**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по индивидуальной части проекта**

**по дисциплине «ВнСУБД»**

Тема: Управление автопарком

|  |  |  |
| --- | --- | --- |
| Студент гр. 2381 |  | Попоп В.Д. |
| Преподаватель |  | Заславский М.М. |

Санкт-Петербург

2025

## Цель работы.

Разработать и развернуть в Docker-окружении прототип веб-приложения для хранения информации об автомобилях, их обслуживании и ремонтах, с поддержкой многокритериальной фильтрации, редактирования данных и функций импорта/экспорта.

## Задание.

В виде отдельного docker-compose приложения реализовать небольшую часть общего проекта. А именно: хранение данных автомобилей, таблица для их многокритериальной фильтрации, страница для добавления и редактирования автомобилей. В рамках автомобилей нужно хранить историю технического обслуживания, замены деталей + обеспечить отдельную страницу с возможностью поиска по всем эпизодам технического обслуживания всех автомобилей. Страница для импорта экспорта всех автомобилей. Несколько захордкоженных элементов данных в БД.

## Выполнение работы.

Модель данных

Коллекция `cars`:  
- name (string)  
- model (string)  
- license\_plate (string)  
- category (string)  
- year (int)  
- service\_history (array of episodes: date, description, replaced\_parts, mileage, cost)  
- created\_at (datetime, UTC+3)  
- updated\_at (datetime, UTC+3)

**1. Хранение данных автомобилей**

* Создана коллекция cars в MongoDB.
* Для каждой машины хранится: название, модель, госномер, категория, год выпуска.
* Дополнительно фиксируются даты создания и обновления (created\_at, updated\_at) — выставляются автоматически сервером с поправкой +3 часа (по мск).

**2. История технического обслуживания (ТО)**

* В каждом документе car хранится массив service\_history.
* Каждый элемент массива содержит дату, описание работ, список заменённых деталей, пробег и стоимость.
* Это решение позволяет хранить полную историю по каждому автомобилю без отдельной коллекции.

**3. Многокритериальная фильтрация автомобилей**

* В API /cars реализованы query-параметры: name, model, license\_plate, min\_year, max\_year, created\_from, created\_to, updated\_from, updated\_to.
* По строковым полям используется регистронезависимый поиск ($regex с опцией "i").
* По числовым и временным полям — фильтры диапазона ($gte, $lte).
* На фронтенде реализована форма с полями для ввода фильтров и кнопкой «Применить».

**4. Добавление и редактирование автомобилей**

* Реализована страница form.html.
* Если открывается без параметра edit → создаётся новый автомобиль (POST /cars).
* Если есть ?edit=license\_plate → выполняется GET /cars/{plate}, поля формы заполняются, при сохранении вызывается PUT /cars/{plate}.
* При изменении госномера PUT всё равно выполняется по старому значению, но обновляет его новым.

**5. Поиск по истории ТО**

* Реализован отдельный маршрут /service\_search.
* Поддерживаются отдельные фильтры: description, replaced\_part, date, min\_mileage, max\_mileage, min\_cost, max\_cost.
* Backend формирует условия для массива service\_history и находит документы, где все эпизоды соответствуют критериям.
* Фронтенд (страница service\_search.html) позволяет пользователю ввести параметры и получить таблицу с подходящими эпизодами.

**6. Импорт и экспорт автомобилей**

* GET /cars/export возвращает все записи в формате JSON (без \_id).
* POST /cars/import принимает JSON-массив машин, очищает коллекцию и вставляет новые данные.
* На фронтенде (страница import.html) есть кнопка «Экспорт» (выводит JSON в textarea) и «Импорт» (берёт JSON из textarea и отправляет на сервер).

**7. Seed — начальные данные**

* В seed.py реализована загрузка трёх заранее заданных автомобилей (КамАЗ, Экскаватор, Погрузчик).
* При запуске seed коллекция очищается и заполняется новыми данными.
* Запускается автоматически при старте backend (в Dockerfile через CMD ["sh", "-c", "python seed.py && uvicorn ..."]).

**8. Развёртывание в Docker Compose**

* Все сервисы упакованы в контейнеры: backend, mongodb, frontend.
* MongoDB подключена через volume (mongo\_data) для сохранения данных.
* Порты проброшены только на 127.0.0.1 (повышение безопасности).
* Backend автоматически выполняет seed перед запуском сервера.

Результаты тестирования см. в приложении А.

## Выводы.

Цель задания выполнена: реализован работающий прототип подсистемы управления автомобилями для проекта спецтехники.  
Использование FastAPI + MongoDB позволило быстро создать REST API и хранение данных, а простые HTML-страницы обеспечили наглядность работы.  
Docker позволил унифицировать запуск и исключить зависимости от локальной среды.

# Приложение A Тестирование

В корне проект, где расположен Dockerfile, в терминале выполняем **docker compose up -- build.**

Изображение выглядит как текст, линия, Шрифт, снимок экрана

Автоматически созданное описаниеВ браузере переходим на стартовую страницу <http://localhost:8080/>

Изображение выглядит как текст, линия, Шрифт, снимок экрана

Автоматически созданное описаниеВ полях фильтра можно указать параметры фильтрации. Например:

Добавим автомобиль, перейдя по ссылке Добавить автомобиль

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Видим результат в таблице

Изображение выглядит как текст, снимок экрана, линия, Шрифт

Автоматически созданное описаниеИзображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описаниеУдалим экскаватор, нажав на кнопку Удалить. Результат:

Изображение выглядит как текст, линия, Шрифт, снимок экрана

Автоматически созданное описаниеНа странице Поиск по ТО мы может отфильтровать ТО машин по критериям

На странице импорт/экспорт мы можем получить текстовый файл базы данных в виде JSON, нажав кнопку export.

*[*

*{*

*"name": "КамАЗ самосвал",*

*"model": "КамАЗ-6520",*

*"license\_plate": "А123БВ77",*

*"category": "Грузовой",*

*"year": 2015,*

*"service\_history": [*

*{*

*"date": "2023-01-10",*

*"description": "Плановое ТО, замена масла",*

*"replaced\_parts": ["Масло", "Фильтр"],*

*"mileage": 120000,*

*"cost": 15000*

*},*

*{*

*"date": "2023-06-20",*

*"description": "Ремонт тормозов",*

*"replaced\_parts": ["Тормозные колодки"],*

*"mileage": 135000,*

*"cost": 22000*

*}*

*]*

*},*

*{*

*"name": "Экскаватор",*

*"model": "CAT 320D",*

*"license\_plate": "В456ГД98",*

*"category": "Строительная",*

*"year": 2018,*

*"service\_history": [*

*{*

*"date": "2022-12-01",*

*"description": "Замена гидравлического насоса",*

*"replaced\_parts": ["Насос"],*

*"mileage": 8000,*

*"cost": 120000*

*}*

*]*

*},*

*{*

*"name": "Погрузчик",*

*"model": "JCB 535-95",*

*"license\_plate": "Е789ЖК150",*

*"category": "Складская техника",*

*"year": 2020,*

*"service\_history": [*

*{*

*"date": "2023-03-15",*

*"description": "Замена аккумулятора",*

*"replaced\_parts": ["Аккумулятор"],*

*"mileage": 4500,*

*"cost": 35000*

*},*

*{*

*"date": "2024-01-22",*

*"description": "Плановое ТО, замена масла и фильтра",*

*"replaced\_parts": ["Масло", "Фильтр"],*

*"mileage": 6000,*

*"cost": 18000*

*}*

*]*

*},*

*{*

*"name": "Гидравлический пресс",*

*"model": "Hidromek HMK 102B",*

*"license\_plate": "З123КЛ99",*

*"category": "Строительная техника",*

*"year": 2019,*

*"service\_history": [*

*{*

*"date": "2023-02-05",*

*"description": "Замена масла в гидросистеме",*

*"replaced\_parts": ["Масло"],*

*"mileage": 5000,*

*"cost": 25000*

*},*

*{*

*"date": "2023-09-15",*

*"description": "Ремонт цилиндра",*

*"replaced\_parts": ["Цилиндр"],*

*"mileage": 7000,*

*"cost": 30000*

*}*

*]*

*}*

*]*

Получим результат

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание