Министерство науки и высшего образования Российской Федерации ФГБОУ ВО

«БАЙКАЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Кафедра математических методов и цифровых технологий

Междисциплинарная курсовая работа

«Разработка программ»

**«Веб-приложение контроля срока годности продуктов с использованием Computer Vision»**

Исполнитель:

А.М. Попова ИС-20-1

Р.О. Кравчук ИС-20-1

А. Н. Мельник ИС-20-1

Руководитель: преподаватель

Марченко В. Н.

Иркутск – 2023

Оглавление

[Введение 3](#_Toc136859573)

[1. Системный анализ 4](#_Toc136859574)

[1.1. Описание процесса контроля качества хранения продуктов 4](#_Toc136859575)

[1.2. Бизнес-процесс as is. 6](#_Toc136859576)

[1.3. Выявление проблем бизнес-процесса as-is. 7](#_Toc136859577)

[1.4. Бизнес-процесс to-be 8](#_Toc136859578)

[1.5. Компенсация 9](#_Toc136859579)

[1.6. Требования к функциям системы 9](#_Toc136859580)

[**2.** **Разработка проекта системы** 17](#_Toc136859581)

[2.1. Диаграмма прецедентов 17](#_Toc136859582)

[2.2. Схема базы данных 20](#_Toc136859583)

[2.3. Проект интерфейса приложения 21](#_Toc136859584)

[3. Тестирование системы 29](#_Toc136859585)

[4. Заключение 31](#_Toc136859586)

[5. Список использованной литературы 32](#_Toc136859587)

[Описание базы данных. Приложение 1. 33](#_Toc136859588)

[Листинг кода. Приложение 2. 45](#_Toc136859589)

# Введение

Весь мир неуклонно развивается, в том числе, многие бизнес-процессы проходят процесс автоматизации для большей эффективности. Это способствует большей точности и уменьшению затрат на время их реализации.

В наше время проблема выброса пищевых отходов стоит особо остро и it-технологии могут помочь с этой проблемой с помощью отслеживания сроков годности и подбора возможности приготовления блюд с продуктами чей срок годности подходит к концу.

Целью данной курсовой работы является разработка web-приложения для отслеживания сроков годности с такими дополнительными функциями как: советы по хранению продуктов и использование продуктов с истекающим сроком годности для приготовления пищи.

Предметом автоматизации является бизнес-процесс отслеживания срока годности.

Задачи курсовой работы включают:

* анализ предметной области;
* построение модели бизнес-процесса «как есть» (as-is);
* выявление недостатков бизнес-процесса;
* построить модель «должно быть» (to-be);
* проектирование информационной системы;
* создание web-приложения;
* тестирование.

# Системный анализ

## Описание процесса контроля качества хранения продуктов

Отслеживание срока годности является одним из важнейших процессов в борьбе с большими объемами выбросов пищевых отходов.

Потребление свежих и качественных продуктов – важный залог здорового питания. Дата изготовления, срок годности – выясняем, что еще важно для того, чтобы не купить в магазине негодный товар, и чем отличаются срок хранения и срок реализации.

По статистике девять из десяти покупателей при выборе продуктов в первую очередь ориентируются на дату изготовления и срок годности. Производитель обычно указывает две даты: изготовления продукта и его упаковки. На напитках и уксусе вместо даты изготовления и упаковывания используют дату розлива, а на лотках с яйцами – дату сортировки. Также только дату изготовления указывают в том случае, если производство продукта не останавливается после расфасовки или бутилирования. Речь идет, например, о процессах стерилизации, охлаждении и созревания продукта.

Даты обозначают число, месяц и год, а для продуктов, срок годности которых исчисляется часами, дополнительно указывается точное время изготовления. Часто даты наносят на упаковку продукта, а на этикетке размещается только ссылка на неё: «Дата изготовления: см. на упаковке».

Если со сроком годности, который исчисляется со времени изготовления продукта и устанавливается с указанием условий его хранения, все более или менее ясно — это период времени, в течение которого продукты остаются безопасными для употребления, то как быть со сроками хранения и реализации, которыми также нередко маркируют товары?

Срок хранения – это период, в течение которого пищевой продукт при соблюдении установленных условий хранения сохраняет свойства, указанные в нормативном или техническом документе. Истечение срока хранения не означает, что продукт не пригоден для использования по назначению.

Есть разные категории товаров, которые требуют повышенного внимания на всех этапах хранения и реализации: косметика, бытовая химия, строительные материалы и даже автомобильные шины. Но самая обширная группа — это продукты питания и лекарства с небольшим сроком годности.

Чаще всего просроченная продукция появляется из-за двух причин:

* неоптимизированные закупки — когда покупают больше товаров, чем способны реализовать;
* неоптимизированная реализация — когда новокупленные продукты используются раньше, чем купленные раньше.

На складах эти проблемы решаются с помощью адресного хранения: в ячейке находятся товары с одним сроком годности и используется метод FIFO (First In First Out, первый поступил — первый выбыл). Таким образом снижается вероятность возникновения ошибок при отгрузке, так как учетная система автоматически выбирает ячейку с продукцией из самой ранней партии.

Контроль сроков годности должен вестись с момента получения товара. Как правило, производители указывают на коробах и палетах линейные штрихкоды GS1-128. При приемке с помощью терминала сбора данных код сканируется и учитывается партия, например, по дате поступления или сроку годности. В учетной программе появляется информация обо всех принятых товарах и их сроках хранения.

В розничных магазинах ситуация сложнее: сроки годности товаров, находящихся на полках, сейчас контролируются только вручную. Автоматизированного решения пока нет, поскольку на упаковках штучной продукции код GS1-128 не размещается, это сложно технологически.

Еще сложнее ситуация складывается у физических клиентов.

## Бизнес-процесс as is.

Процесс отслеживания срока годности является одним из важных этапов в борьбе за уменьшение количества выкидываемых продуктов. Автоматизация такого процесса может значительно упростить его, сделать быстрее и надежнее.

Процесс отслеживания срока годности проходит с помощью информации о сроке годности продукта, информации о его количестве, знаниях о правилах хранения. Человек должен либо запоминать срок годности, либо ввести специальные записи с этой информацией

Входы данного бизнес-процесса:

* информация о продукте;
* информация о количестве продукта.

Управление:

* правила хранения продуктов.

Механизмы:

* человек.

Выход:

* список продуктов с истекшим сроком годности.

Схема бизнес-процесса 1 уровня «Отслеживания срока годности» показана на рисунке ниже (Рисунок 1)

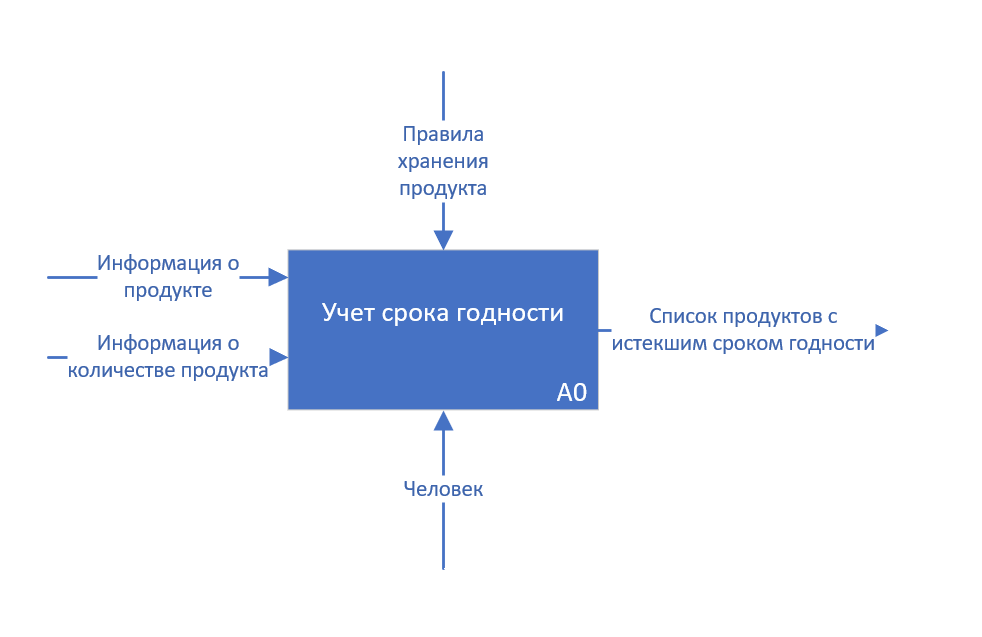


Рисунок 1.Процесса 1 уровня «Отслеживания срока годности»

Процесс декомпозируется на подпроцесс 2 уровня «Отслеживания срока годности». Модель 2 уровня показана на рисунке ниже ( Рисунок 2).

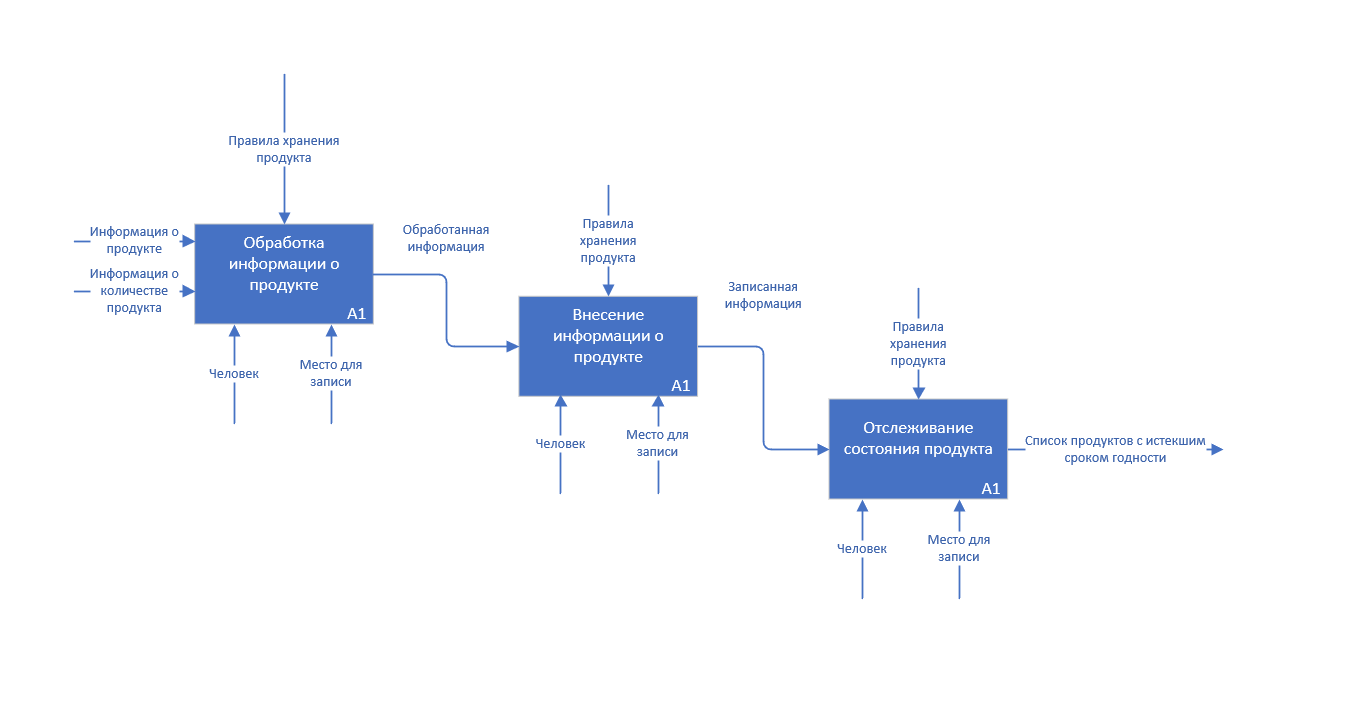


Рисунок 2.Процесс 2 уровня «Отслеживание срока годности продукта»

**Блок «Обработка информации о продукте»**. Входными данными является информация о продукте и его количестве. Управлением будет являться правила хранения продуктов. Механизмами являются: человек и место для записи. Технологией выполнения: человек после покупки продукта записывает его срок годности в место для записи. Выходом является обработанная информация.

**Блок «Внесение информации о продукте».** На входе процесса будет обработанная человеком информация. Управление – правила хранения продукта. Технология выполнения: человек вносит информацию в место записи вручную. Выходом процесса будет записанная информация.

**Блок «Отслеживание состояния продукта».** Входными данными будет информация о записанном перечне продуктов. Управление – правила хранения продукта. Технология выполнения: человек отслеживает свежесть своих продуктов. Выходом процесса будет информация о списке продуктов с истекшим сроком годности.

## Выявление проблем бизнес-процесса as-is.

В бизнес-процессе «Отслеживания срока годности» «как есть» можно выделить ряд проблем:

* запись происходит вручную. Человеку это может быть неудобно;
* все записи на прием ведутся на каком-то носителе. Разрозненность информации может грозить её потерей или нарушениями в качестве данных;
* процесс занимает большое количество времени;
* человек не может самостоятельно отследить все правила хранения продуктов в домашних условиях.

Для решения данных проблем необходимо автоматизировать процесс с помощью сайта, на котором человек может систематизировано хранить информацию о своих продуктах и местах, где они хранятся. Такой способ более удобный для людей.

## Бизнес-процесс to-be

На модели бизнес-процесса «Учет срока годности» to-be, в отличие от as-is изменилась технология выполнения: в модели to-be теперь участвует веб-приложение, так же управляют процессом теперь правила ГОСТ.

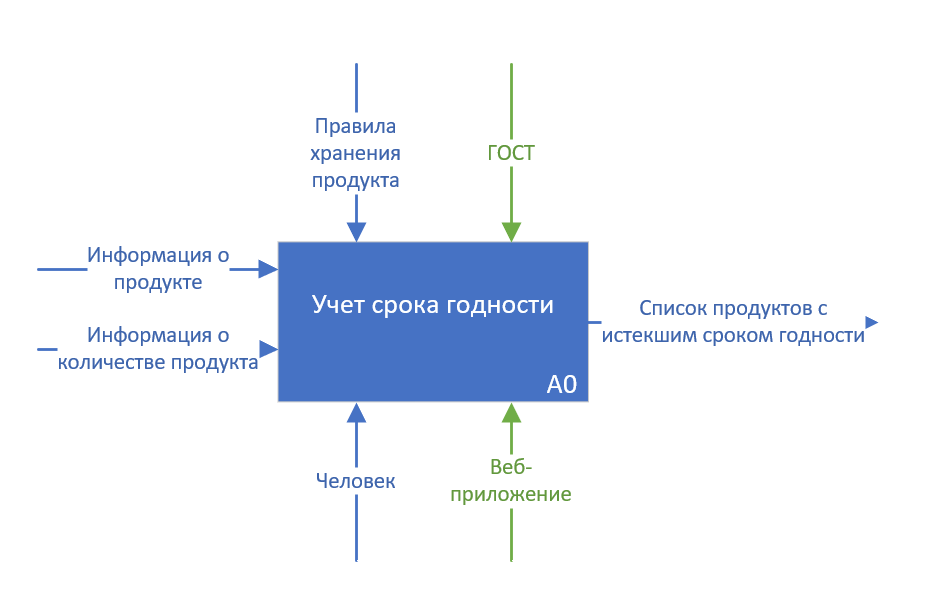


Рисунок 3. Срок годности to be

При декомпозиции процесса более явно становится видно высвобождение полезного времени человека:

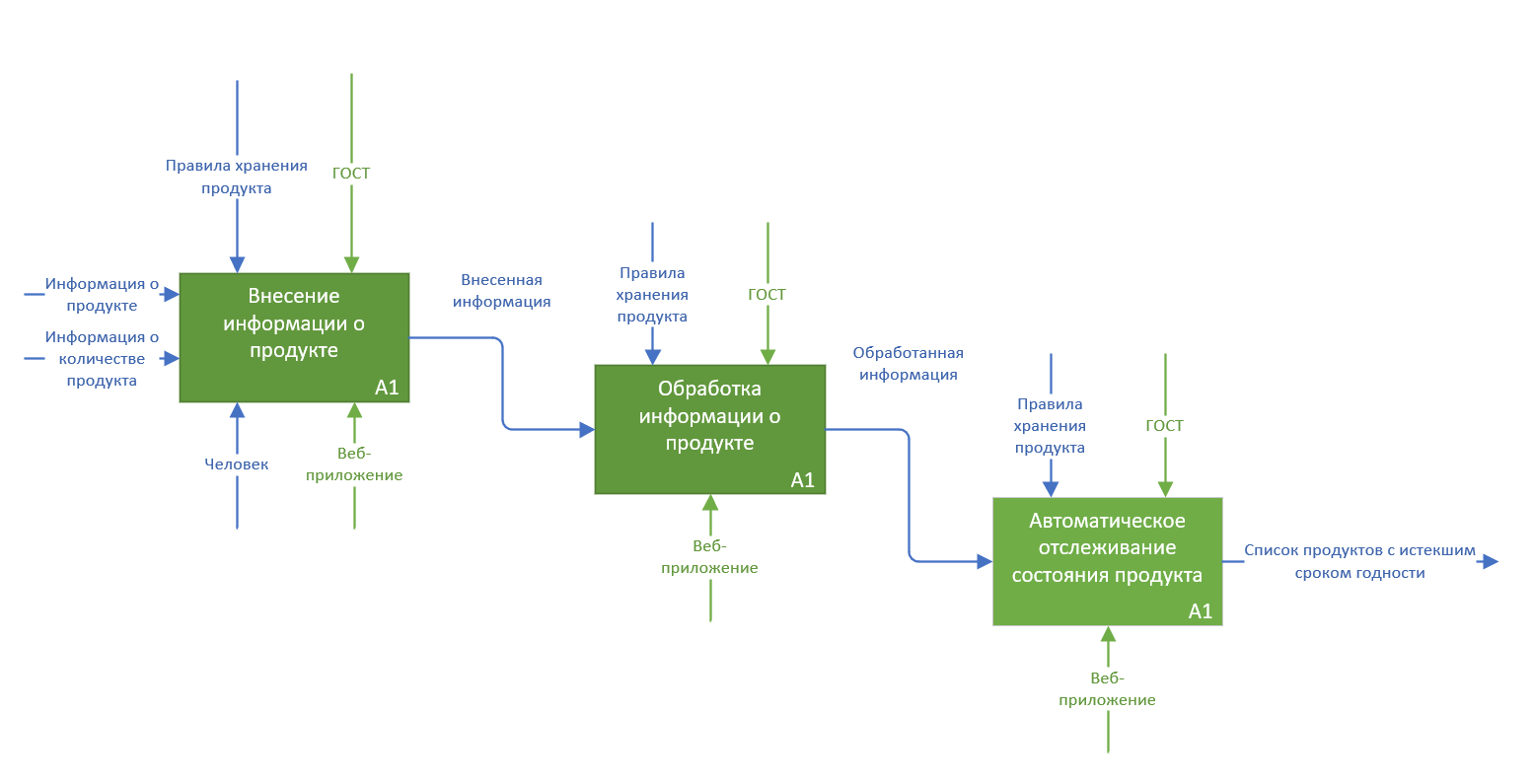


Рисунок 4. Декомпозированный процесс to be

## Компенсация

Когда человек заполнил форму создания продукта и внесения информации о сроке годности данного продукта, необходимо сделать дополнительные проверки.

1. Правильно ли заполнены все поля.
2. Нет ли нарушения логики заполнения полей. Например, дата окончания срока годности не должна быть меньше даты начала срока годности.

Если при проверке всех условий появляется ошибка, тогда запись на прием не добавляется в базу данных, а клиенту приходит сообщение об ошибке и просьба исправить некорректные данные, внесенные на форму.

## Требования к функциям системы

Требования к функциям системы представлены в виде спецификации API. А также списком страниц, которые должны быть на сайте.

|  |  |  |
| --- | --- | --- |
| Authentication |  |  |
| POST | auth/login | Login обрабатывает запрос пользователя на вход и возвращает токены доступа и обновления. |
| POST | auth/logout | Выйти из системы |
| POST | auth/refresh | Обновить токен доступа |
| POST | auth/sign-up | Зарегистрируйте нового пользователя |
| Measures |  |  |
| GET | measures | Найти меры |
| POST | measures | Создать новую запись меры |
| GET | measures/{measure\_id} | Найти меру по идентификатору |
| PUT | measures/{measure\_id} | Обновить меру |
| DELETE | measures/{measure\_id} | Удалить меру |
| Product Categories |  |  |
| GET | product-categories | Найти категории продуктов |
| POST | product-categories | Создать категорию продукта |
| GET | product-categories/{category\_id} | Найти категорию продукта по идентификатору |
| PUT | product-categories/{category\_id} | Обновление существующей категории продуктов по идентификатору |
| DELETE | product-categories/{category\_id} | Удалить категорию продукта |
| PATCH | product-categories/{category\_id} | Восстановить удаленную категорию товаров |
| Products |  |  |
| GET | products | Получить список продуктов |
| POST | /products | Создать новый продукт |
| GET | products/{product\_id} | Получить товар по ID |
| PUT | /products/{product\_id} | Обновить продукт |
| DELETE | /products/{product\_id} | Удалить продукт |
| PATCH | /products/{product\_id}/ | Восстановить удаленный продукт |
| GET | /products/{product\_id}/categories | Получить категории продукта |
| POST | /products/{product\_id}/categories/{category\_id} | Добавить категорию к продукту |
| DELETE | /products/{product\_id}/categories/{category\_id} | Удалить категорию из продукта |
| GET | /products/{product\_id}/recipes | Получить рецепты продукта |
| POST | /products/{product\_id}/tip/{tip\_id} | Добавить совет для продукта |
| DELETE | /products/{product\_id}/tip/{tip\_id} | Удалить чаевые из продукта |
| GET | /products/{product\_id}/tips | Найдите советы по продукту |
| Recipes |  |  |
| GET | /recipes | Найти рецепты |
| POST | /recipes | Создать новый рецепт |
| GET | /recipes/{recipe\_id} | Найти рецепт по ID |
| PUT | /recipes/{recipe\_id} | Обновить рецепт |
| DELETE | /recipes/{recipe\_id} | Удалить рецепт |
| PATCH | /recipes/{recipe\_id} | Восстановить удаленный рецепт по ID |
| GET | /recipes/{recipe\_id}/ingredients | Добавить ингредиент в рецепт |
| PUT | /recipes/{recipe\_id}/ingredients | Обновление ингредиента рецепта |
| DELETE | /recipes/{recipe\_id}/ingredients | Удаление ингредиента из рецепта |
| GET | /recipes/{recipe\_id}/steps | Найти все шаги рецепта |
| POST | /recipes/{recipe\_id}/steps/{step\_id} | Создать шаг рецепта |
| DELETE | /recipes/{recipe\_id}/steps/{step\_id} | Удаление шага рецепта. |
| Roles |  |  |
| GET | /roles | Найдите много ролей |
| POST | /roles | Создать роль |
| GET | /roles/{role\_id} | Найдите одну роль |
| PUT | /roles/{role\_id} | Обновить роль |
| DELETE | /roles/{role\_id} | Удалить роль |
| PATCH | /roles/{role\_id} | Восстановить роль |
| Settings |  |  |
| GET | /settings | Найти настройки |
| POST | /settings | Создать настройку |
| GET | /settings/{id\_setting} | Найти настройку |
| PUT | /settings/{id\_setting} | Обновить настройку |
| PATCH | /settings/{id\_setting} | Восстановить настройки |
| Shelf Life Detector |  |  |
| POST | /shelf-life-detector | Определить срок годности из файла |
| Shelf Life Status |  |  |
| GET | /shelf-life-statuses | Найдите много статусов срока годности |
| POST | /shelf-life-statuses | Создать статус срока годности |
| GET | /shelf-life-statuses/{id\_status} | Найдите один статус срока годности |
| PUT | /shelf-life-statuses/{id\_status} | Обновить статус срока годности |
| DELETE | /shelf-life-statuses/{id\_status} | Удалить статус срока годности |
| Shelf Lives |  |  |
| GET | /shelf-lives | Найти много сроков годности |
| POST | /shelf-lives | Создайте срок годности |
| GET | /shelf-lives/{shelf\_life\_id} | Узнать срок годности по id |
| PUT | /shelf-lives/{shelf\_life\_id} | Обновить срок годности |
| DELETE | /shelf-lives/{shelf\_life\_id} | Удалить срок годности |
| PATCH | /shelf-lives/{shelf\_life\_id} | Восстановить срок годности |
| GET | /shelf-lives/{shelf\_life\_id}/statuses | Найти статусы срока годности |
| POST | /shelf-lives/{shelf\_life\_id}/statuses/{status\_id} | Добавить статус срока годности |
| DELETE | /shelf-lives/{shelf\_life\_id}/statuses/{status\_id} | Удалить статус срока годности |
| Steps |  |  |
| GET | /steps | Найти много шагов |
| POST | /steps | Создать шаг |
| GET | /steps/{id\_step} | Найдите один шаг |
| PUT | /steps/{id\_step} | Обновить шаг |
| DELETE | /steps/{id\_step} | Удалить шаг |
| PATCH | /steps/{id\_step} | Восстановить шаг |
| Storage Types |  |  |
| GET | /storage-types | Найти типы хранилищ по фильтру |
| POST | /storage-types | Создать тип хранилища |
| GET | /storage-types/{id\_type} | Найти тип хранилища по id |
| PUT | /storage-types/{id\_type} | Обновить тип хранилища |
| DELETE | /storage-types/{id\_type} | Удалить тип хранилища |
| GET | /storage-types/{id\_type}/storages | Поиск хранилищ по идентификатору типа хранилища |
| GET | /storage-types/{id\_type}/tips | Поиск подсказок по идентификатору типа хранилища |
| POST | /storage-types/{id\_type}/tips/{id\_tip} | Добавить подсказку к типу хранилища |
| DELETE | /storage-types/{id\_type}/tips/{id\_tip} | Удалить подсказку из типа хранения |
| Storages |  |  |
| GET | /storages | Найти хранилища |
| POST | /storages | Создать хранилище |
| GET | /storages/{id\_storage} | Найти хранилище |
| PUT | /storages/{id\_storage} | Обновить хранилище |
| DELETE | /storages/{id\_storage} | Удалить хранилище |
| PATCH | /storages/{id\_storage} | Восстановить хранилище |
| GET | /storages/{id\_storage}/shelf-lives | Найдите срок годности |
| GET | /storages/{id\_storage}/tips | Найдите советы по хранению |
| POST | /storages/{id\_storage}/tips/{id\_tip} | Добавить чаевые в хранилище |
| DELETE | /storages/{id\_storage}/tips/{id\_tip} | Удалить наконечник из хранилища |
| Tips |  |  |
| GET | /tips | Найдите много советов |
| POST | /tips | Создать совет |
| GET | /tips/{id\_tip} | Найти подсказку по id |
| PUT | /tips/{id\_tip} | Совет по обновлению |
| DELETE | /tips/{id\_tip} | Удалить совет |
| PATCH | /tips/{id\_tip} | Совет по восстановлению |
| GET | /tips/{id\_tip}/products | Найти продукты |
| POST | /tips/{id\_tip}/products/{id\_product} | Добавить продукт |
| DELETE | /tips/{id\_tip}/products/{id\_product} | Удалить продукт |
| GET | /tips/{id\_tip}/storages | Найти хранилища |
| POST | /tips/{id\_tip}/storages/{id\_storage} | Добавить хранилище |
| DELETE | /tips/{id\_tip}/storages/{id\_storage} | Удалить хранилище |
| Users |  |  |
| GET | /users | Найти пользователей |
| PUT | /users | Обновить пользователя |
| POST | /users | Создать пользователя |
| DELETE | /users | Удалить пользователя |
| GET | /users/{id\_user} | Найти пользователя по id |
| GET | /users/{id\_user}/roles | Найти роли пользователей |
| GET | /users/{id\_user}/settings | Найти пользовательские настройки |
| PUT | /users/{id\_user}/settings/{id\_setting} | Обновить настройки пользователя |
| GET | /users/{id\_user}/shelf-lives | Найдите срок годности пользователя |
| POST | /users/{id\_user}/shelf-lives | Создание пользовательского срока годности |
| PUT | /users/{id\_user}/shelf-lives/{id\_shelf\_life} | Обновить срок годности пользователя |
| DELETE | /users/{id\_user}/shelf-lives/{id\_shelf\_life} | Удалить срок годности пользователя |
| PATCH | /users/{id\_user}/shelf-lives/{id\_shelf\_life} | Восстановить срок годности пользователя |
| GET | /users/{id\_user}/storages | Найти пользовательские хранилища |
| POST | /users/{id\_user}/storages/{id\_storage} | Добавить пользовательское хранилище |
| DELETE | /users/{id\_user}/storages/{id\_storage} | Удалить пользовательское хранилище |

# **Разработка проекта системы**

## Диаграмма прецедентов

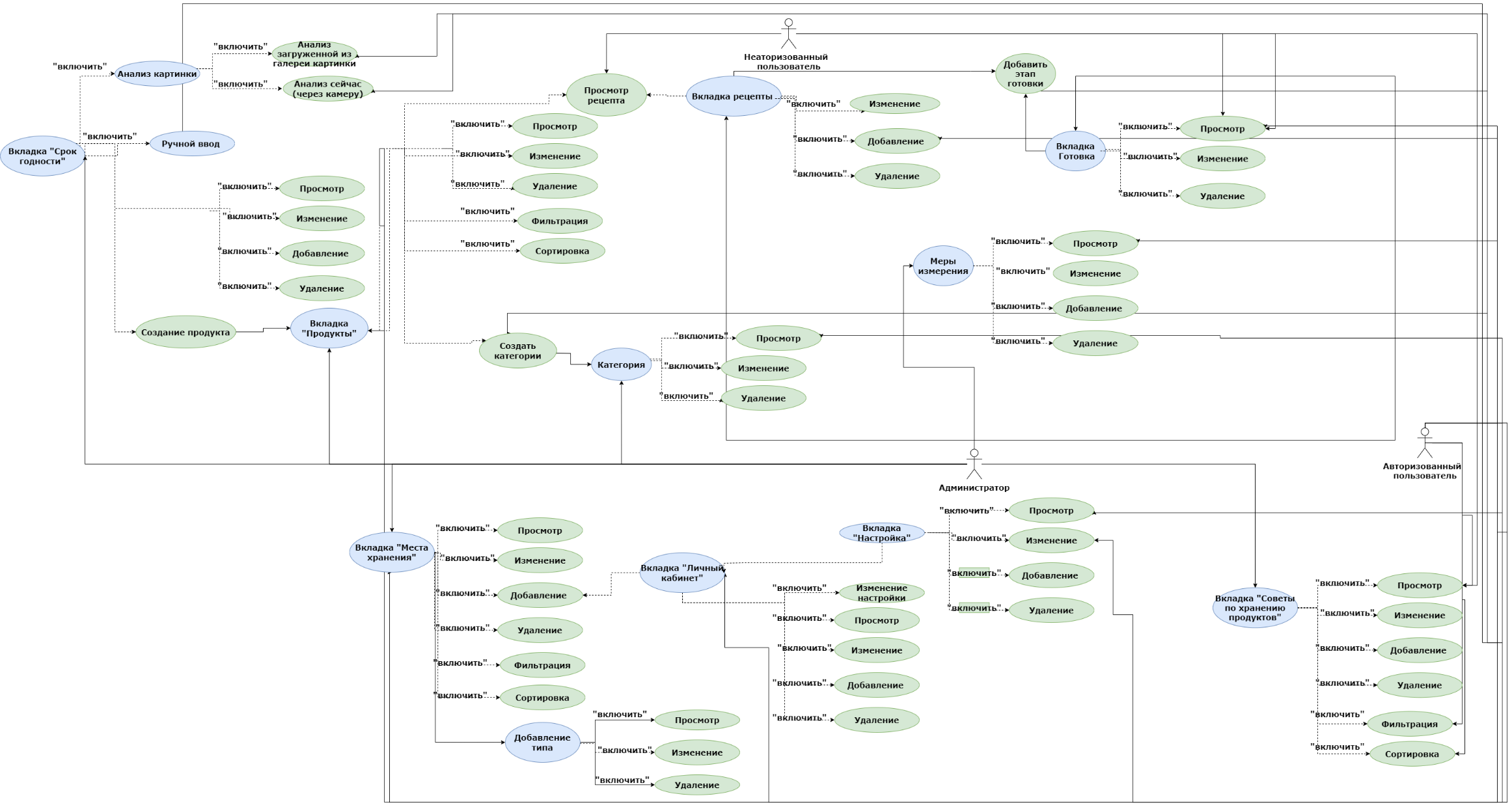
Use-case или же диаграмма прецедентов в данном веб-приложении показывает наличие следующих подсистем, с которыми могут взаимодействовать пользователи:

* рецепты;
* продукты;
* личный кабинет;
* места хранения;
* советы по хранению продуктов.

В системе предполагается наличие трех типов акторов:

* администратор;
* авторизованный пользователь;
* неавторизованный пользователь.

Соответственно ролям акторы получают определенные возможности взаимодействия с системой.



Далее приводится подробное описание основных прецедентов системы:

Таблица 1. Описание прецедента "Создание продукта"

|  |  |
| --- | --- |
| Имя прецедента | Создание продукта |
| Краткое описание | Прецедент позволяет создавать продукт в системе |
| Акторы | Администратор, Авторизованный пользователь |
| Предусловие | ‒ |
| Основной поток | Выводится форма для заполнения данных о продукте |
| Постусловия | Если прецедент был успешно завершен, то пользователь может выполнять другие действия. |
| Альтернативные потоки | Если страница не загружается, то выводится страница «404» |

Таблица 2. Описание прецедента "Просмотр страницы Рецепты"

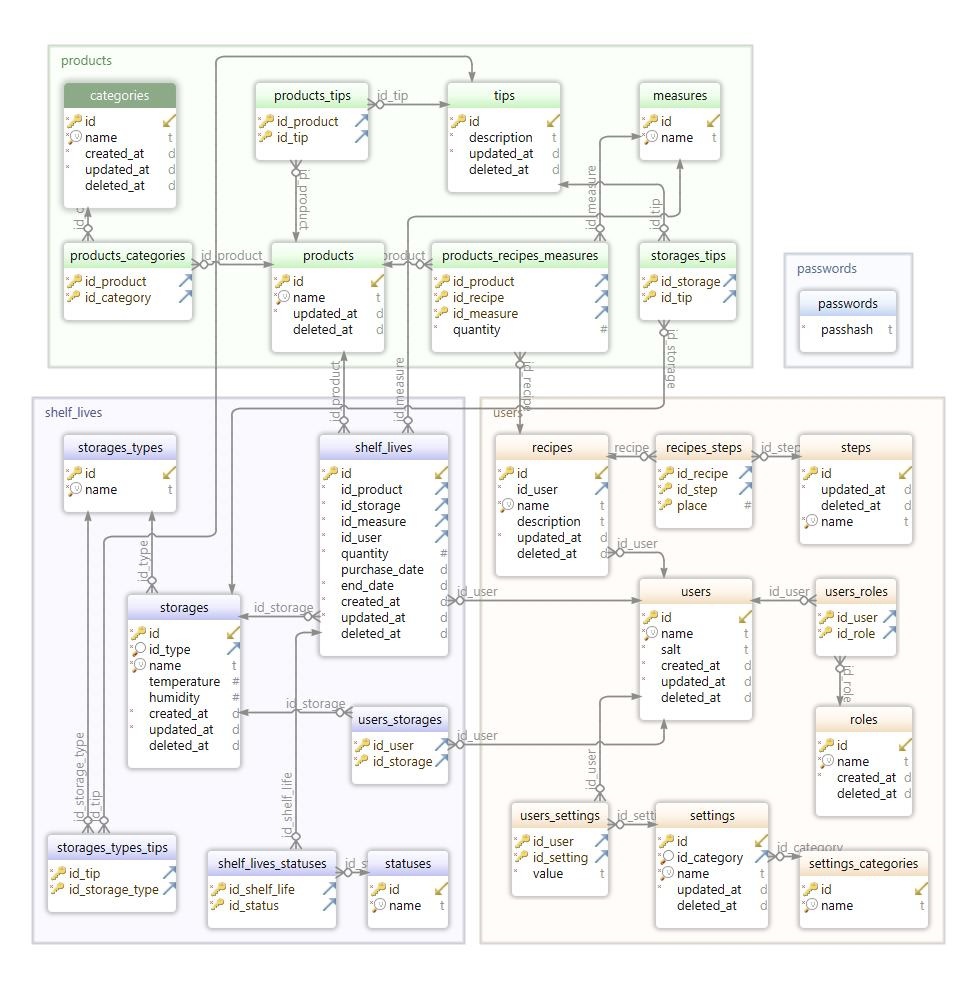
|  |  |
| --- | --- |
| Имя прецедента | Просмотр страницы «Рецепты» |
| Краткое описание | Прецедент позволяет просматривать страницу «Рецепты» |
| Акторы | Администратор, Авторизованный пользователь, Неавторизованный пользователь |
| Предусловие | ‒ |
| Основной поток | Выводится вся информация, которая должна быть в разделе «Рецепты» |
| Постусловия | Если прецедент был успешно завершен, то пользователь может выполнять другие действия. |
| Альтернативные потоки | Если страница не загружается, то выводится страница «404» |

Таблица 3. Описание прецедента "Просмотр страницы Советы"

|  |  |
| --- | --- |
| Имя прецедента | Просмотр страницы «Советы» |
| Краткое описание | Прецедент позволяет просматривать страницу «Советы» |
| Акторы | Администратор, Авторизованный пользователь, Неавторизованный пользователь |
| Предусловие | ‒ |
| Основной поток | Выводится вся информация, которая должна быть в разделе «Советы» |
| Постусловия | Если прецедент был успешно завершен, то пользователь может выполнять другие действия. |
| Альтернативные потоки | Если страница не загружается, то выводится страница «404» |

## Схема базы данных

В проекте предполагается использование реляционной SQL базы данных. Для неё была разработана схема базы данных. С подробным описанием можно ознакомиться в Приложении 1.



## Проект интерфейса приложения

Первым делом для проекта была выбрана цветовая палитра:

1. Нежно розовый
2. Бледно желтый
3. Пастельно зеленый
4. Темно зеленый

Соответственно, следующие цвета:

* #FFD4D4, rgb(255, 212, 212);
* #FFFFE8, rgb(255, 255, 232);
* #CDE990, rgb(205, 233, 144);
* #AACB73, rgb(170, 203, 115).

Следующим этапом непосредственно был разработан дизайн интерфейса:

Главная страница, на которую попадает пользователь

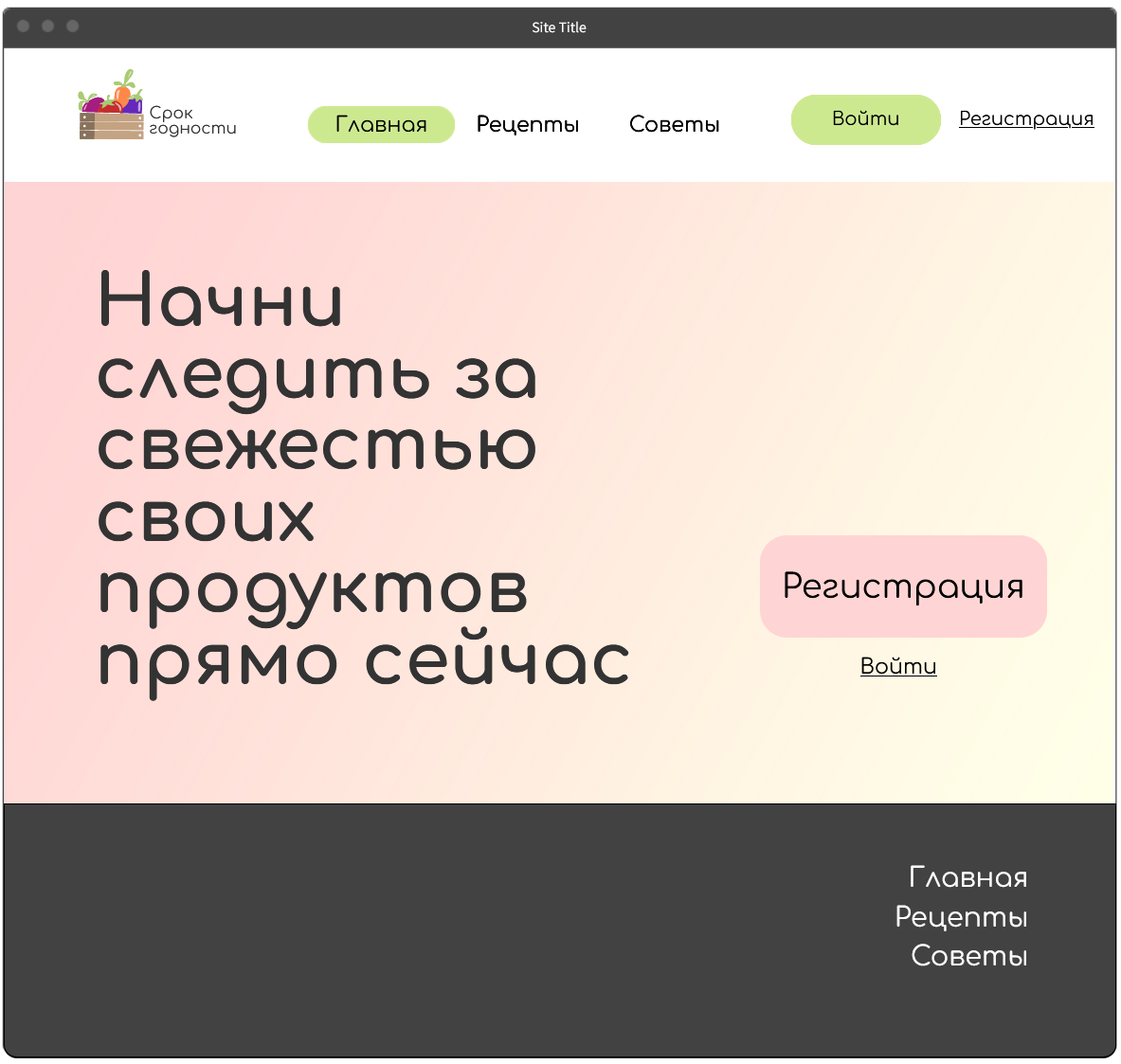


Рисунок 6. Главная страница

Страница с советами:

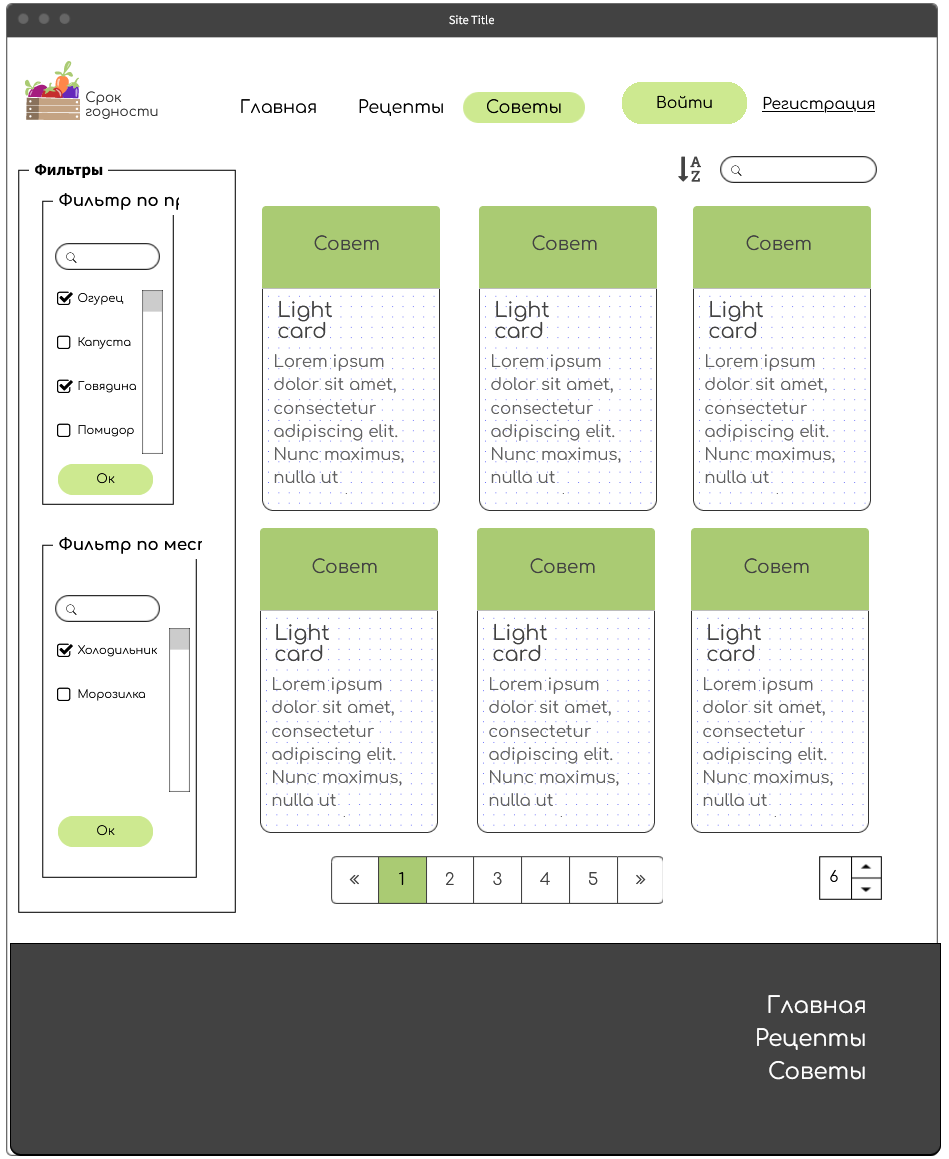


Рисунок 7. Страница "Советы"

Страница «Рецепты»:

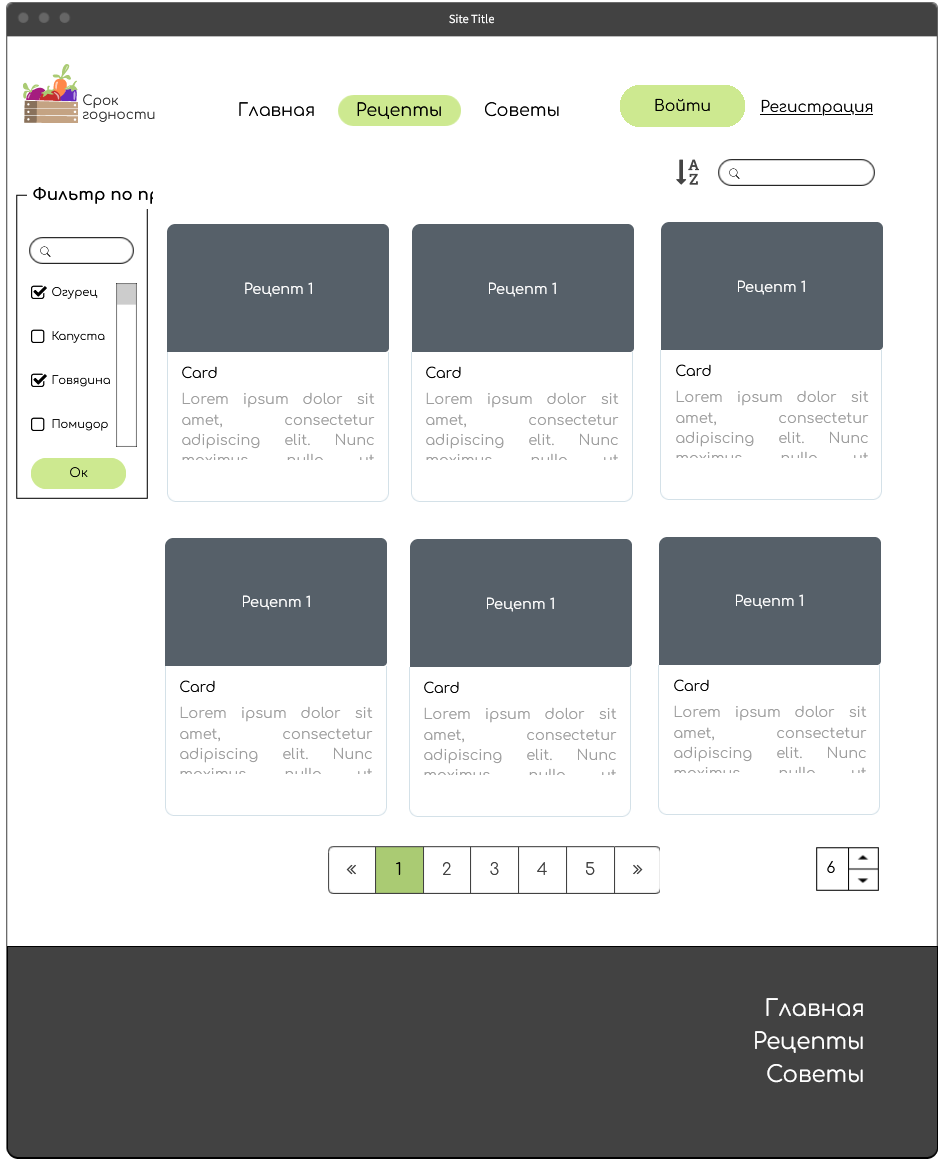


Рисунок 8. Страница "Рецепты"

Страница регистрации:

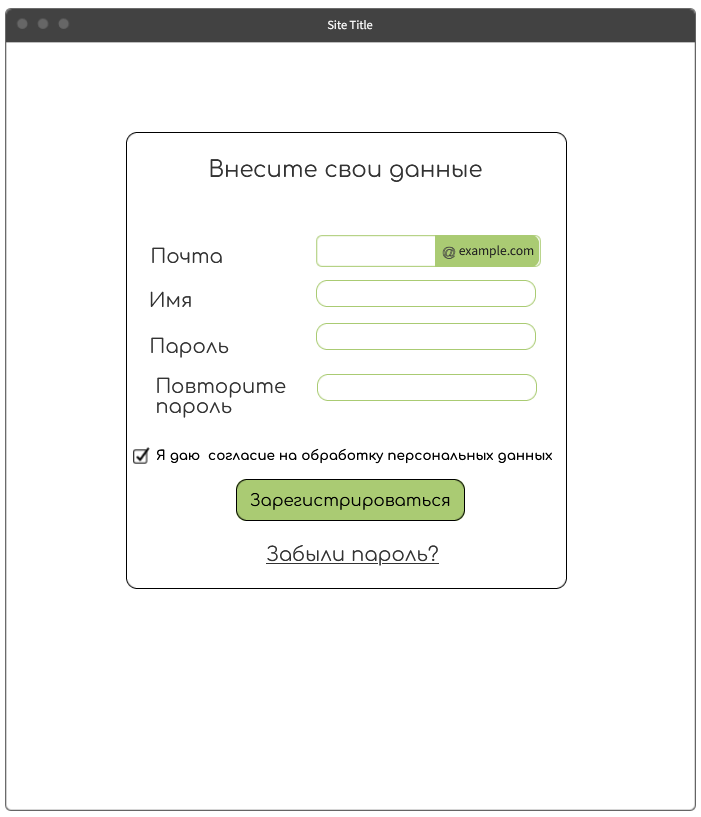


Рисунок 9. Окно регистрации

Страница входа:

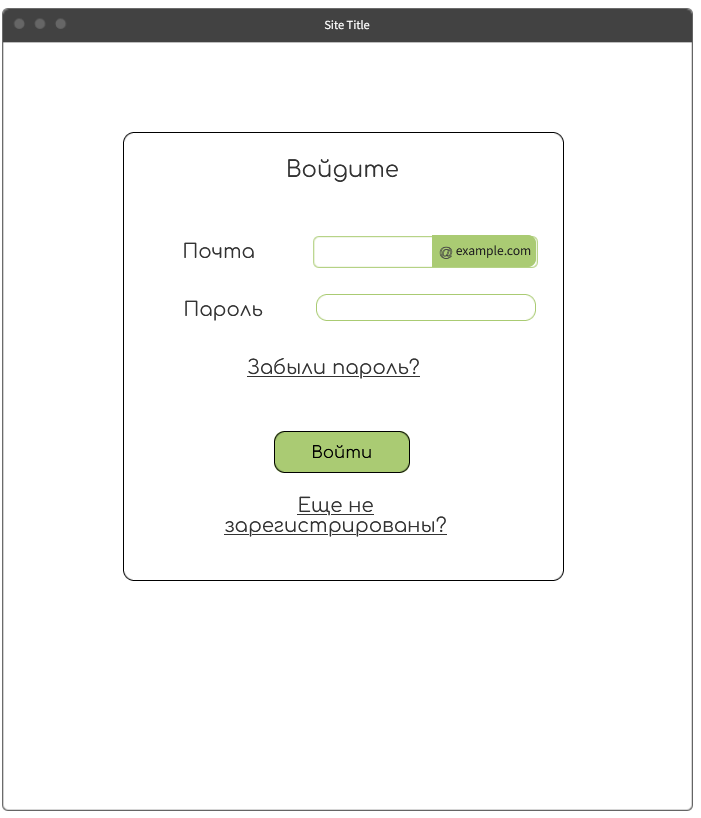


Рисунок 10. Окно входа

Окно «Забыли пароль?»:

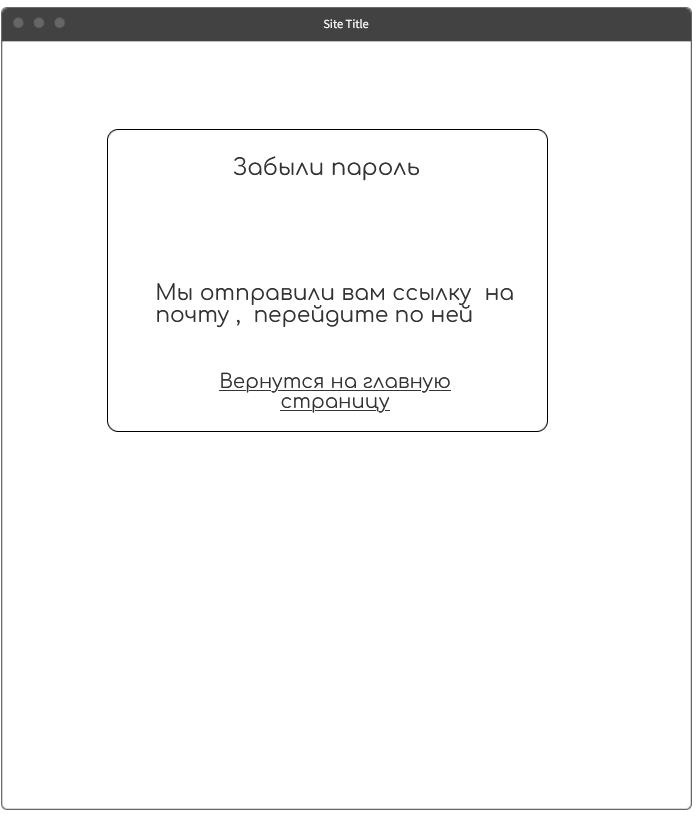


Рисунок 11. Окно "Забыли пароль?"

Добавление категории:

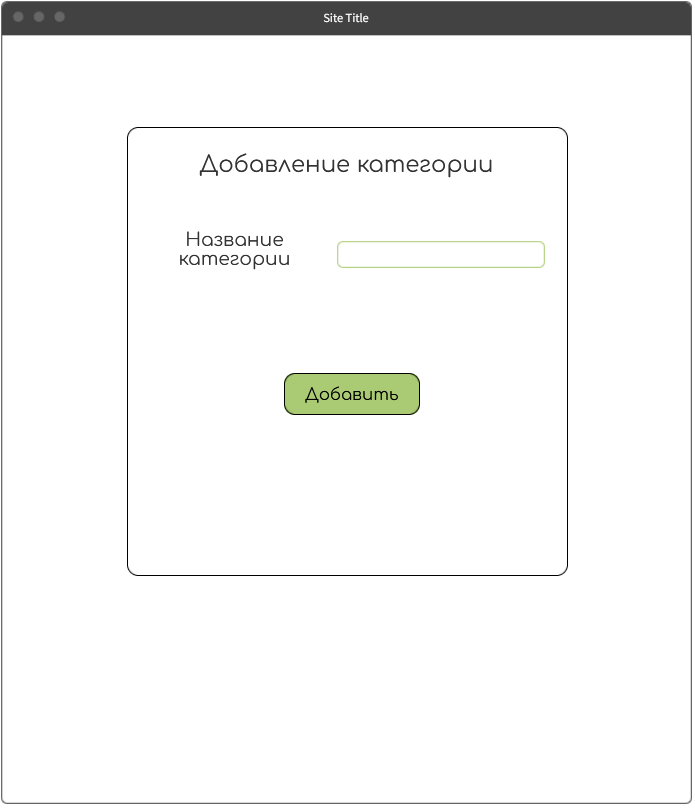


Рисунок 12. Окно "Добавление категории"

Добавление места хранения:

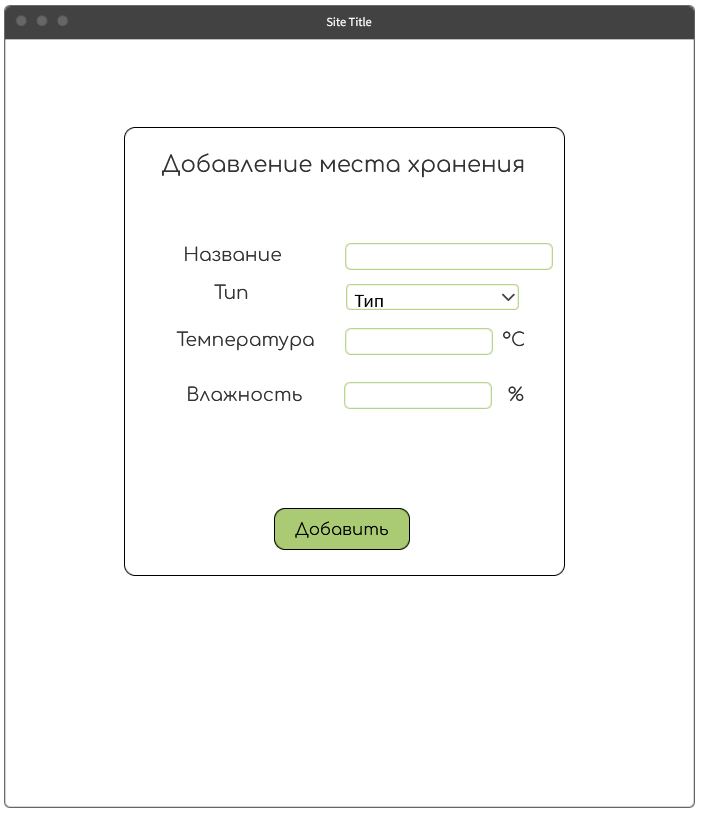


Рисунок 13. Окно "Добавление места хранения"

Изменение пароля:

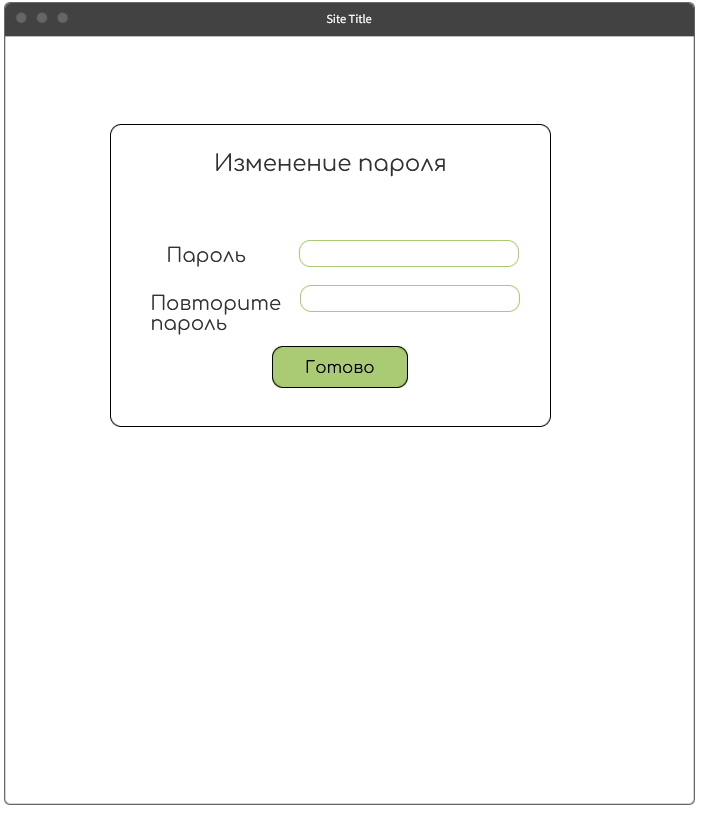


Рисунок 14. Окно "Изменение пароля"

Добавление продукта:

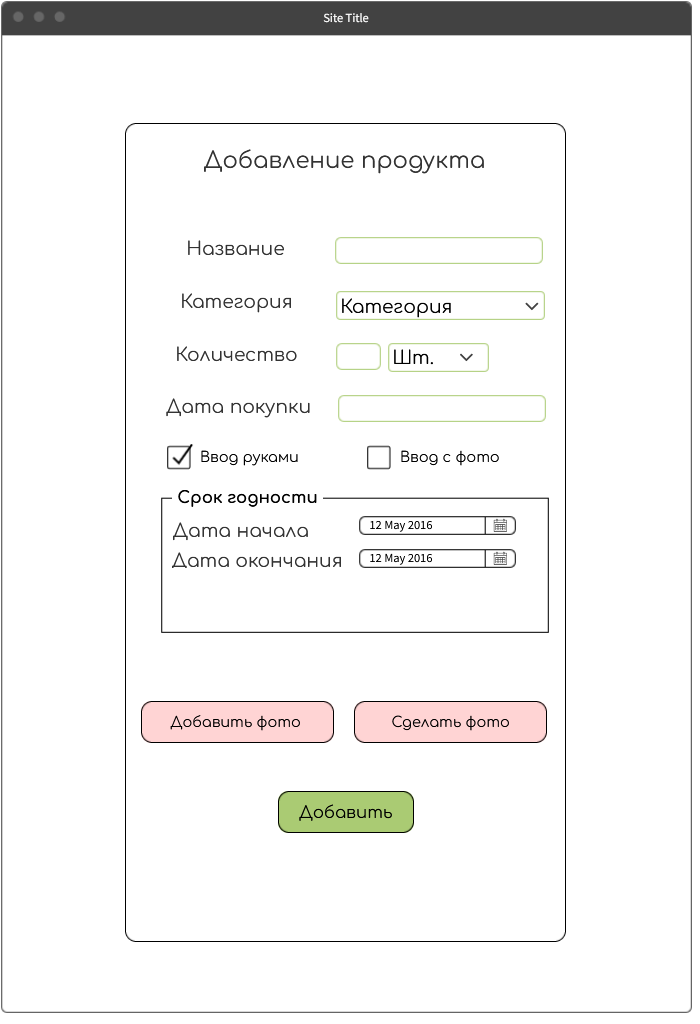


Рисунок 15. Окно "Добавление продукта"

Добавление рецепта:

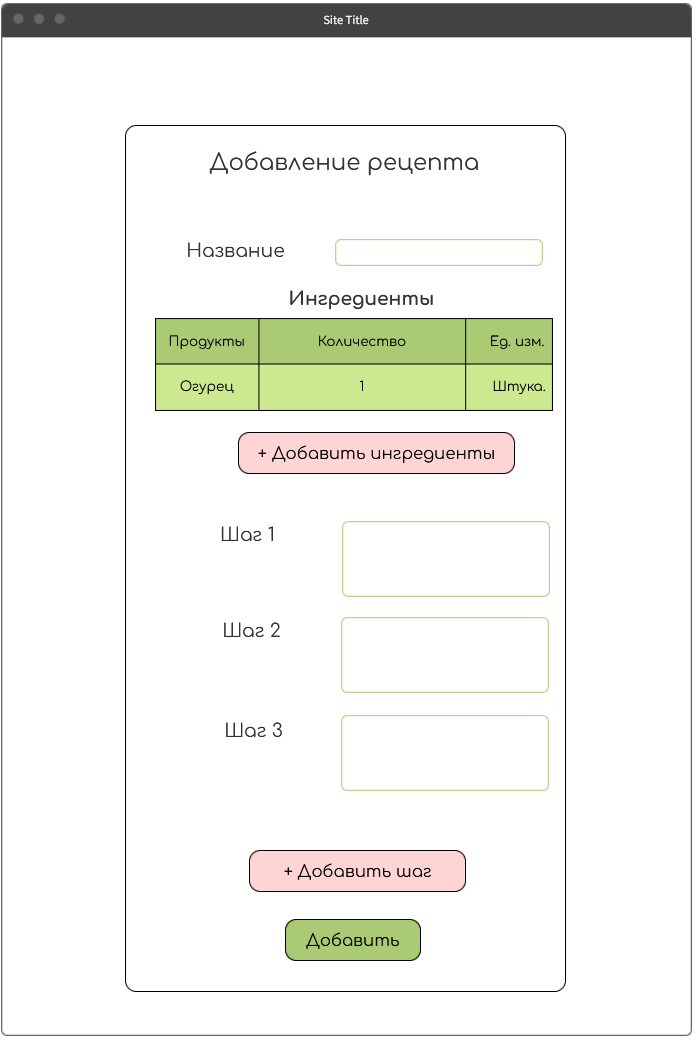


Рисунок 16. Окно "Добавление рецепта»

# Тестирование системы

Таблица 4. Таблица обработки ошибок форм

|  |  |  |  |
| --- | --- | --- | --- |
| **Название формы** | **Название поля ввода** | **Ошибочные типы** | **Обработка ошибки** |
| Регистрация | Почта | Пустое поле | Подсвечивается красным цветом с надписью «Поле не должно быть пустым» |
| Имя | Поле пустое | Подсвечивается красным цветом с надписью «Поле не должно быть пустым» |
| Формат не текстовый | Подсвечивается красным цветом с надписью «Неправильный формат» |
| Пароль | Поле пустое | Подсвечивается красным с надписью «Поле не должно быть пустым» |
| Меньше восьми символов | Подсвечивается красным с надписью «Слишком короткий пароль» |
| Отсутствие цифр | Подсвечивается красным с надписью «Слишком простой пароль» |
| Отсутствие заглавных букв | Подсвечивается красным с надписью «Слишком простой пароль» |
| Чекбокс «Я даю согласие на обработку персональных данных» | Не нажата | Кнопка «Зарегистрироваться» неактивна |
| Добавление места хранения | Название | Пустое поле | Подсвечивается красным с надписью «Поле не должно быть пустым» |
| Температура | Поле пустое | Подсвечивается красным с надписью «Поле не должно быть пустым» |
| Формат не цифры | Подсвечивается красным с надписью «Неправильный формат» |
| Влажность | Поле пустое | Подсвечивается красным с надписью «Поле не должно быть пустым» |
| Формат не цифры | Подсвечивается красным с надписью «Неправильный формат» |
| Добавление продукта | Количество | Пустое поле | Подсвечивается красным с надписью «Поле не должно быть пустым» |
| Формат не цифры | Подсвечивается красным с надписью «Неправильный формат» |
| Дата покупки | Поле пустое | Подсвечивается красным с надписью «Поле не должно быть пустым» |
| Формат не дата | Подсвечивается красным с надписью «Неправильный формат» |

# Заключение

В процессе написания курсовой работы был изучен процесс контроля срока годности продуктов как в секторе бизнеса, так и в секторе личного пользования. На основе полученных данных был проведён системный анализ отрасли и, непосредственно, анализ процесса контроля срока годности.

Были выявлены главные проблемы и пробелы бизнес-процесса, а также предложен путь их решения и автоматизации.

Представленным решением выявленных проблем было создание веб-приложения. Для выполнения данной задачи было составлено техническое задание, схема базы данных, основные требования, диаграммы use-case.

Следующим этапом была выбрана цветовая схема и макет интерфейса веб-приложения.

После этого техническое задание было передано на реализацию команде разработчиков.

Результатом работы стало веб-приложение для контроля срока годности с использованием Computer vision с возможностью хранения, изменения, удаления и просмотра необходимой информации, что полностью соответствует предъявленным в курсовой требованиям.

# Список использованной литературы

1. Евсеев Д. А. Дмитрий Андреевич, Трофимов В. В. Валерий Владимирович, Трофимов В. В. Web-дизайн в примерах и задачах: Текст/ Д. А. Евсеев, В. В. Трофимов.- М.: КноРус, 2009.-263 с.
2. Введение в UML [электронный ресурс] /режим доступа: <http://www.intuit.ru/studies/courses/1007/229/info>
3. ГОСТ 34.602-89. Техническое задание на создание автоматизированной системы.  — Москва: Изд-во стандартов, 1989. — 14с.
4. Дакетт Джон HTML и CSS. Разработка и дизайн веб-сайтов; Эксмо - Москва, 2013. - 480 c.
5. Дронов Владимир HTML 5, CSS 3 и Web 2.0. Разработка современных Web-сайтов; БХВ-Петербург - Москва, 2011. - 416 c.
6. Китинг, Джоди Flash MX. Искусство создания web-сайтов; ТИД ДС - Москва, 2012. - 848 c.
7. Костин С. П. Самоучитель создания Web-сайтов; Триумф - Москва, 2009. - 176 c.

# Описание базы данных. Приложение 1.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Описание таблицы** | **Поля** | **Ключ** | **Данные (MySQL)** | **Описание поля** | **Обязательное поле** | **Автоматическое значение** | **Ограничения** | **Индекс** |
| **Table “Fresh”** | | | | | | | | |
| Таблица содержит основную информацию о сроке годности | id\_fresh | PK | INT | Уникальный идентификационный номер срока годности | + | AUTO INCREMENT |  | PRIMARY |
| id\_product | FK | INT | Ключ к таблице Product | + |  |  | product.id\_product |
| id\_storage | FK | INT | Ключ к таблице Storage | - |  |  | storage.id\_storage |
| id\_measure | FK | INT | Ключ к таблице Measure | + | Автоматически проставляет тот, что означает штуку |  | measure.id\_measure |
| quantity |  | REAL | Количество продуктов | - | Автоматически простовляеться 1 | не может быть меньше 0 |  |
| date\_purchase |  | DATETIME | Дата покупки | - |  |  |  |
| date\_add |  | DATETIME | Дата внесения | + | DATETIME.NOW |  |  |
| date\_end |  | DATETIME | Дата окончания срока годности | + | Автоматически рассчитывается |  |  |
| row\_deleted\_time |  | DATETIME | Дата удаления строки | + | 31.12.2999 |  |  |
| row\_change\_time |  | DATETIME | Дата изменения строки | + | 31.12.2999 |  |  |
| **Table “Fresh\_Status”** | | | | | | | | |
| Таблица содержит связь Сроков годности и статуса | id\_fresh | FK | INT | Ключ к таблице Fresh | + | Последний добавленный id\_fresh |  | fresh.id\_fresh |
| id\_status | FK | INT | Ключ к таблице Status | + | Автоматически проставляет тот, что означает внесен |  | status.id\_status |
| **Table “Status”** | | | | | | | | |
| Таблица содержит справочник со статусами | id\_status | PK | INT | Уникальный идентификационный номер статуса | + | AUTO INCREMENT |  |  |
| name |  | TEXT | Название статуса | + | Здесь должен быть совет, но его нет , сообщите в тех.поддержку |  |  |
| **Table “Measure”** | | | | | | | | |
| Таблица содержит справочник с мерами измерения | id\_measure | PK | INT | Уникальный идентификационный номер срока меры измерения | + | AUTO INCREMENT |  |  |
| name |  | TEXT | Название меры | + | Нельзя создать запись без этого поля |  |  |
| **Table “Product”** | | | | | | | | |
| Таблица содержит информацию о продукте | id\_product | PK | INT | Уникальный идентификационный номер продукта | + | AUTO INCREMENT |  | PRIMARY |
| product\_name |  | VARCHAR(200) | Название продукта | + | Нельзя создать запись без этого |  |  |
| row\_deleted\_time |  | DATETIME | Дата изменения строки | + | 31.12.2999 |  |  |
| row\_change\_time |  | DATETIME | Дата удаления строки | + | 31.12.2999 |  |  |
| **Table “Products\_Tips”** | | | | | | | | |
| Таблица содержит связь продуктов и советов | id\_product | FK | INT | Ключ к таблице Product | + | Нельзя создать запись без этого поля |  | product.id\_product |
| id\_tips | FK | INT | Ключ к таблице Tips | + | Нельзя создать запись без этого поля |  | tips.id\_tips |
| **Table “Recipes\_products”** | | | | | | | | |
| Таблица содержит связь продуктов и рецептов | id\_product | FK | INT | Ключ к таблице Product | + | 31.12.2999 |  | product.id\_product |
| id\_recipes | FK | INT | Ключ к таблице Recipies | + | 31.12.2999 |  | recipes.id\_recipes |
| product\_quantity |  | REAL | Количество продуктов | + | 1 | не может быть меньше 0 |  |
| id\_measure | FK | INT | Ключ к таблице Measure | + | 31.12.2999 |  | measure.id\_measure |
| **Table “Recipes”** | | | | | | | | |
| Таблица справочник о рецетах | id\_recipes | PK | INT | Уникальный идентификационный номер рецепта | + | AUTO INCREMENT |  |  |
| id\_step | FK | INT | Ключ к таблице Steps | + | Нельзя создать запись без этого поля |  |  |
| name |  | TEXT | Название рецепта | + | Нельзя создать запись без этого поля |  |  |
| description |  | LONGTEXT | Описание блюда | - |  |  |  |
| row\_deleted\_time |  | DATETIME | Дата удаления строки | + | 31.12.2999 |  |  |
| row\_change\_time |  | DATETIME | Дата изменения строки | + | 31.12.2999 |  |  |
| **Table “Steps”** | | | | | | | | |
| Таблица c этапами готовки |  |  |  |  |  |  |  |  |
| id\_step | PK | INT | Уникальный идентификационный номер шага | + | AUTO INCREMENT |  | PRIMARY |
| id\_parent |  | INT | Номер предыдущего шага(для первого шага id\_step=id\_parent) | + | Для первого шага id\_step=id\_parent |  |  |
| row\_deleted\_time |  | DATETIME | Дата удаления строки | + | 31.12.2999 |  |  |
| row\_change\_time |  | DATETIME | Дата изменения строки | + | 31.12.2999 |  |  |
| **Table “Products\_Tips”** | | | | | | | | |
| Таблица содержит связь продуктов и советов | id\_product | FK | INT | Ключ к таблице Product | + | Нельзя создать запись без этого поля |  | product.id\_product |
| id\_tips | FK | INT | Ключ к таблице Tips | + | Нельзя создать запись без этого поля |  | tips.id\_tips |
| **Table “Tips”** | | | | | | | | |
| Таблица содержит информацию о советах | id\_tips | PK | INT | Уникальный идентификационный номер совета | + | AUTO INCREMENT |  | PRIMARY |
| description |  | LONGTEXT | Описание совета | + | Нельзя создать запись без этого поля |  |  |
| row\_deleted\_time |  | DATETIME | Дата удаления строки | + | 31.12.2999 |  |  |
| row\_change\_time |  | DATETIME | Дата изменения строки | + | 31.12.2999 |  |  |
| **Table “Storage\_Tips”** | | | | | | | | |
| Таблица содержит связь советов и мест | id\_tips | FK | INT | Ключ к таблице Tips | + | Нельзя создать запись без этого поля |  | tips.id\_tips |
| id\_type | FK | INT | Ключ к таблице Type\_Storage | + | Нельзя создать запись без этого поля |  | type.id\_type |
| **Table “Type\_Storage”** | | | | | | | | |
| Таблица типов мест хранения | id\_type | PK | INT | Уникальный идентификационный номер типа | + | Нельзя создать запись без этого поля |  | PRIMARY |
| name |  | TEXT | Имя типа хранения | + | Нельзя создать запись без этого поля |  |  |
| **Table “Storage”** | | | | | | | | |
| Таблица содержит информацию о местах хранения и условиях | id\_storage | PK | INT | Уникальный идентификационный номер места | + | AUTO INCREMENT |  | PRIMARY |
| name | FK | TEXT | Название места хранения | + | Нельзя создать запись без этого поля |  |  |
| id\_type |  | INT | Тип места хранения | + | Нельзя создать запись без этого поля |  |  |
| temperature |  | REAL | Температура меств хранения | - |  | от -40 до 60 гадосов C\* |  |
| humidity |  | REAL | Влажность места хранения | - |  | от 0 до 100% |  |
| date\_add |  | DATETIME | Дата внесения | + | DATETIME.NOW |  |  |
| row\_deleted\_time |  | DATETIME | Дата удаления строки | + | 31.12.2999 |  |  |
| row\_change\_time |  | DATETIME | Дата изменения строки | + | 31.12.2999 |  |  |
| **Table “Storage\_User”** | | | | | | | | |
| Таблица содержит связь пользователей и мест | id\_storage | FK | INT | Ключ к таблице Storage | + | Нельзя создать запись без этого поля |  | storage.id\_storage |
| id\_user | FK | INT | Ключ к таблице User | + | Нельзя создать запись без этого поля |  | user.id\_user |
| **Table “User”** | | | | | | | | |
| Таблица содержит информацию о пользователей | id\_user | PK | INT | Уникальный идентификационный номер пользователя | + | AUTO INCREMENT |  | PRIMARY |
| id\_settings | FK | INT | Ключ к таблице Settings | + | Нельзя создать запись без этого поля |  | settings.id\_settings |
| name |  | TEXT | Имя пользователя | + | Нельзя создать запись без этого поля |  |  |
| salt |  | TEXT | Соль пароля | + | Нельзя создать запись без этого поля |  |  |
| mail |  | TEXT | Почта пользователя | + | Нельзя создать запись без этого поля |  |  |
| date\_add |  | DATETIME | Дата регистраци | + | Нельзя создать запись без этого поля |  |  |
| row\_deleted\_time |  | DATETIME | Дата удаления строки | + | 31.12.2999 |  |  |
| row\_change\_time |  | DATETIME | Дата изменения строки | + | 31.12.2999 |  |  |
| **Table “User\_Settings”** | | | | | | | | |
| Таблица содержит связь пользователей и настроек | id\_user | FK | INT | Ключ к таблице User | + | Нельзя создать запись без этого поля |  | user.id\_user |
| id\_settings | FK | INT | Ключ к таблице Settings | + | Нельзя создать запись без этого поля |  | settings.id\_settings |
| **Table “User”** | | | | | | | | |
| Таблица содержит информацию о настройках | id\_settings | PK | INT | Уникальный идентификационный номер настроек | + | AUTO INCREMENT |  | PRIMARY |
| status |  | TEXT | Настройки | + | Нельзя создать запись без этого поля |  |  |
| row\_deleted\_time |  | DATETIME | Дата удаления строки | + | 31.12.2999 |  |  |
| row\_change\_time |  | DATETIME | Дата изменения строки | + | 31.12.2999 |  |  |
| **Table “Password”** | | | | | | | | |
| Таблица содержит все хэши паролей пользователей | hash |  | TEXT | Хранит в себе хэшированную строку типа "Соль+Пароль" | + | Нельзя создать запись без этого поля |  |  |

# Листинг кода. Приложение 2.

Далее предПолный код проекта представлен по адресу https://github.com/romankravchuk/muerta

ShelfLives.vue

<template>

<div class="container">

<div class="flex gap-4">

<!--<my-button>Добавить категорию</my-button>

<my-button @click="isAdding = !isAdding"

>Добавить срок годности</my-button

>-->

<router-link class="px-4 py-2" to="/shelf-lives"

>Сроки годности</router-link

>

<router-link class="px-4 py-2 hover:bg-green rounded-lg" to="/products"

>Продукты</router-link

>

</div>

<my-dialog v-model:show="isAdding">

<shelf-form @create="createShelf">Добавление срока годности</shelf-form>

<shelf-form-create />

</my-dialog>

<my-dialog v-model:show="isChanging"

><shelf-form

:shelf\_life="current\_shelf"

@create="updateShelf

"

>Изменение срока годности</shelf-form

></my-dialog

>

<top-bar :not\_show="['sort', 'tip', 'filter', 'search']" @click\_create="isAdding = !isAdding" />

<div class="mt-5 flex">

<!-- <sidebar class="flex-1" /> -->

<table class="flex-1 rounde-xl overflow-hidden">

<!-- <caption>

Продукты

</caption> -->

<tr>

<th>Продукт</th>

<th>Кол-во</th>

<th>Место хранения</th>

<th>Дата изготовления</th>

<th>Годен до</th>

<th>Статус</th>

<th>Изменить</th>

<th>Удалить</th>

</tr>

<tr v-for="shelf\_life in shelf\_lives" :key="shelf\_life.id">

<td class="hover:underline cursor-pointer" @click="this.$router.push('/products/'+shelf\_life.product.id)">{{ shelf\_life.product.name }}</td>

<td>{{ shelf\_life.quantity }} {{ shelf\_life.measure.name }}</td>

<td>{{ shelf\_life.storage.name }}</td>

<td>

{{

new Date(shelf\_life.purchase\_date).toLocaleString("ru", {

year: "numeric",

month: "long",

day: "numeric",

})

}}

</td>

<td>

{{

new Date(shelf\_life.end\_date).toLocaleString("ru", {

year: "numeric",

month: "long",

day: "numeric",

})

}}

</td>

<td v-if="shelf\_life.purchase\_date >= shelf\_life.end\_date">Подошел к концу</td>

<td v-else-if="Math.floor((shelf\_life.purchase\_date - shelf\_life.end\_date) / (1000 \* 60 \* 60 \* 24)) < 3">Подходит к концу</td>

<td>Годен</td>

<td style="text-align: center !important">

<font-awesome-icon

@click="

current\_shelf = { ...shelf\_life };

current\_shelf.id\_product = shelf\_life.product.id;

current\_shelf.id\_measure = shelf\_life.measure.id;

current\_shelf.id\_storage = shelf\_life.storage.id;

isChanging = true;

"

:icon="['far', 'pen-to-square']"

class="text-gray-700/70 transition-all text-2xl cursor-pointer hover:text-gray-700"

/>

</td>

<td style="text-align: center !important">

<font-awesome-icon

@click="store.deleteShelf(shelf\_life.id)"

:icon="['far', 'trash-can']"

class="text-gray-700/70 transition-all text-2xl cursor-pointer hover:text-gray-700"

/>

</td>

</tr>

</table>

</div>

</div>

</template>

<script>

import Sidebar from "../components/Sidebar.vue";

import ShelfForm from "../components/ShelfForm.vue";

import { useShelfLivesStore } from "../stores/ShelfLivesStore";

import { storeToRefs } from "pinia";

import { ref } from "vue";

import { useStoragesStore } from "../stores/StoragesStore";

import TopBar from "../components/TopBar.vue";

export default {

data() {

return {

isAdding: false,

isChanging: false,

current\_shelf: {},

};

},

components: {

Sidebar,

ShelfForm,

TopBar,

},

methods: {

createProduct(product) {

this.isAdding = false;

this.store.createProduct(product);

},

changeProduct(product) {

this.isChanging = false;

this.store.updateProduct(product);

},

createShelf(shelf) {

this.store.createShelf(shelf);

this.isAdding = false;

},

updateShelf(shelf) {

this.store.updateShelf(shelf);

this.isChanging = false;

},

},

setup() {

const store = useShelfLivesStore();

const store\_storages = useStoragesStore();

const { shelf\_lives } = storeToRefs(store);

store.fetchShelfs();

/\* store.fetchStorages(); \*/

/\* const storage\_options = ref([]); \*/

/\* store\_storages.fetchStorages(); \*/

/\* store\_storages.storages.forEach((st) => { \*/

/\* storage\_options.value.push({ \*/

/\* name: st.name, \*/

/\* value: st.id, \*/

/\* }); \*/

/\* }); \*/

return { shelf\_lives, store };

},

};

</script>

<style lang="postcss" scoped>

table,

tr,

td,

th {

/\* display: block; \*/

text-align: left;

padding: 10px;

font-size: 0.9rem;

}

table {

width: 100%;

}

tr:nth-child(odd) {

@apply bg-green/40;

}

tr:first-child {

@apply bg-green/60 text-gray-900;

}

tr:nth-child(even) {

@apply bg-green/20;

}

tr {

padding: 0.7em 2vw;

}

</style>

Main.js

import { createApp } from "vue";

import "./style.css";

import App from "./App.vue";

import router from "./router";

import components from "./components/UI";

import { createPinia } from "pinia";

/\* import the fontawesome core \*/

import { library } from "@fortawesome/fontawesome-svg-core";

/\* import font awesome icon component \*/

import { FontAwesomeIcon } from "@fortawesome/vue-fontawesome";

import axios from "axios";

// axios.defaults.headers["Access-Control-Allow-Origin"] = "\*"

axios.defaults.baseURL = "https://muerta.up.railway.app/api/v1";

// axios.defaults.withCredentials = true;

// axios.defaults.headers.common['Content-Type'] = 'application/json';

axios.defaults.withCredentials = true;

// axios.defaults.headers.common["ngrok-skip-browser-warning"] = "123";

import \* as yup from "yup";

yup.setLocale({

string: {

min: "Минимум ${min} символов",

},

mixed: {

required: "Данное поле обязательно",

},

number: {

moreThan: "Число должно быть больше ${moreThan}",

},

});

import {

faPlusSquare,

faPenToSquare,

faTrashCan,

} from "@fortawesome/free-regular-svg-icons";

import {

faChevronUp,

faBars,

faArrowLeft,

} from "@fortawesome/free-solid-svg-icons";

const pinia = createPinia();

const app = createApp(App);

library.add(

faBars,

faPlusSquare,

faChevronUp,

faPenToSquare,

faTrashCan,

faArrowLeft

);

components.forEach((component) => {

app.component(component.name, component);

});

app

.component("font-awesome-icon", FontAwesomeIcon)

.use(pinia)

.use(router, axios)

.mount("#app");

index.js

import { createRouter, createWebHistory } from "vue-router";

import Home from "../views/Home.vue";

import Recipes from "../views/Recipes.vue";

import ShelfLives from "../views/ShelfLives.vue";

import RecipeDetail from "../views/RecipeDetail.vue";

import ProductDetail from "../views/ProductDetail.vue";

import Storages from "../views/Storages.vue";

import SignUp from "../views/SignUp.vue";

import LogIn from "../views/LogIn.vue";

import NotFound from "../views/NotFound.vue";

import { useUserStore } from "../stores/UserStore";

import Tips from "../views/Tips.vue";

import Products from "../views/Products.vue";

import StorageTypes from "../views/StorageTypes.vue";

import Steps from "../views/Steps.vue";

const routes = [

{

path: "/",

name: "Home",

component: Home,

},

{

path: "/recipes",

name: "Recipes",

component: Recipes,

},

{

path: "/recipes/:id",

name: "Recipe",

component: RecipeDetail,

},

{

path: "/shelf-lives",

name: "ShelfLives",

component: ShelfLives,

meta: {

requiresAuth: true,

},

},

{

path: "/products/:id",

name: "Product",

component: ProductDetail,

meta: {

requiresAuth: true,

},

},

{

path: "/storages",

name: "Storages",

component: Storages,

meta: {

requiresAuth: true,

},

},

{

path: "/sign-up",

name: "SignUp",

component: SignUp,

},

{

path: "/login",

name: "Login",

component: LogIn,

},

{

path: "/404",

component: NotFound,

},

{

path: "/:catchAll(.\*)",

redirect: "/404",

},

{

path: "/tips",

component: Tips,

name: "Tips",

},

{

path: "/products",

component: Products,

name: "Products",

meta: {

requiresAuth: true,

},

},

{

path: "/storage-types",

component: StorageTypes,

name: "StorageTypes",

meta: {

requiresAuth: true,

},

},

{

path: "/steps",

component: Steps,

name: "Steps",

meta: {

requiresAuth: true,

},

},

];

const router = createRouter({

history: createWebHistory("/kors/"),

routes,

});

router.beforeEach((to, from, next) => {

const user\_store = useUserStore();

user\_store.fetchCurrentUser();

if (to.matched.some((record) => record.meta.requiresAuth)) {

if (user\_store.user == "") {

next({

name: "Login",

});

} else {

next();

}

} else {

next();

}

});

export default router;

ShelfLivesStore.js

import { defineStore } from "pinia";

import { useStoragesStore } from "./StoragesStore";

import { useUserStore } from "./UserStore";

import axios from "axios";

import { storeToRefs } from "pinia";

export const useShelfLivesStore = defineStore("shelf\_lives", {

state: () => ({

shelf\_lives: [],

}),

actions: {

fetchShelfs() {

const store\_user = useUserStore();

const { user } = storeToRefs(store\_user);

axios.get(`/users/${user.value.id}/shelf-lives/?limit=30`).then((res) => {

if (res.status == 200) {

this.shelf\_lives = res.data.data["shelf-lives"];

}

});

},

fetchStorages() {

const storages = useStoragesStore();

storages.fetchStorages();

this.products.forEach((product) => {

product.storage = storages.getStorageById(product.storage)[0];

});

// this.products = this.products.map(

// (x) => (x.storage = storages.getStorageById(x.storage))

// );

},

createShelf({

id\_product,

id\_storage,

id\_measure,

quantity,

purchase\_date,

end\_date,

}) {

const store\_user = useUserStore();

const { user } = storeToRefs(store\_user);

axios

.post(`/users/${user.value.id}/shelf-lives`, {

id\_product: id\_product,

id\_storage: id\_storage,

id\_measure: id\_measure,

quantity: quantity,

purchase\_date: purchase\_date,

end\_date: end\_date,

})

.then((res) => {

if (res.status == 200) {

this.fetchShelfs();

this.fetchStorages();

alert("Срок годности успешно создан");

} else {

alert("Не удалось создать срок годности");

}

})

.catch((e) => console.log(e));

},

deleteShelf(id) {

let isCertain = confirm("Вы действительно хотите удалить срок годности?");

if (!isCertain) {

return false

}

axios.delete(`/shelf-lives/${id}`).then((res) => {

if (res.status == 200) {

this.shelf\_lives = this.shelf\_lives.filter((sh) => sh.id !== id);

alert("Срок годности успешно удален");

this.fetchShelfs();

this.fetchStorages();

}

});

},

updateShelf({

id,

id\_product,

id\_storage,

id\_measure,

quantity,

purchase\_date,

end\_date,

}) {

axios

.put(`/shelf-lives/${id}`, {

id\_product: id\_product,

id\_storage: id\_storage,

id\_measure: id\_measure,

quantity: quantity,

purchase\_date: purchase\_date,

end\_date: end\_date,

})

.then((res) => {

if (res.status == 200) {

alert("Срок годности успешно изменен");

this.fetchShelfs();

}

})

.catch((e) => {

console.log(e);

alert("Не удалось изменить срок годности");

});

},

},

});

TopBar.vue

<template>

<div class="flex max-[452px]:flex-wrap gap-y-3 justify-between mt-2 mb-4">

<div class="flex w-full sm:w-auto" style="align-content: center">

<div

style="user-select: none"

class="shadow-xl py-2 px-4 bg-gray-100 rounded-2xl hover:bg-gray-200 cursor-pointer transition-all inline-block w-fit"

v-if="!not\_show.includes('filter')"

@click="

sidebarIsHidden = !sidebarIsHidden;

emit('click\_filter');

"

>

<svg

xmlns="http://www.w3.org/2000/svg"

width="24"

height="24"

viewBox="0 0 24 24"

>

<path

d="M19.479 2l-7.479 12.543v5.924l-1-.6v-5.324l-7.479-12.543h15.958zm3.521-2h-23l9 15.094v5.906l5 3v-8.906l9-15.094z"

/>

</svg>

</div>

<div

v-if="!not\_show.includes('create')"

:class="{'hidden': !user}"

style="user-select: none"

class="shadow-xl max-[452px]:text-center py-2 px-4 bg-gray-100 rounded-2xl hover:bg-gray-200 cursor-pointer transition-all ml-4 inline-block w-fit max-[452px]:w-full"

@click="createClick"

>

Создать

</div>

</div>

<div class="flex max-[640px]:w-full" style="align-content: center">

<div

style="user-select: none"

v-if="!not\_show.includes('sort')"

@click="emit('click\_sort')"

class="shadow-xl py-2 px-4 bg-gray-100 rounded-2xl hover:bg-gray-200 cursor-pointer transition-all inline-block w-fit"

>

<svg

xmlns="http://www.w3.org/2000/svg"

width="24"

height="24"

viewBox="0 0 24 24"

>

<path

d="M8 10v4h4l-6 8-6-8h4v-4h-4l6-8 6 8h-4zm10.19-1.016h3.237l.581 1.807h1.992l-3.11-8.791h-2.078l-3.138 8.792h1.921l.595-1.808zm1.641-4.974l1.11 3.459h-2.251l1.141-3.459zm-1.393 16.439h4.562v1.551h-6.787v-1.551l4.493-5.684h-4.38v-1.557h6.662v1.473l-4.55 5.768z"

/>

</svg>

</div>

<input

type="text"

class="shadow-xl py-2 px-4 bg-gray-100 rounded-2xl hover:bg-gray-200 transition-all ml-4 inline-block w-full sm:w-fit max-[452px]:w-full"

:value="search"

v-if="!not\_show.includes('search')"

@input="emit('update:search', $event.target.value)"

placeholder="Поиск..."

/>

<!-- <svg xmlns="http://www.w3.org/2000/svg" width="24" height="24" viewBox="0 0 24 24"><path d="M21.172 24l-7.387-7.387c-1.388.874-3.024 1.387-4.785 1.387-4.971 0-9-4.029-9-9s4.029-9 9-9 9 4.029 9 9c0 1.761-.514 3.398-1.387 4.785l7.387 7.387-2.828 2.828zm-12.172-8c3.859 0 7-3.14 7-7s-3.141-7-7-7-7 3.14-7 7 3.141 7 7 7z"/></svg> -->

</div>

</div>

</template>

<script setup>

import { useUserStore } from "../stores/UserStore";

import { storeToRefs } from "pinia";

import { ref } from "vue";

const sidebarIsHidden = ref(true);

const props = defineProps({

search: String,

not\_show: {

type: Array,

default: [],

},

});

const emit = defineEmits(['click\_create'])

const createClick = () => {

emit("click\_create")

}

const store = useUserStore();

store.fetchCurrentUser();

const { user } = storeToRefs(store);

</script>

<style></style>

ShelfFormEdit.vue

<template>

<div>

<h1 class="text-lg font-semibold"><slot></slot></h1>

<form

class="space-y-4 flex flex-col [&>\*]:p-4 mt-5"

@submit.prevent="createShelf"

>

<select v-model.number="shelf\_life.id\_product">

<option value="" disabled selected>Продукт</option>

<option

v-for="product in products"

:key="product.id"

:value="product.id"

>

{{ product.name }}

</option>

</select>

<input

v-model.number="shelf\_life.quantity"

type="text"

placeholder="Количество"

/>

<select v-model.number="shelf\_life.id\_measure">

<option value="" disabled selected>Мера измерения</option>

<option

v-for="measure in measures"

:key="measure.id"

:value="measure.id"

>

{{ measure.name }}

</option>

</select>

<select v-model.number="shelf\_life.id\_storage">

<option value="" disabled selected>Место хранения</option>

<option

v-for="storage in storages"

:value="storage.id"

:key="storage.id"

>

{{ storage.name }}

</option>

</select>

<!-- <input

v-model="product.purchase\_date"

type="text"

placeholder="Дата начала"

/> -->

<VueDatePicker

style="padding: 0 !important"

class="p-0"

:clearable="false"

placeholder="Дата изготовления"

locale="ru"

format="dd.MM.yyyy"

:enable-time-picker="false"

auto-apply

v-model="purchase\_date"

required

></VueDatePicker>

<VueDatePicker

style="padding: 0 !important"

:clearable="false"

placeholder="Годен до"

locale="ru"

format="dd.MM.yyyy"

:enable-time-picker="false"

auto-apply

v-model="end\_date"

required

></VueDatePicker>

<!-- <input

v-model="product.date\_end"

type="text"

placeholder="Дата окончания"

/> -->

<my-button type="submit">Создать</my-button>

</form>

</div>

</template>

<script>

import { useMeasureStore } from "../stores/MeasuresStore";

import { useStoragesStore } from "../stores/StoragesStore";

import { useProductsStore } from "../stores/ProductsStore";

import { storeToRefs } from "pinia";

import VueDatePicker from "@vuepic/vue-datepicker";

import "@vuepic/vue-datepicker/dist/main.css";

import { ref } from "vue";

export default {

props: {

shelf\_life: {

default: {

id\_product: "",

quantity: "",

id\_storage: "",

purchase\_date: null,

end\_date: null,

id\_measure: "",

},

},

},

data() {

return {

purchase\_date: null,

end\_date: null,

};

},

components: {

VueDatePicker,

},

methods: {

createShelf() {

this.shelf\_life.purchase\_date = this.purchase\_date.toISOString();

this.shelf\_life.end\_date = this.end\_date.toISOString();

this.$emit("create", { ...this.shelf\_life });

this.shelf\_life.name = "";

this.shelf\_life.quantity = "";

this.shelf\_life.storage = "";

this.shelf\_life.date\_start = "";

this.shelf\_life.date\_end = "";

},

},

setup() {

const store\_measure = useMeasureStore();

const store\_storages = useStoragesStore();

const store\_products = useProductsStore();

const { measures } = storeToRefs(store\_measure);

const { storages } = storeToRefs(store\_storages);

const { products } = storeToRefs(store\_products);

store\_measure.fetchMeasures();

store\_storages.fetchStorages();

store\_products.fetchProducts();

return { measures, storages, products };

},

updated() {

if (this.shelf\_life.purchase\_date !== null) {

let t\_purchase\_date = new Date(this.shelf\_life.purchase\_date);

let t\_end\_date = new Date(this.shelf\_life.end\_date);

this.purchase\_date = t\_purchase\_date.toISOString()

this.end\_date = t\_end\_date.toISOString()

}

},

};

</script>

<style scoped></style>