в файл

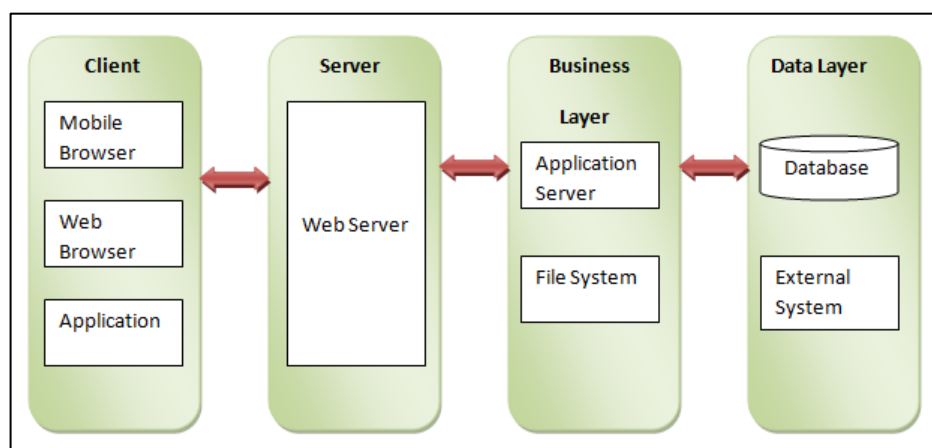
Название класса	Описание объекта
User	Пользователь
Auto	Автомобиль
Owner	Владелец

2. Мы сделали внешний вид веб-странички

**МояГИБДД**  
**Список владельцев**  
ID Фамилия Имя Телефон Почта Город  
**Список авто**  
ID Марка Модель Цвет Госномер  
**Регистрация авто**  
Владелец:  Автомобиль:    
Принадлежит авто:

Теперь мы внесем изменения в backend, чтобы он взаимодействовал с базой данных MySQL. Также предложим структуру базы данных, которая позволит хранить данные о владении несколькими автомобилями одним человеком.

Вспомним, как работает сложный динамический сайт:



DataBase (База данных) – это основное хранилище данных в сложных информационных системах. Данные хранить в коде – ЗАПРЕЩЕНО, мы научимся использовать базы данных.

На ваших компьютерах стоит комплекс программ OpenServer.

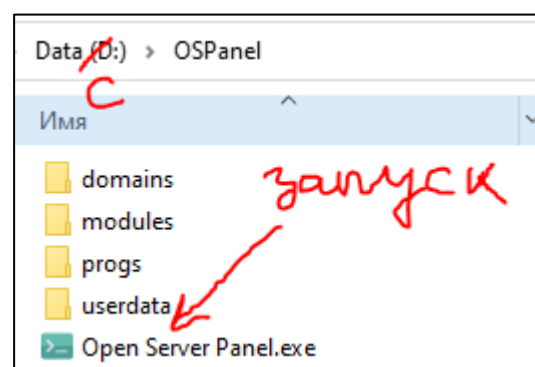
Этот комплекс используется для веб-разработки.

Из него нам потребуется «Система управления базой данных» MySQL

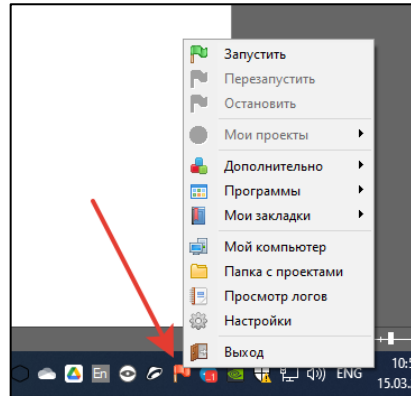
Запуск **OpenServer**:

Найдите папку Ospanel

И запустите файл:

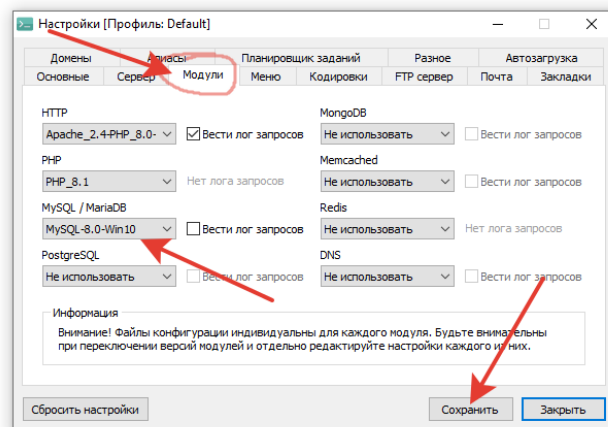


Мы увидим следующее:



Флажок – это иконка для управления OpenServer.

Зайдем в настройки и проверим включен ли модуль баз данных



Желательно использовать версию MySQL 8, но если не доступна, то достаточно и MySQL 5.7

Для проекта "МояГИБДД" реализуем следующую структуру базы данных:

1. **Таблица users:**

- Хранит информацию о владельцах.
- Атрибуты:
  - id (INT, PRIMARY KEY, AUTO\_INCREMENT)
  - last\_name (VARCHAR(50)) — фамилия
  - first\_name (VARCHAR(50)) — имя
  - phone (VARCHAR(15)) — телефон
  - email (VARCHAR(100)) — почта
  - city (VARCHAR(50)) — город

2. **Таблица autos:**

- Хранит информацию об автомобилях.
- Атрибуты:
  - id (INT, PRIMARY KEY, AUTO\_INCREMENT)
  - brand (VARCHAR(50)) — марка

- model (VARCHAR(50)) — модель
- color (VARCHAR(20)) — цвет
- license\_plate (VARCHAR(15)) — госномер

### 3. Таблица owners:

- Хранит информацию о владении автомобилями.
- Атрибуты:
  - id (INT, PRIMARY KEY, AUTO\_INCREMENT)
  - user\_id (INT, FOREIGN KEY REFERENCES users(id)) — ID владельца
  - auto\_id (INT, FOREIGN KEY REFERENCES autos(id)) — ID автомобиля
  - registration\_date (DATE) — дата регистрации
  - deregistration\_date (DATE, NULL) — дата снятия с учета (может быть NULL, если автомобиль еще зарегистрирован)


Изменить код приложения бэкенда

1. Установим необходимые зависимости в терминале в папке проекта:

```
npm install mysql2 express
```

это в наш проект подгружается из интернета пакет работы с базой данных, мы будем использовать функции, хранящиеся в этом пакете

2. Переименуйте старый файл index.js в папке backend в index\_old.js
3. Напишите код в файл index.js в папке backend:
  - a. Зададим параметры подключения к базе данных (БД)

```
hapt_00-3_full_project > backend >  index.js > ...
1  const express = require('express');
2  const mysql = require('mysql2');
3  const app = express();
4  const port = 3000;
5
6  // Подключение к базе данных MySQL
7  const connection = mysql.createConnection({
8    host: 'localhost',
9    user: 'useradmin',
10    password: 'super@dm1n',
11    database: 'databasegibdd',
12  });
13
```

- b. Создадим подключение к БД

```
14  connection.connect((err) => {
15    if (err) {
16      console.error('Ошибка подключения к базе данных:', err);
17      return;
18    }
19    console.log('Подключение к базе данных успешно установлено');
20  });
21
```

- c. Приступим к части получения данных из БД

```

22 // Middleware для обработки JSON
23 app.use(express.json());
24
25 // Маршрут для получения списка владельцев
26 app.get('/api/users', (req, res) => {
27     connection.query('SELECT * FROM users', (err, results) => {
28         if (err) {
29             console.error('Ошибка при получении данных:', err);
30             res.status(500).send('Ошибка сервера');
31             return;
32         }
33         res.json(results);
34     });
35 });
36

```

- i.
- ii. Аналогично напишите для АВТО

```

37 // Маршрут для получения списка
38 app.get('/api/autos', (req, res) => {
39     connection.query('SELECT *
40         if (err) {
41             console.error('Оши
42             res.status(500).se
43             return;
44         }
45         res.json(results);
46     });
47 });
48

```



- iii. И владельцев

```

49 // Маршрут для получения списка владе
50 app.get('/api/owners', (req, res) =>
51     connection.query('SELECT * FROM o
52     if (err) {
53         console.error('Ошибка при
54         res.status(500).send('Оши
55         return;
56     }
57     res.json(results);
58 });
59 });
60

```

- d. Важный блок: регистрация авто на владельца

```

61 // Маршрут для регистрации автомобиля за владельцем
62 app.post('/api/owners', (req, res) => {
63   const { userId, autoId, registrationDate } = req.body;
64
65   if (!userId || !autoId || !registrationDate) {
66     res.status(400).send('Недостаточно данных');
67     return;
68   }
69
70   const query = `
71     INSERT INTO owners (user_id, auto_id, registration_date)
72     VALUES (?, ?, ?)
73   `;
74
75   connection.query(query, [userId, autoId, registrationDate], (err, results) => {
76     if (err) {
77       console.error('Ошибка при добавлении записи:', err);
78       res.status(500).send('Ошибка сервера');
79       return;
80     }
81     res.status(201).send('Автомобиль успешно зарегистрирован');
82   });
83 });

```

#### 4. Код для запуска сервера

```

85 // Запуск сервера
86 app.listen(port, () => {
87   console.log(`Сервер запущен на http://localhost:\${port}`);
88 });

```

Обновляем frontend (JavaScript)

В файле frontend/script.js заменим загрузку данных из JSON на запросы к backend:

```

1 document.addEventListener('DOMContentLoaded', () => {
2   const API_URL = 'http://localhost:3000/api';
3
4   // Загружаем данные с сервера
5   async function loadData() {
6     const [users, autos, owners] = await Promise.all([
7       fetch(`${API_URL}/users`).then(res => res.json()),
8       fetch(`${API_URL}/autos`).then(res => res.json()),
9       fetch(`${API_URL}/owners`).then(res => res.json()),
10    ]);
11
12    return { users, autos, owners };
13  }
14
15

```

Остальной код без изменений

Запускаем приложение

Как это работает:

База данных: Данные хранятся в MySQL.

Таблицы связаны через внешние ключи.

Backend:

Подключается к MySQL.

Предоставляет API для получения данных и регистрации автомобилей.

Frontend:

Загружает данные с backend через API.

Отображает их в интерфейсе.

Следующие шаги:

Запустите MySQL и создайте базу данных с помощью SQL-запросов.

Запустите backend:

```
node index.js
```

Откройте frontend в браузере.

Теперь проект полностью интегрирован с базой данных MySQL!

New chat