Московский авиационный институт

(национальный исследовательский университет)

**Институт информационных технологий и прикладной математики**

Кафедра вычислительной математики и программирования

**Курсовой проект по курсу ООП**

Работу выполнил:

Группа: М8О-203Б-20

ФИО: Большаков Максим Владимирович

Руководитель: Кузнецова С. В.

Дата:  
Оценка:

МАИ 2021

Оглавление

[1. Тема работы 1](#_Toc91182619)

[2. Задача 1](#_Toc91182620)

[3. Программное обеспечение 1](#_Toc91182621)

[4. Аппаратное обеспечение 1](#_Toc91182622)

[5. Описание используемой технологии разработки. Достоинства и недостатки 2](#_Toc91182623)

[6. Создание и настройка сайта 7](#_Toc91182624)

[7. Структура проекта web-приложения 8](#_Toc91182625)

[8. Контроллеры, модели, представление 8](#_Toc91182626)

[9. Структура сайта 9](#_Toc91182627)

[10. Тестирование сайта/демонстрационные примеры 10](#_Toc91182628)

[11. Выводы 17](#_Toc91182629)

# Тема работы

Создание веб-приложения на базе сочетания серверных сервисов с использованием технологии ASP.NET Core, и клиентских одностраничных приложений (single page applications, SPA), проектируемых с применением современных технологий фронтенд-разработки, таких как Angular, React.

# Задача

Разработать веб-приложение, позволяющее создавать и отправлять на серверную часть составленный список продуктов, реализовать динамическое перемещение, добавление и удаление товаров (т.е. без взаимодействия с серверной частью), а также API для связи с клиентской частью посредством таких запросов, как “отправить список на сервер”, “удалить с сервера” и “вернуть данные”, серверную часть реализовать в соответствии с паттерном MVC.

# Программное обеспечение

Семейство ОС: Windows

Операционная система: Windows 10 для образовательных учреждений

Среда разработки: Microsoft Visual Studio Enterprise 2022 Preview Версия: 17.1.0 Preview 1.1, Visual Studio Code, Adobe Photoshop 2022.

# Аппаратное обеспечение

Модель ноутбука: Lenovo ThinkPad e14 gen2

Процессор: AMD Ryzen 7 4700U with Radeon Graphics 2.00 GHz

ОП: 24,0 ГБ

Дисковые устройства:

SSD KINGSTON\_SA 256GB

SSD SAMSUNG\_MZALQ 512GB

Первый монитор: Wide viewing angle & High density FlexView Display 1920x1080

Второй монитор: встроенный в ноутбук

# Описание используемой технологии разработки. Достоинства и недостатки

Для реализации курсового проекта были использованы следующие технологии:

1. Asp.Netcore

*ASP.NET Core* – это инструмент для создания веб-приложений для платформы *Microsoft.NET Core*. Он широко используется для корпоративных или облачных приложений и подходит для полнофункциональной или серверной разработки *MVC*. Главная идея всех MVC – это *разделение логики приложения* на отдельные относительно независимые компоненты: бизнес логику, сервисы, представления, маршрутизацию, а также вводит дополнительные средства для работы с расширениями и конфигурированием. Таким образом asp.net core – это мощный инструмент, который обладает кроссплатформенностью, модульностью, открытым исходным кодом и многим другим. В основе специальности лежит удобный и надежный язык C#.

Плюсы

* Скорость запуска
* Расширение для браузера помогающее отлаживать код
* Кроссплатформенность
* Opensource
* Перспективность

Минусы

* Не такое большое количество уже написанных библиотек
* Меньше возможностей по сравнению с .Net Framework

1. React

*React* — это инструмент для создания пользовательских интерфейсов. Его главная задача — *обеспечение вывода на экран того, что можно видеть на веб-страницах*. React значительно облегчает создание интерфейсов благодаря разбиению каждой страницы на небольшие фрагменты.

*Компонент React* — это, если по-простому, участок кода, который представляет часть веб-страницы. Каждый компонент — это JavaScript-функция, которая возвращает кусок кода, представляющего фрагмент страницы.

Плюсы

#### Виртуальная объектная модель документа (эффективность, высокая производительность)

#### *Повторное применение компонентов*

#### Нисходящий поток данных Меньше ошибок, легче отладка

#### Огромное сообщество

#### *Браузерные инструменты React-разработчика*

Минусы

#### Плохая документация

#### Запутанный синтаксис JSX

#### Фокусировка на пользовательском интерфейсе

1. Redux

*Redux* — это инструмент для управления состоянием приложения. Построен на принципах технологии Flux и функционального программирования. Создан компанией FaceBook, но вопреки распространенному мнению может использоваться не только в связке с React, но также и с другими фреймворками/библиотеками.

Плюсы

#### Быстрый

#### Можно иметь много reducer

#### Эффективность

#### Быстрота разработки

Минусы

#### Лучше не использовать больше одного store

#### Действует только по одной схеме

1. JQuery

*JQuery* — библиотека JavaScript, содержащая в себе готовые функции языка JavaScript, все операции jQuery выполняются из кода JavaScript.

Библиотека jQuery производит *манипуляции с html-элементами, управляя их поведением и используя DOM* для изменения структуры веб-страницы. При этом исходные файлы HTML и CSS не меняются, изменения вносятся лишь в отображение страницы для пользователя.

Плюсы

#### Кроссплатформенность

#### Вариативность

#### Расширения

#### Сообщество

#### Краткость записи

Минусы

#### загружается как дополнительный ресурс

1. Less

[*LESS*](http://lesscss.org/) — это язык динамических таблиц стилей, который расширяет CSS и, также как и SASS, характеризуется динамическим поведением, примером чего могут служить переменные, примеси, операции и функции.

Это позволяет составлять CSS-код более просто, используя только примеси, функции, другие динамические элементы и их комбинации. LESS также позволяет повысить читаемость и организацию CSS-кода с помощью импорта, вложенных правил и комментариев с расширением .less.

Плюсы

#### простой и удобный

#### написан на JS

#### быстрее, чем Sass

Минус:

#### нет массивов, циклов и условий

1. Webpack

*Webpack* — это инструмент, позволяющий скомпилировать, например, JavaScrit модули в единый JS-файл. Webpack также известен как сборщик модулей. При большом количестве файлов он создает один объемный файл (или несколько файлов) для запуска вашего приложения.

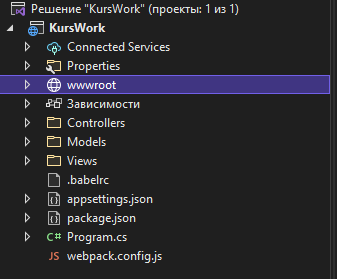
**Плюсов:**он отлично подходит для работы с *одностраничными приложениями*. Также вебпак может осуществлять продвинутое разделение кода. Из-за этих и других преимуществ он является одним из наиболее популярных инструментов JS-разработки на данный момент.

**Минусы:** немного сложно разобраться в его работе, часть документации устарела из-за большого количества изменений в обновлениях.

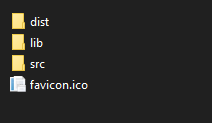
1. Babel

*Babel* — это транслятор, который, в основном, используется для преобразования конструкций, принятых в свежих версиях стандарта ECMAScript, в вид, понятный как современным, так и не самым новым браузерам и другим средам, в которых может выполняться JavaScript. Babel, кроме того, умеет преобразовывать в JavaScript и JSX-код, используя [@babel/preset-react](https://babeljs.io/docs/en/babel-preset-react).

# Создание и настройка сайта

Для создания серверной части использовалась IDE Visual Studio, для сбора и настройки клиентской части использовался npm (пакетный менеджер) вместе с node.js (программная платформа, основанная на движке V8, превращающая JavaScript из узкоспециализированного языка в язык общего назначения). Создание веб-приложения производитесь отдельно для серверной части и отдельно для внутренней части. Для сборки всех .js .css файлов использовался такой инструмент как webpack, для преобразования .jsx и .less файлов .js и .css использовался инструмент babel. Расположение файлов в проекте следующее:

в папке Controls располагаются контроллеры, в Models – модели, в Views – располагается визуальная составляющая, точнее базовые файлы razer pages;

в папке wwwroot находятся файлы относящиеся к клиентской части, в папке node\_modules -–пакеты загруженные для клиентской части webpack.config.js конфиг webpack; .babelrc –конфиг babel; package.json– конфигурационный файл для npm, в Properties находится конфиг для backend, в директории wwwroot, расположены папки dist – в которую собираются все готовые файлы .js, .css, src -папка с исходными файлами, lib – директория с некоторыми библиотеками.

# Структура проекта web-приложения

Все react приложение разбито на множество компонент, каждую компоненту описывать достаточно долго и выйдет большой объем, поэтому здесь будут описаны основные составляющие:

1. Основной класс App – с него начинается работа всего react приложения;
2. InteractionServer – это компонента строит панель взаимодействия с сервером (отправка на сервер, отправка с сервера, удаление всех данных везде), также из нее делаются запросы, ответ на который передается в reducer.
3. CreateNewShoppingList – выводит панельку для создания списка, вызывает события создания списка.
4. ShoppingList – основная компонента, содержащая важные компоненты такие как сами элементы списка(purchases), а также панели для фильтрации (FiltersPurchases), сортировки(sortPurchases) и создания(addPurchase).
5. В приложении используется два store компоненты для обеспечения изменения состояния программы, находятся в папке stores.

Серверная часть основана на паттерне MVC, поэтому ее основные компоненты можно разбить на три группы Model, Controls и Views, расположенные по соответствующим папкам.

# Контроллеры, модели, представление

1. К Model относятся DataBase – класс в котором определено небольшое хранилище данных, основанное на списке. ListProducts – единица, определяющая один список. Product – соответственно класс описывающий структуру продукта, а также SecialOffers – класс, определяющий вид специальных предложений.
2. К Controls относятся HomeController который позволяет получить главную страницу. В ProductsListsController написано основное api позволяющее взаимодействовать клиенту с сервером.
3. И наконец к Views относится единственный файл Home\index.cshtml (т.к. по заданию одностраничное приложение), содержащий базовый шаблон программы.

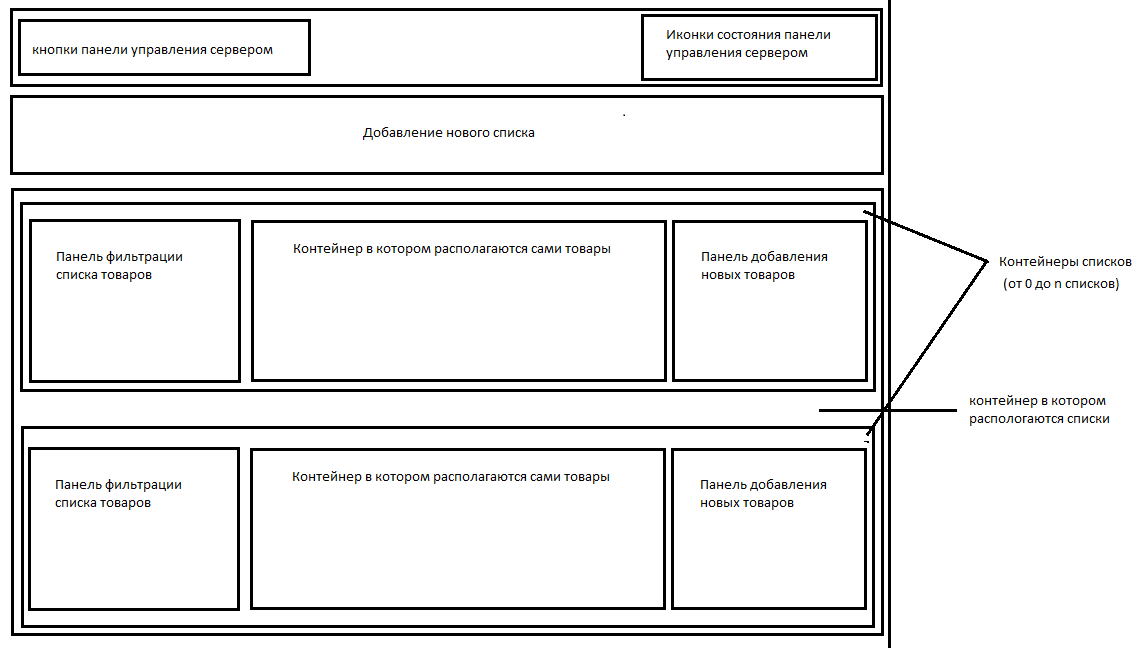
# Структура сайта

Внутренняя структура

Внутренняя структура сайта состоит из одной страницы с веб приложением, а также из api методов:

\ и \home - основная страница сайта, на которой находится веб приложение

\api\ProductsLists\ - реализовано для get и post запросов

\api\ProductsLists\-1 – реализовано для удаления всех данных на стороне сервера

Внешняя структура сайта:

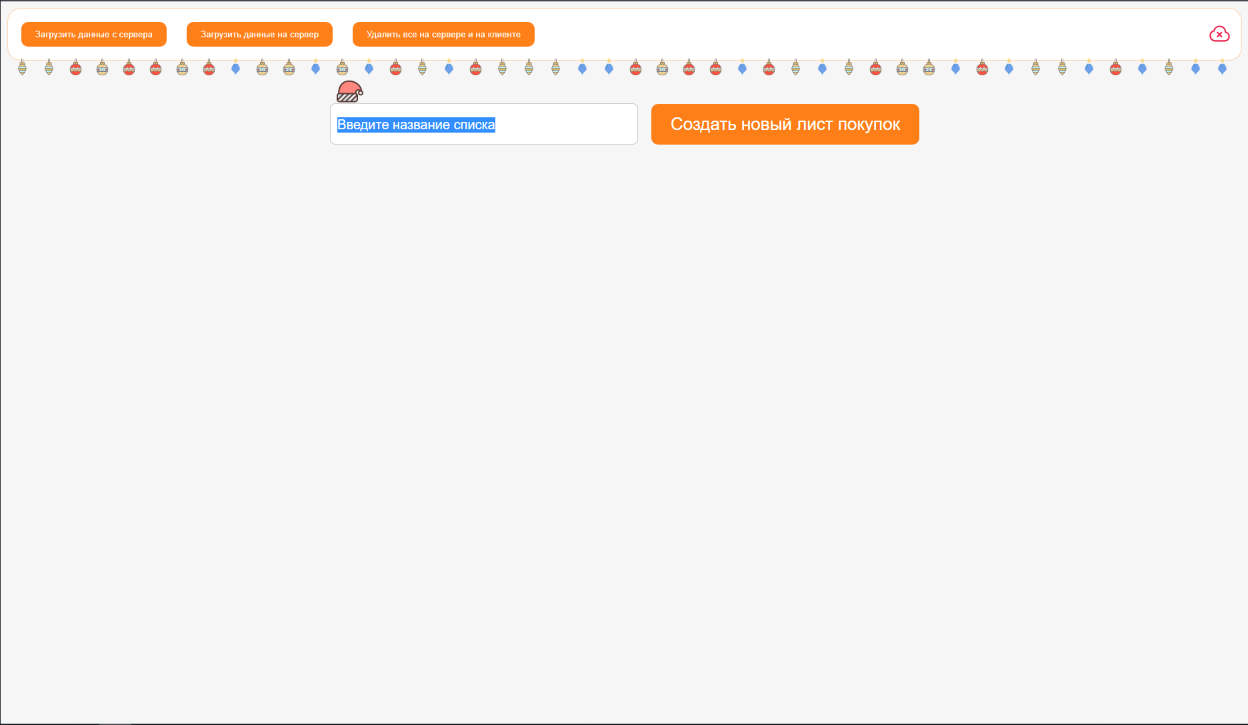
В верхней части окна располагается панель взаимодействия с сервером, в которой также находятся кнопки в левой части и в правой иконки, показывающие состояние сервера.

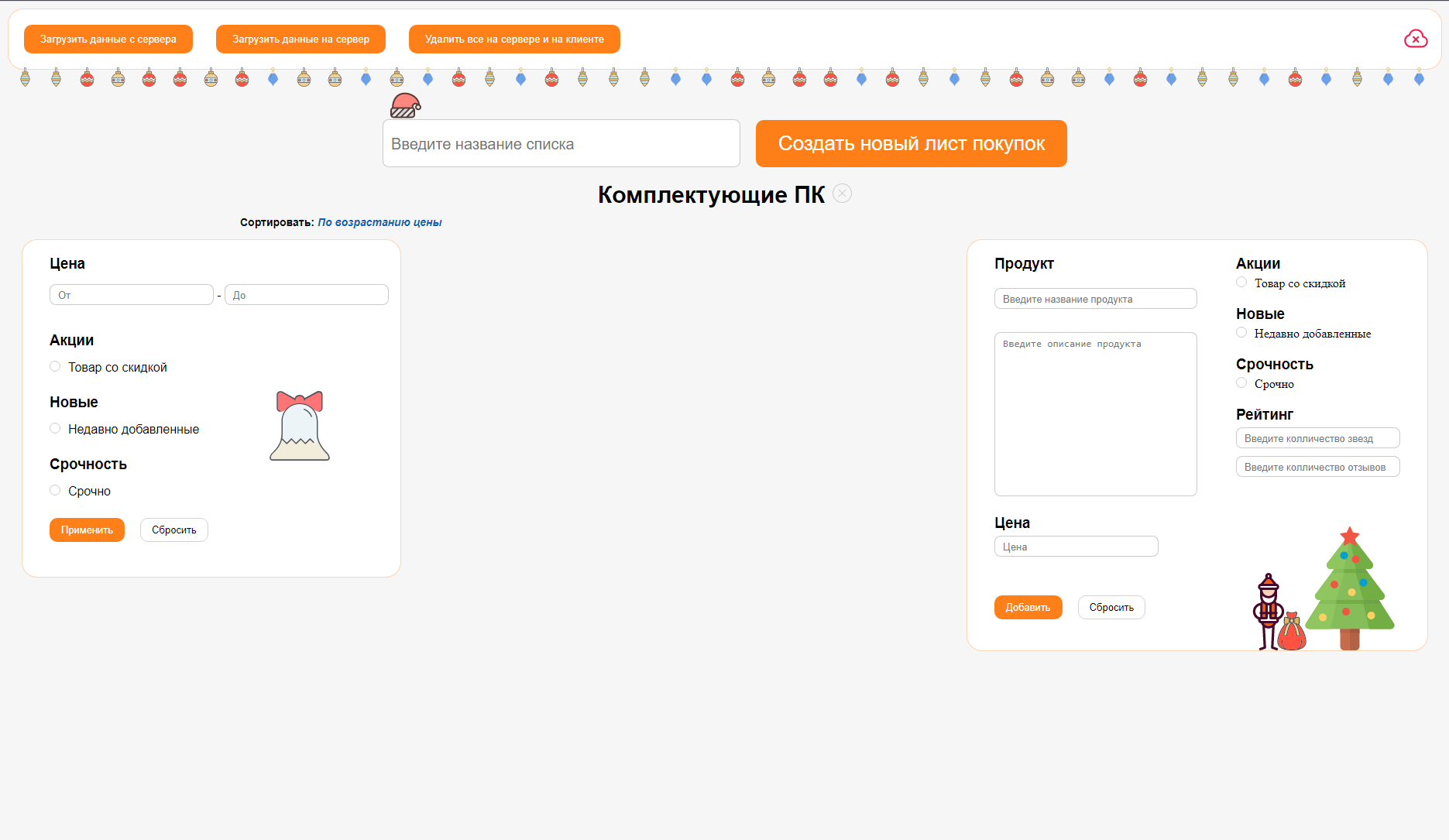
Под панелью находится часть, отвечающая за создание нового списка.

Опускаясь ниже, можно встретить контейнер списков, в котором располагаются все списки.

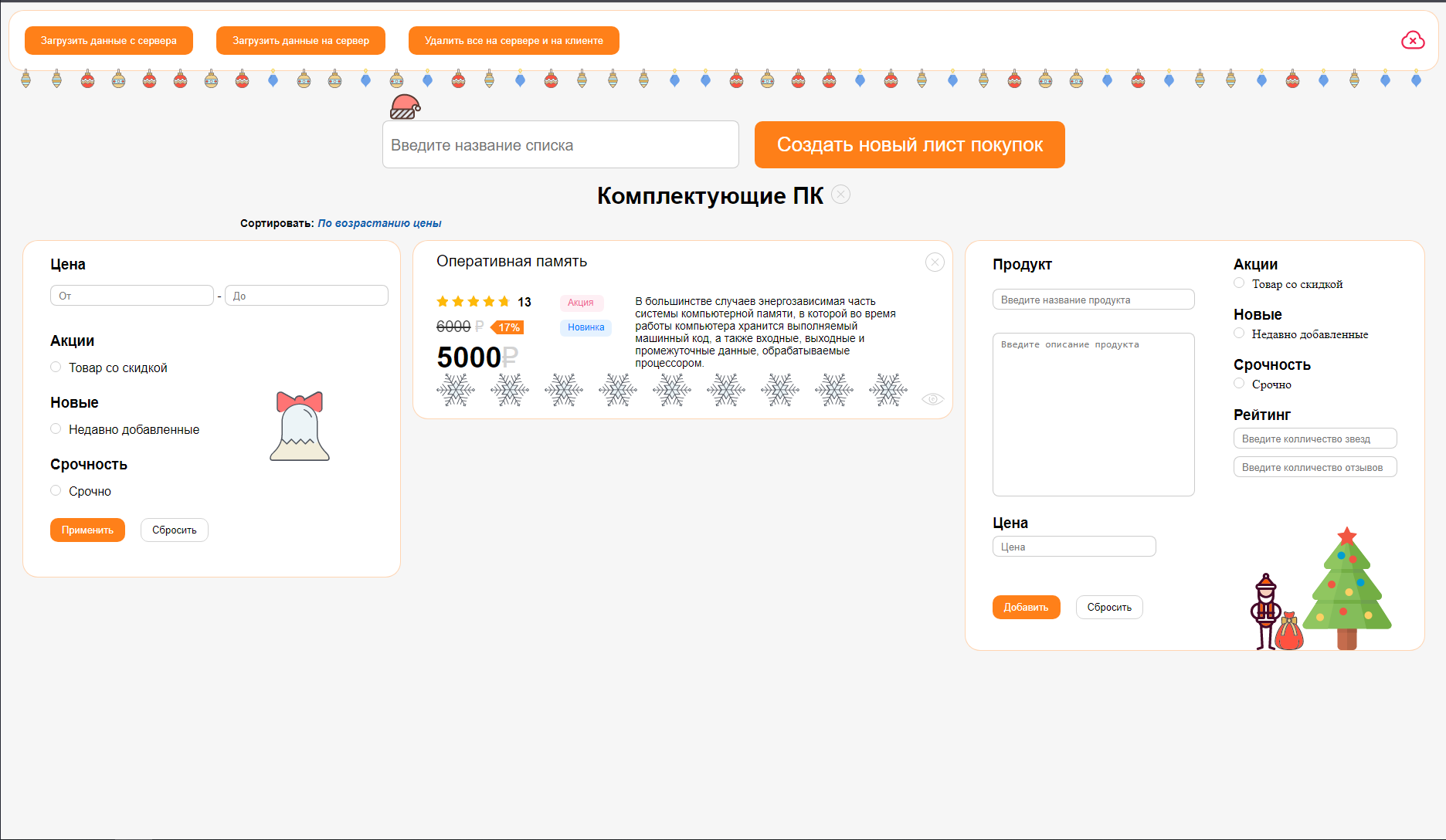
В самом списке находится панель фильтрации и сортировки товаров, контейнер, содержащий сами товары и панель добавления новых товаров.

# Тестирование сайта/демонстрационные примеры

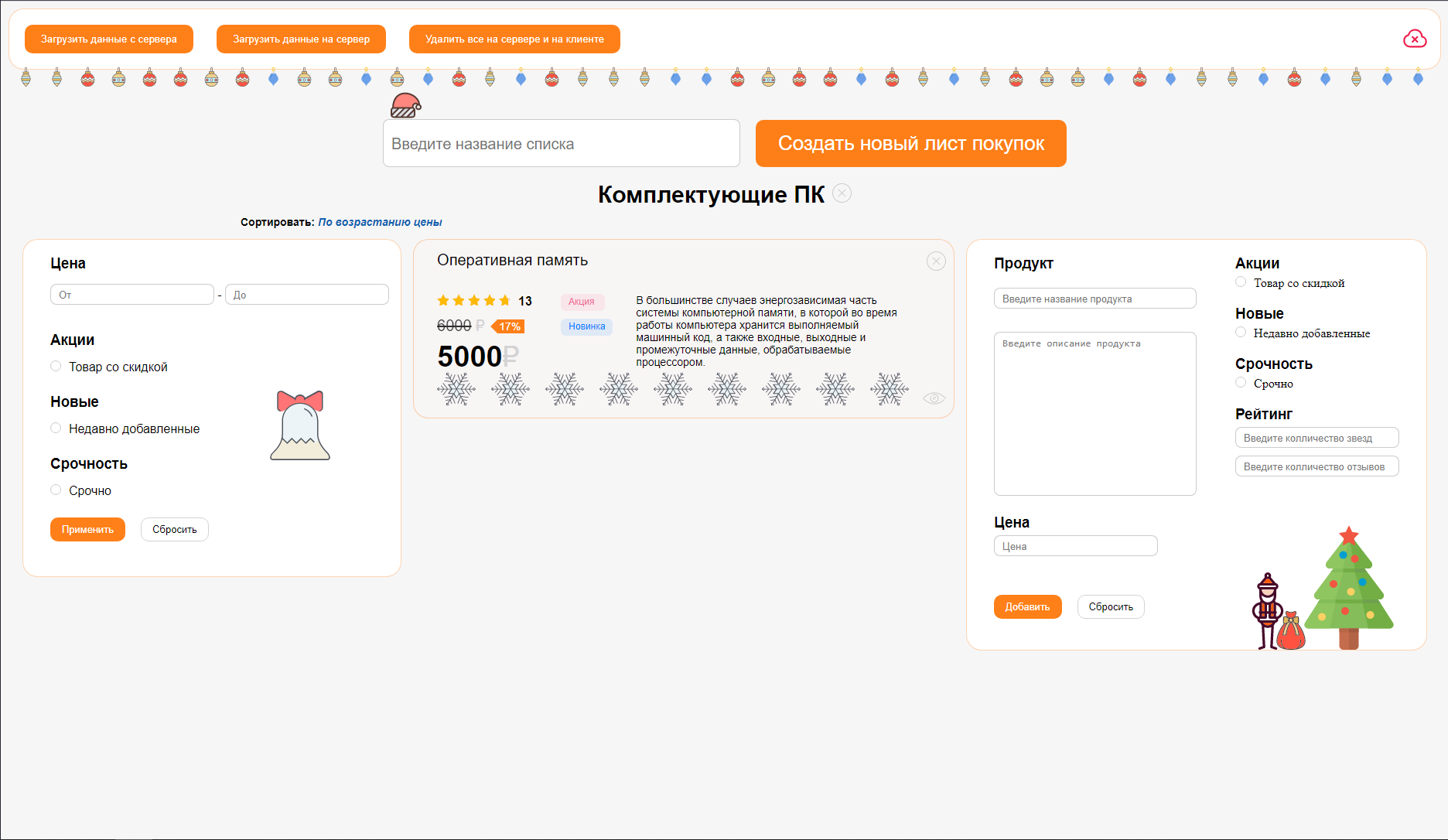
Главная страница выглядит данным образом, справа иконка на панели инструментов показывает, что сервер и текущее состояние не синхронизированы. Для создания списка необходимо ввести название списка и нажать кнопку создания.

После создания списка сайт выглядит следующим образом

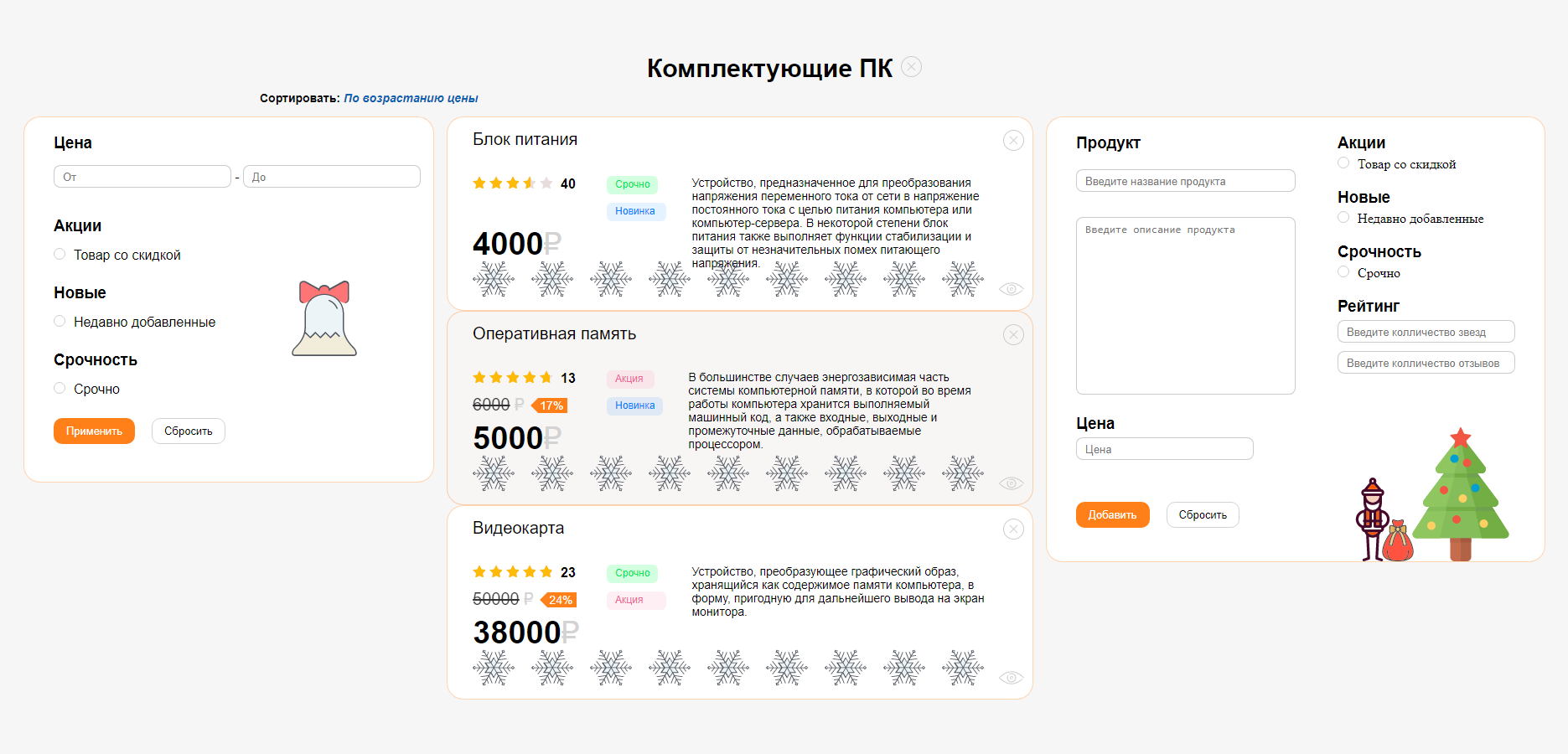
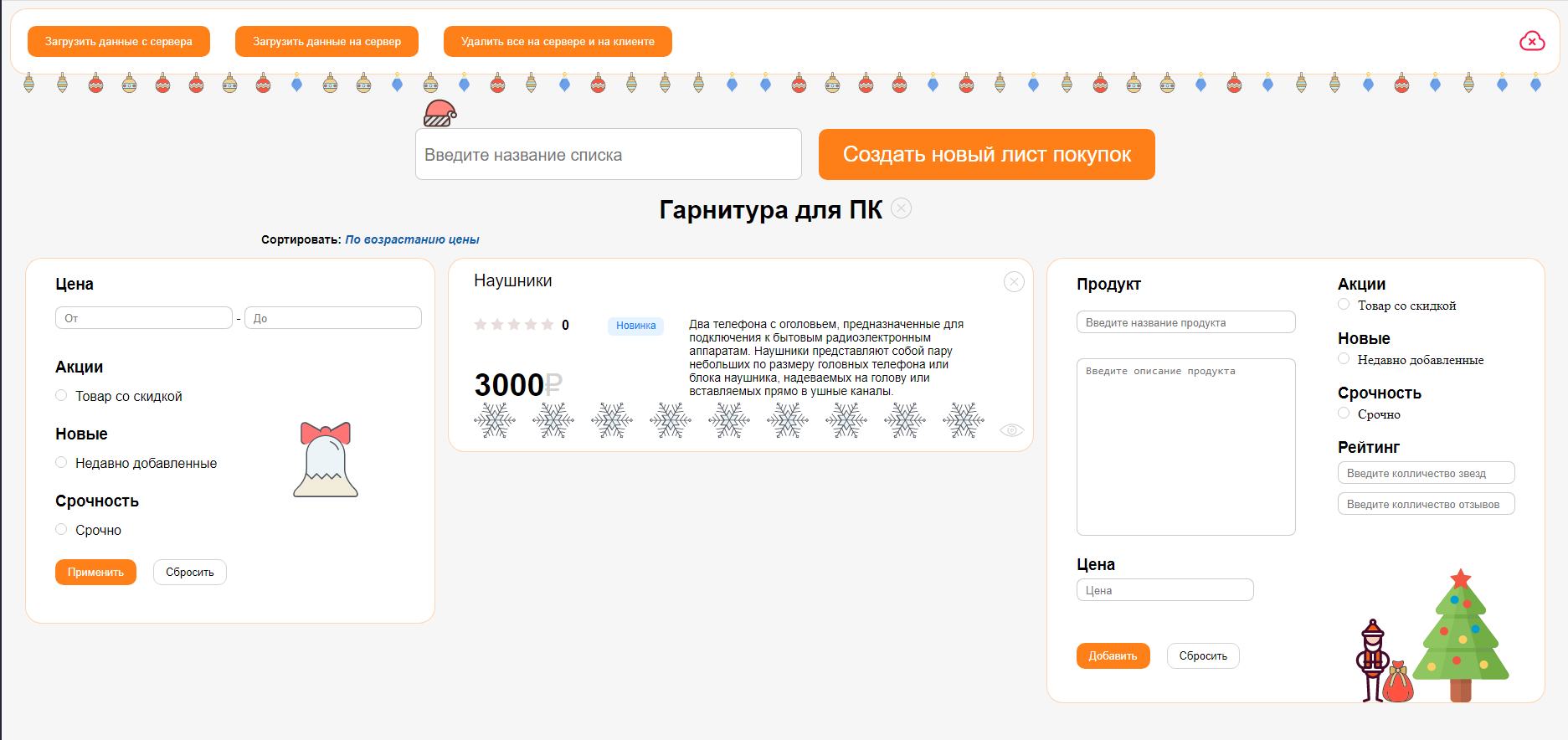
Справа есть панель создания продуктов, необходимо ввести название и цену, а также можно указать прочую информацию о товаре. Результат после добавления товара



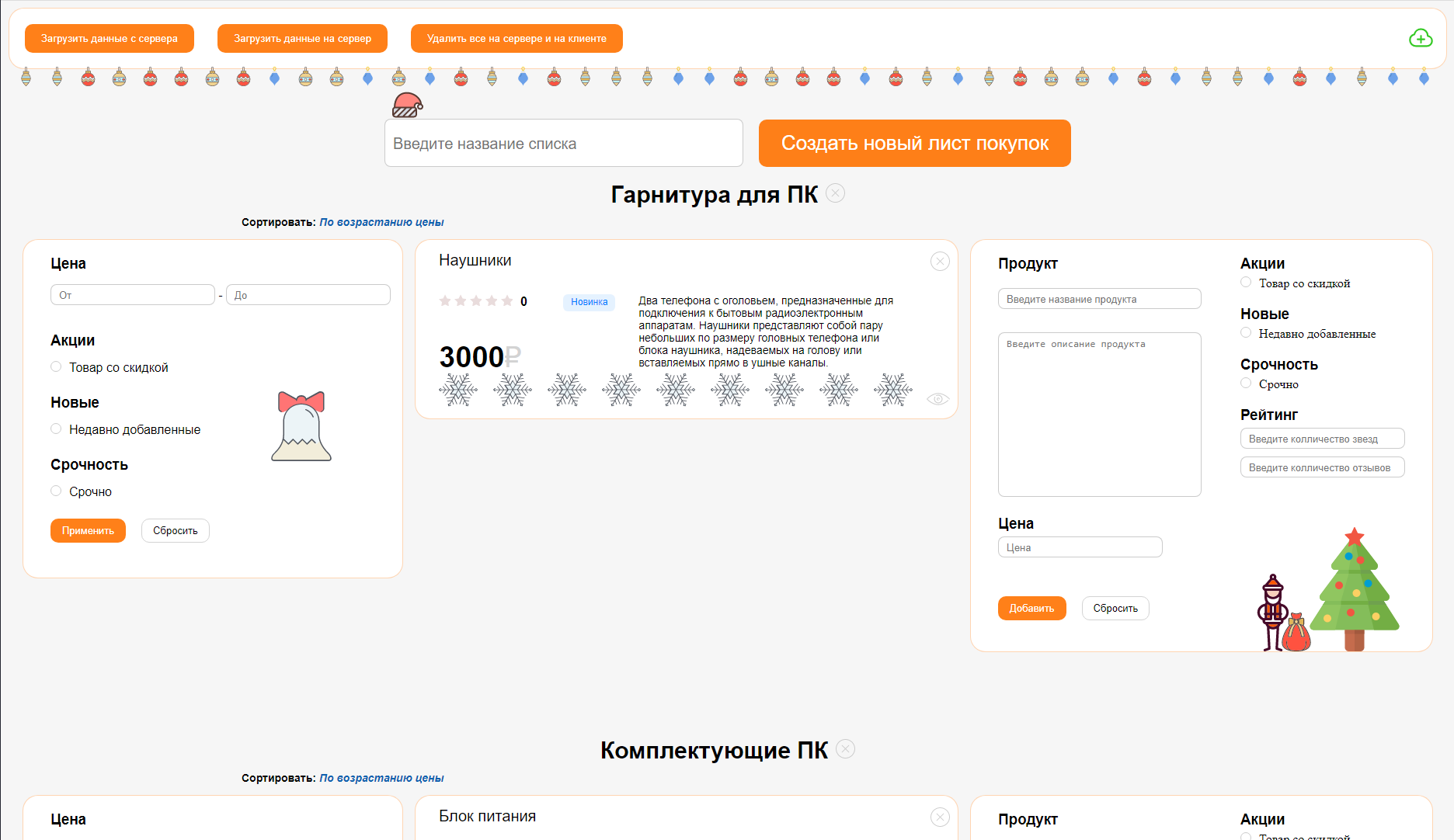
Данный товар можно пометить, нажав на значок глаза в левой нижней части контейнера товара.



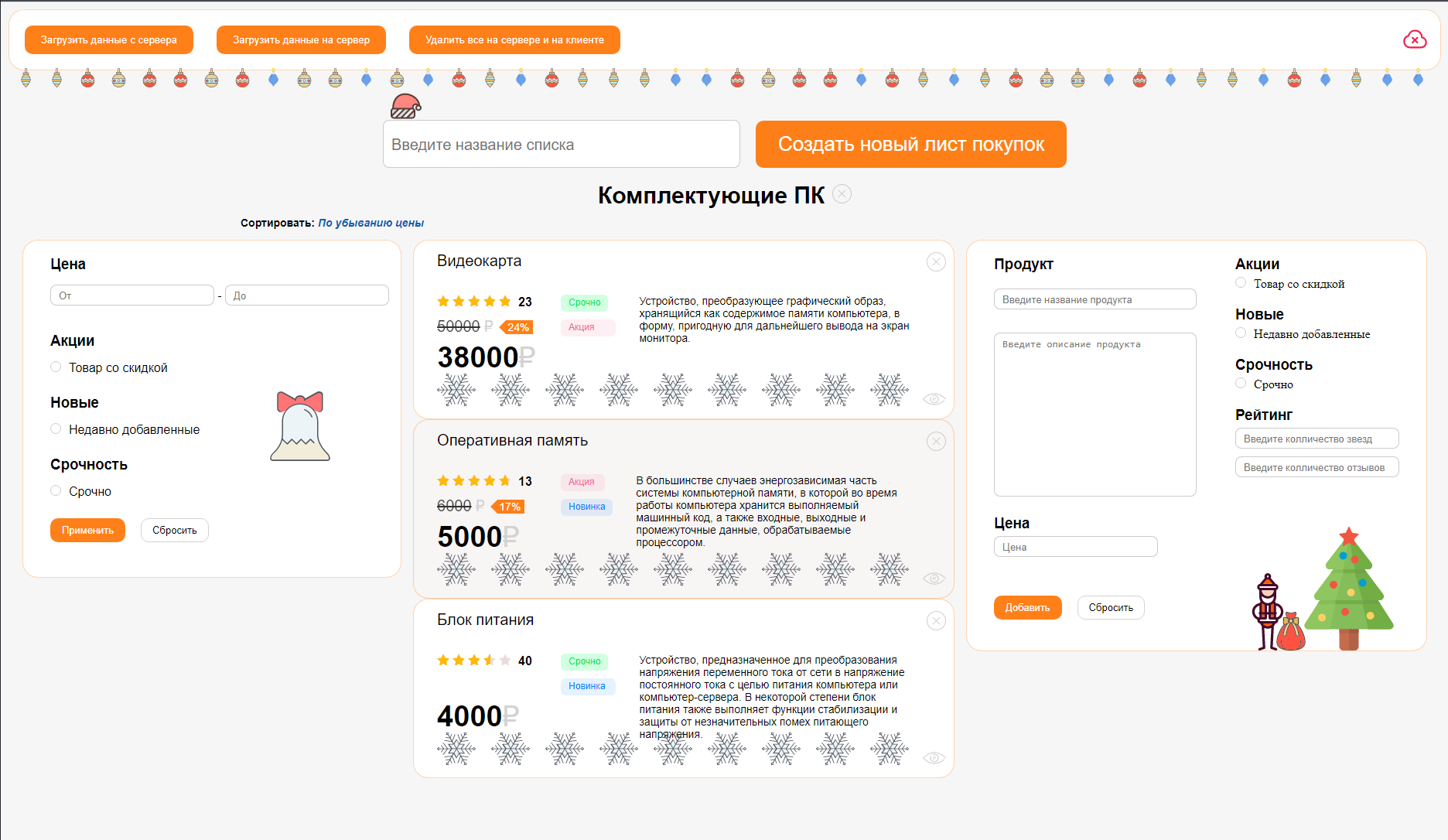
Для наглядности добавим еще пару товаров и создадим еще один список.



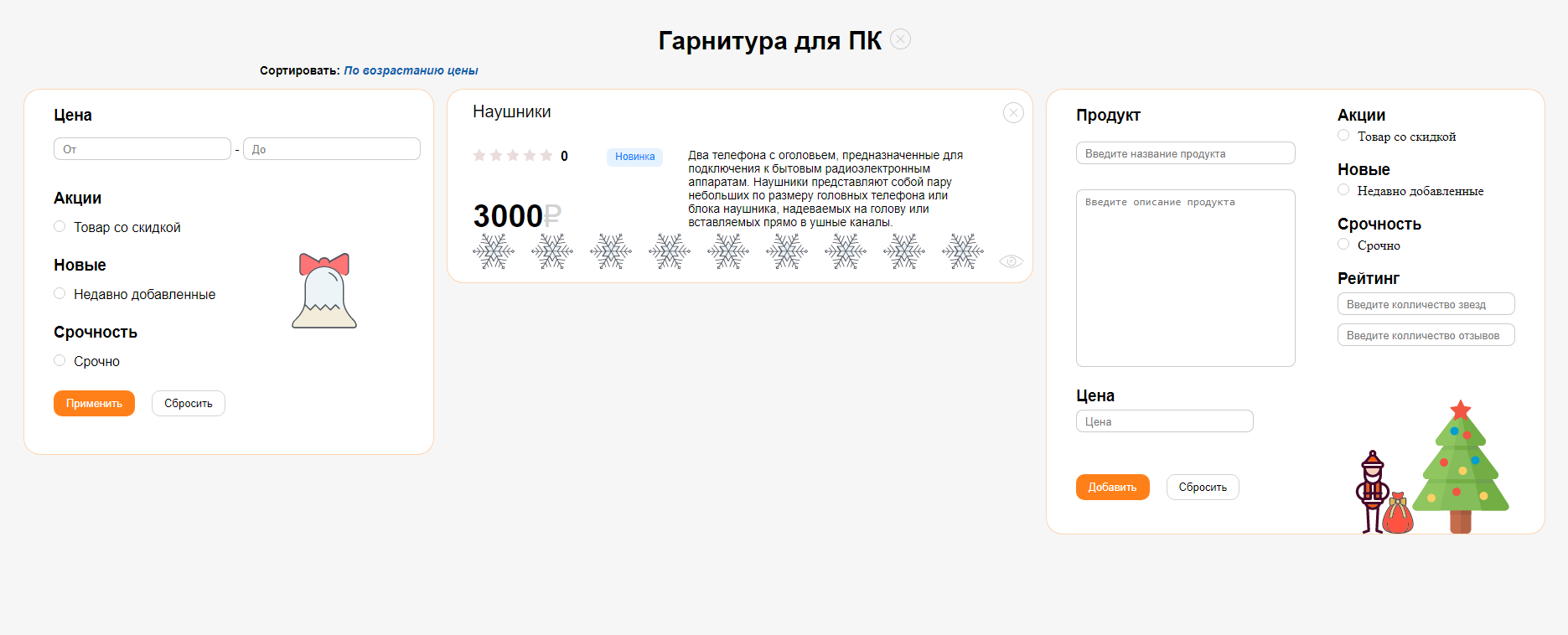
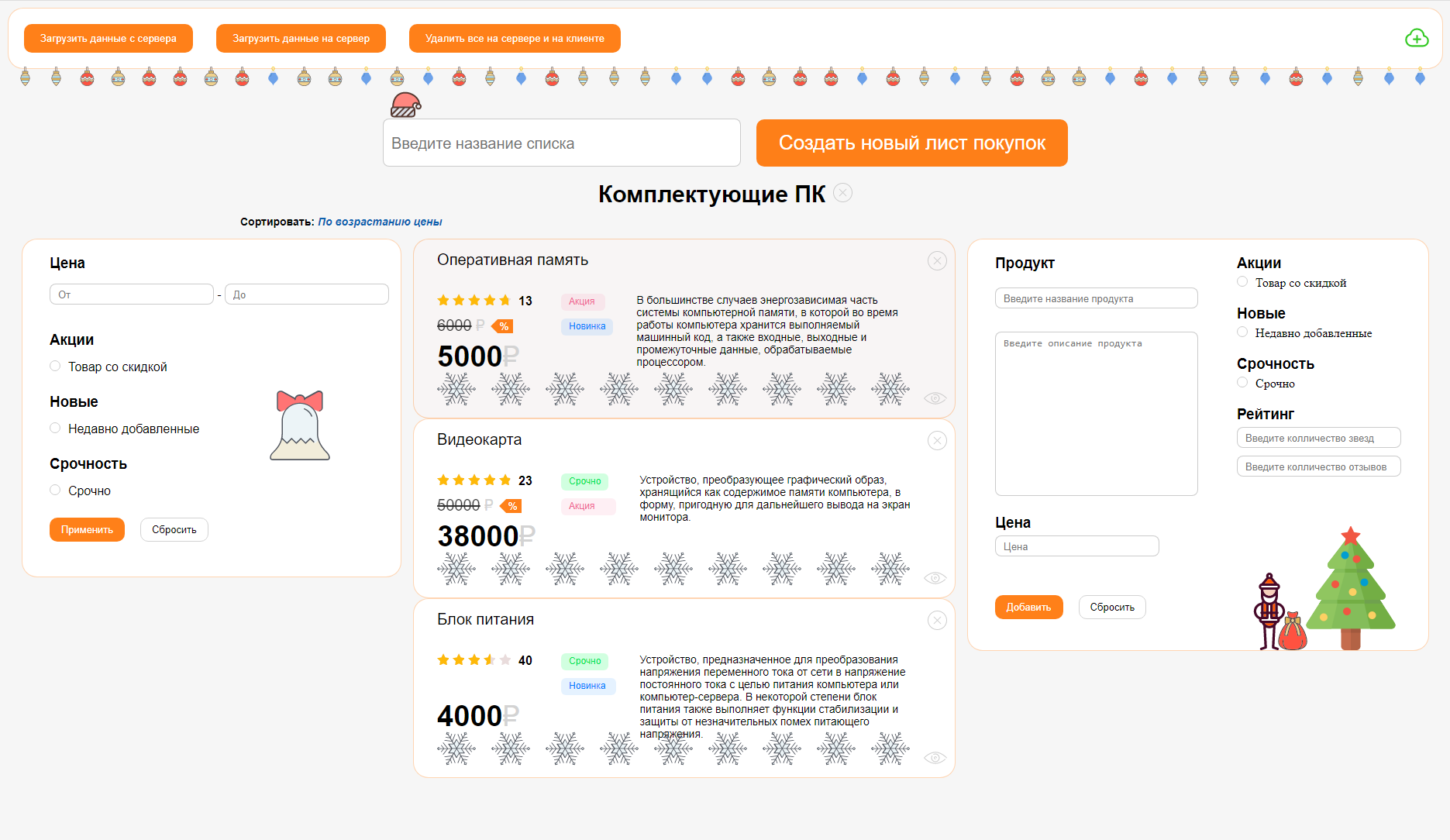
На данный момент значок синхронизации горит красным цветом, что показывает, что данные на сервере и на клиенте отличаются, поэтому необходимо “загрузить данные на сервер”. После нажатия красная иконка стала зеленой и крестик стал плюсиком.



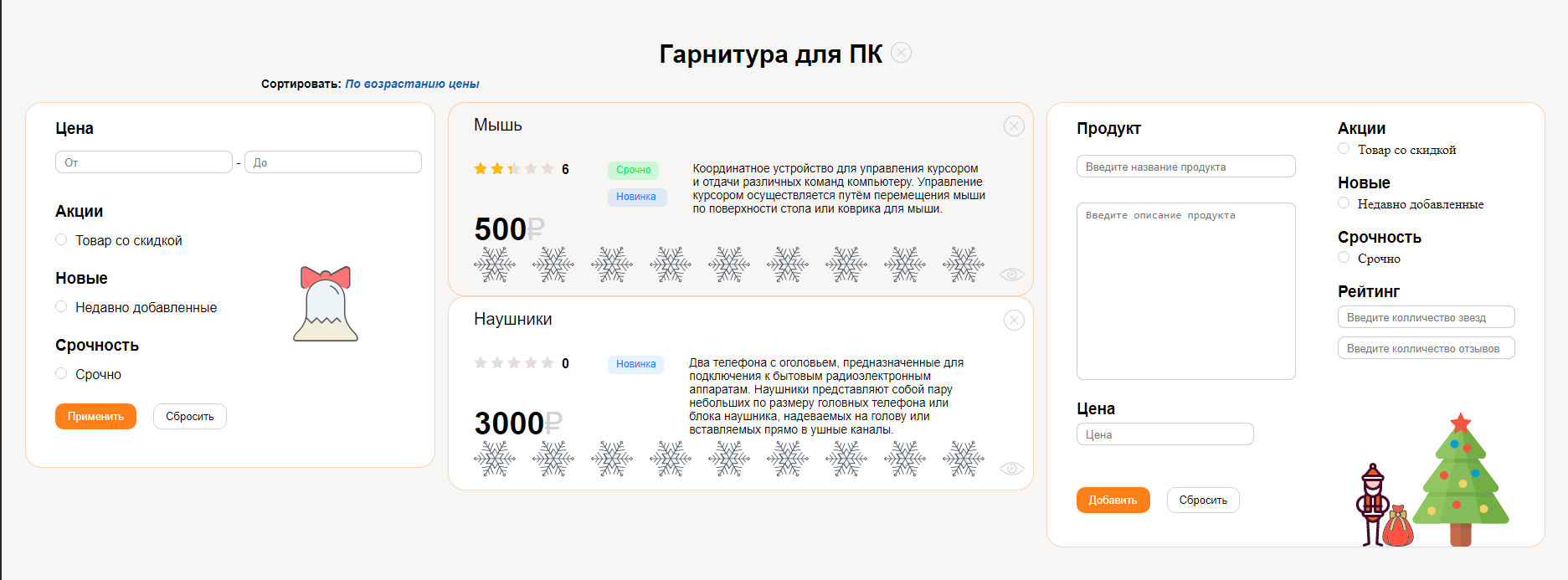
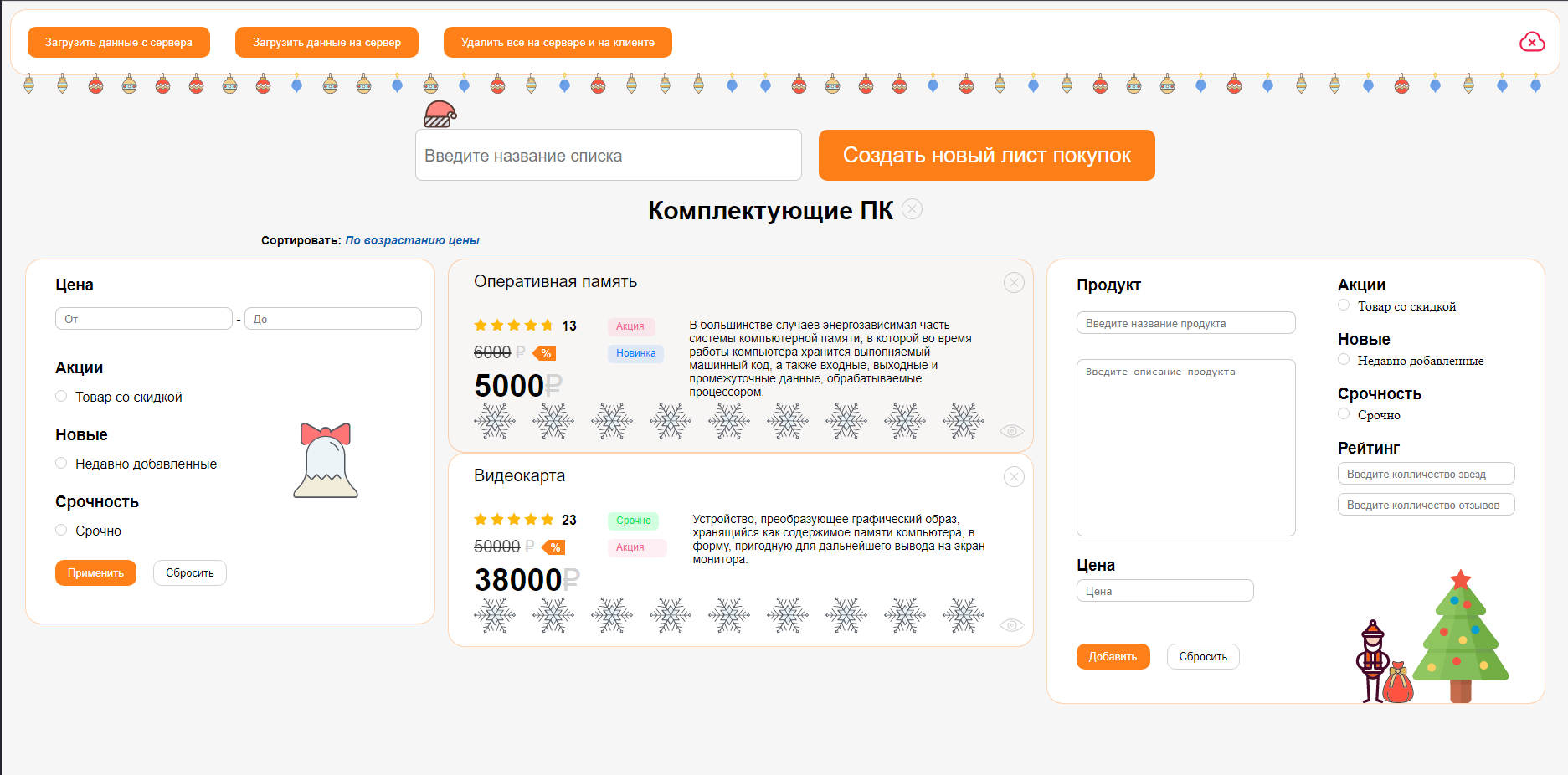
Для демонстрации загрузки данных удалим список “Гарнитура к ПК” и отсортируем “Комплектующие для ПК” по убыванию цены.



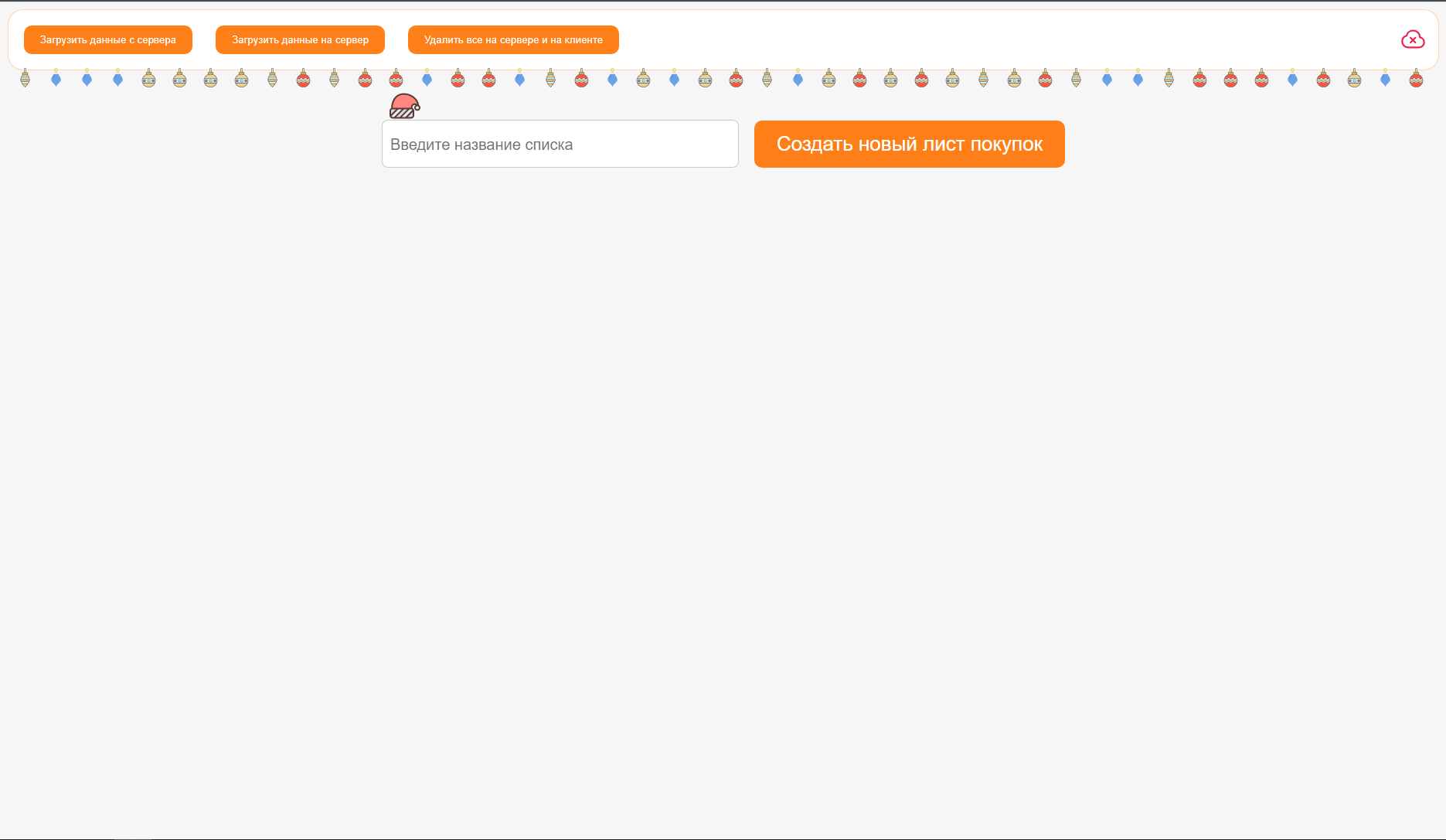
Значок синхронизации стал опять красным, для восстановления списка из сервера нажмем на кнопку “Загрузить данные с сервера”.



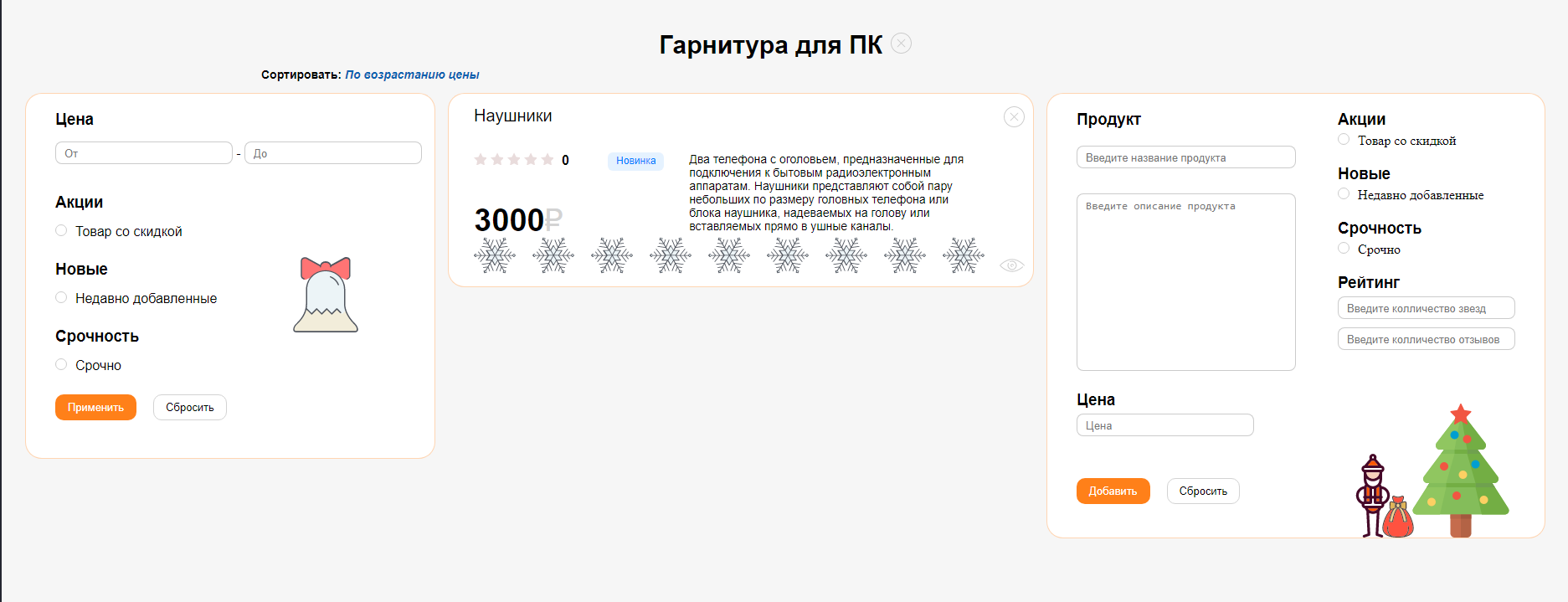
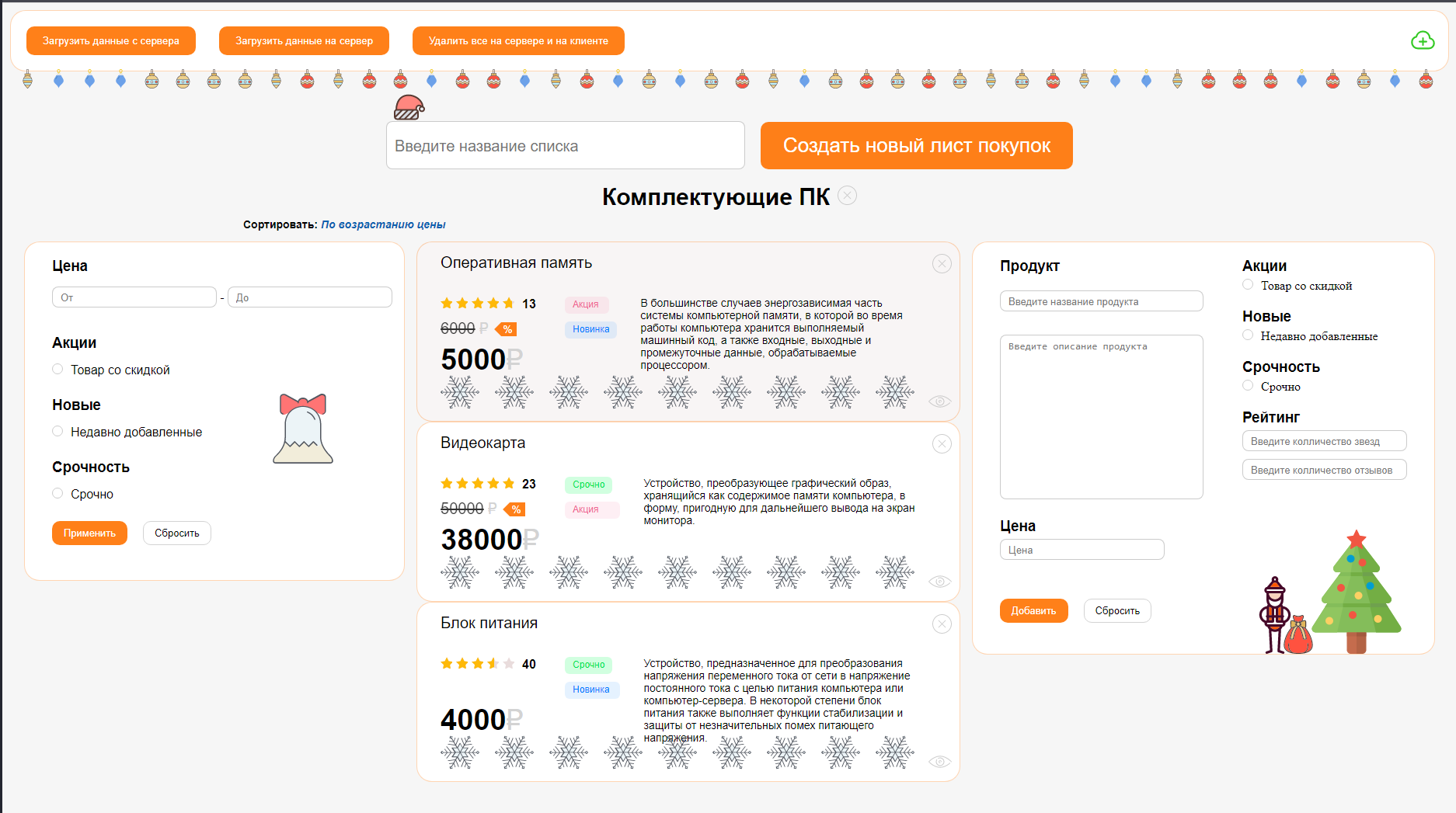
Еще раз проведем тест: добавим в “Гарнитура для ПК” объект и пометим его, из “Комплектующие для ПК” удалим блок питания.



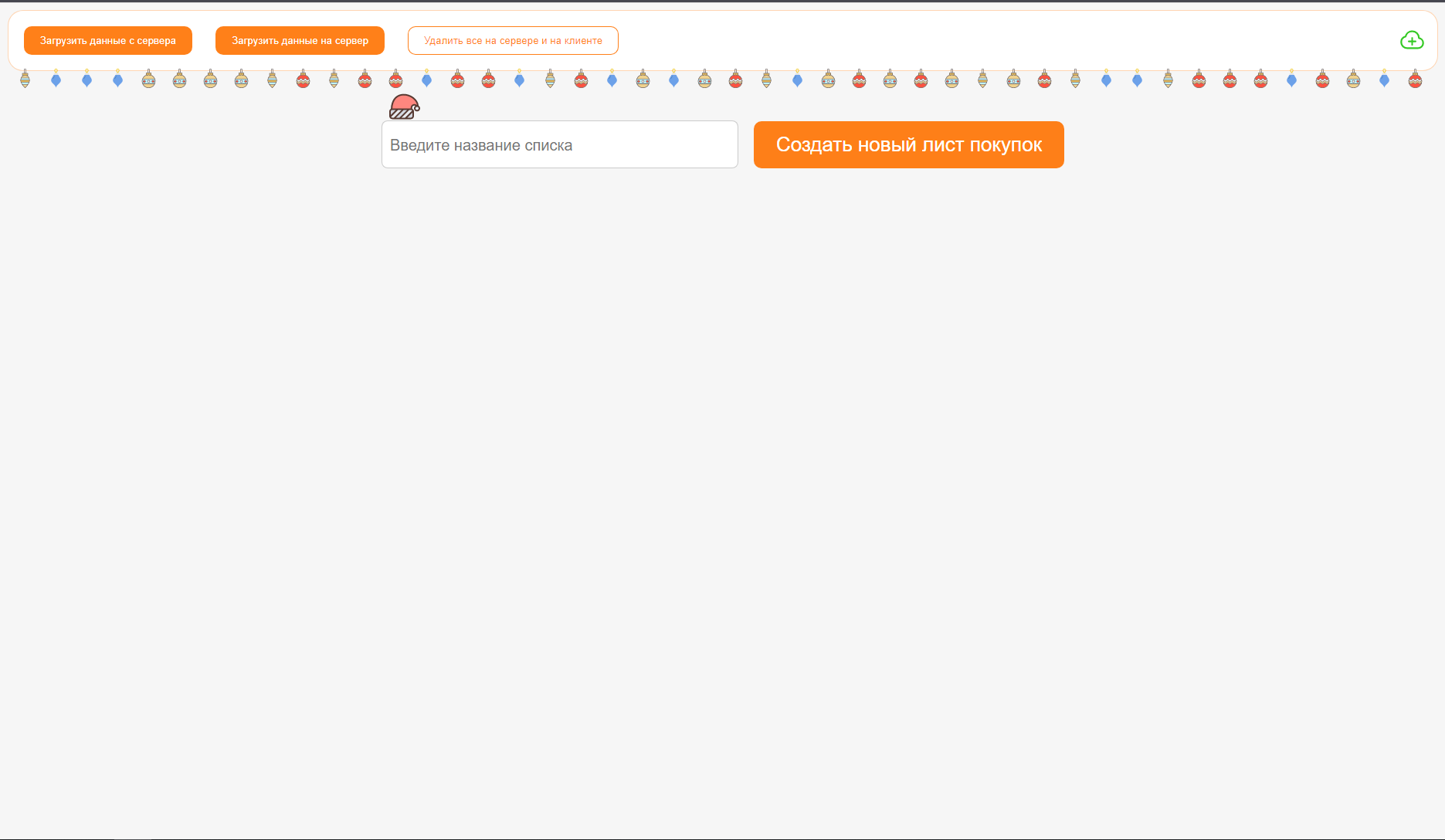
Синхронизируем и перейдем на другую вкладку браузера



Загрузим данные с сервера



И под конец нажмем на кнопку “Удалить все на сервере и клиенте”



# Выводы

Задача данного курсового проекта потребовала использовать множество новых технологий для создания как backend, так и frontend частей. Для backend я использовал ASP.NET с использованием таких паттернов программирования как MVC и Singleton, т.к. это необходимо условиями проектирования, ведь при одностраничном приложении (один вид), может быть множество контроллеров и моделей, а именно эти паттерны позволяют это реализовать. все необходимые объекты были успешно реализованы: Модель, Контроллер и Вид. Таким образом выполнить задачу удалось. Для frontend преимущественно использовалась библиотека react, которая позволяет делать эффективные приложения, т.к. не нужно обновлять неизменяемые компоненты. Также в следствие того, что для создания используется синтаксис html позволяет быстрее разрабатывать web приложения. Вместе с react использовался redux, что позволило достаточно просто организовать событийную модель. Таким образом считаю, что поставленная задача была достигнута.