

Григоровский Денис lafr2302



Каталог товаров

[+ Создать товар](#)[Сбросить фильтры](#)[Найдено: 50](#)

Essence Mascara Lash Princess
Beauty

50 MDL

[Просмотр](#)[Редактировать](#)[Удалить](#)

Eyeshadow Palette with Mirror
Beauty

100 MDL

[Просмотр](#)[Редактировать](#)[Удалить](#)

Powder Canister
Beauty

75 MDL

[Просмотр](#)[Редактировать](#)[Удалить](#)

Red Lipstick
Beauty

65 MDL

[Просмотр](#)[Редактировать](#)[Удалить](#)

Red Nail Polish
Beauty

45 MDL

[Просмотр](#)[Редактировать](#)[Удалить](#)

Calvin Klein CK One
Fragrances

250 MDL

[Просмотр](#)[Редактировать](#)[Удалить](#)

Chanel Coco Noir Eau De
Fragrances



Dior J'adore
Fragrances



Dolce Shine Eau de
Fragrances



Каталог товаров

[+ Создать товар](#)

Создание нового товара

Название *

Описание

Цена *

Категория

URL изображения

Активация Windows

Чтобы активировать Windows, перейдите в [Параметры](#).

Активация Windows

Чтобы активировать Windows, перейдите в [Параметры](#).

Essence Mascara Lash Princess



Описание

The Essence Mascara Lash Princess is a popular mascara known for its volumizing and lengthening effects. Achieve dramatic lashes with this long-lasting and cruelty-free formula.

Категория

beauty

Цена

50 MDL

Add to cart

Buy now

Share

REACT

Лабораторная работа №1

Цель: создание базовых компонентов, работа с useState, useEffect, CRUD.

Пример кода и объяснение

```
useEffect(() => {  
  fetchEntities(); },  
[]);
```

→ useEffect выполняется один раз при загрузке страницы и вызывает загрузку сущностей.

```
const [entities, setEntities] = useState([]);
```

→ Создаём локальное состояние, в котором хранится список загруженных сущностей.

```
const addEntity = (newEntity) => {  
  setEntities(prev => [...prev, newEntity]);  
};
```

→ Функция создания сущности — добавляет новый объект в список без перезагрузки страницы.

```
const updateEntity = (upd) => {  
  setEntities(prev => prev.map(e => e.id === upd.id ? upd : e));  
};
```

→ Редактирование — обновление нужной сущности в массиве.

```
const deleteEntity = (id) => {  
  setEntities(prev  
=> prev.filter(e => e.id !== id));  
};
```

→ Удаление сущности — фильтрация списка, удаляем элемент по id.

EntitiesContext.jsx ✓ Загрузка данных через useEffect (п. 1) ✓ CRUD (add, update, delete, getById)

ListPage.jsx ✓ Отображение списка сущностей ✓ Модальное окно просмотра ✓
Удаление

EntityForm.jsx ✓ Создание и редактирование товара

CreateEditPage.jsx ✓ Реализация «создать» и «редактировать»

EntityCard.jsx ✓ Отображение карточки сущности

EntityPage.jsx ✓ Просмотр сущности и удаление

App.jsx ✓ Маршрут на список + формы (базовый роутинг для CRUD)

Лабораторная работа №2

Цель: поиск, фильтрация, сортировка, useMemo, оптимизация.

Примеры кода

```
const [q, setQ] = useState("");
```

→ Состояние строки поиска.

```
const filtered = useMemo(() => {  
  return entities.filter(e => e.title.includes(q));  
},  
[entities, q]);
```

→ useMemo кеширует результат фильтрации, чтобы не пересчитывать при каждом рендере.

```
const categories = useMemo(() => {  
  return Array.from(new Set(entities.map(e => e.category)));  
},  
[entities]);
```

→ Получение уникальных категорий для выпадающего списка.

```
if (loading) return <div>Загрузка...</div>;
```

→ Отображение состояния загрузки.

```
if (!loading && filtered.length === 0) return <div>Ничего не найдено</div>;
```

✓ Поиск ✓ Фильтры (категория, цена) ✓ Сортировка ✓ useMemo ✓ Лоадер,
ListPage.jsx пустой результат ✓ «Сбросить фильтры» ✓ «Найдено X»

EntitiesContext.jsx ✓ Флаг loading ✓ Флаг error

Лабораторная работа №3

Цель: Context API, маршрутизация, использование useParams и useNavigate.

Примеры кода

```
const EntitiesContext = createContext(null);
```

→ Создание контекста для хранения данных.

```
<EntitiesContext.Provider value={{entities, addEntity, updateEntity}}>
  {children}
</EntitiesContext.Provider>
```

→ Провайдер передаёт глобальные данные во всё приложение.

```
const { id } = useParams();
```

→ Получение параметров из URL — используется на странице деталей и редактирования.

```
navigate('/edit/' + entity.id);
```

→ Программная навигация для перехода между страницами без перезагрузки.

```
<Routes>
  <Route path='/entity/:id' element={}<EntityPage/>}/> </Routes>
```

→ Маршруты приложения — определяют, какая страница показывается по URL.

EntitiesContext.jsx ✓ Глобальное хранилище данных ✓ Методы fetchEntities, getByid, addEntity, updateEntity, deleteEntity

main.jsx ✓ Оборачивание приложения в EntitiesProvider ✓ BrowserRouter

App.jsx ✓ Создание маршрутов /, /create, /edit/:id, /entity/:id

EntityPage.jsx ✓ Использование useParams ✓ Использование useNavigate ✓
Получение данных из контекста

CreateEditPage.jsx ✓ Получение id через useParams ✓ getByid из контекста ✓
Сохранение через контекст