**Техническое задание на разработку списка заданий**

1) Используемый стек (обязательно)

Vue,apollo, TS, VUEX,vuetify, graphql, Webpack

2) В качестве исходных данных использовать mock-объекты apollo

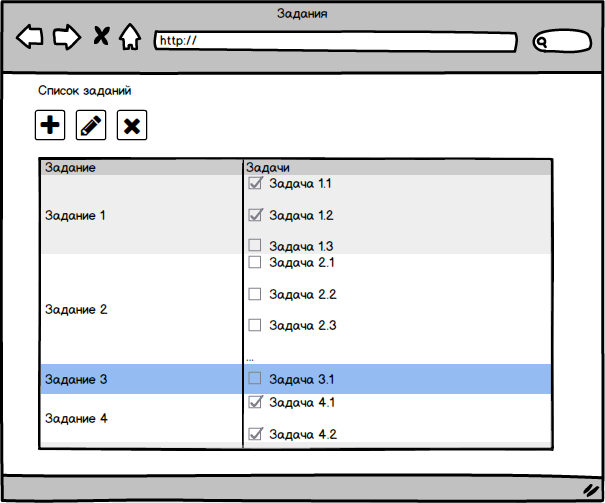
Средствами Vue.js реализуйте небольшое SPA приложение для заданий.

Каждое задание имеет название и список задач (taskList), далее - taskList. Каждый пункт taskList состоит из чекбокса и относящейся к нему текстовой подписи.

Приложение состоит из двух форм.

На **главной странице** отображается список всех заданий. Для каждого задания отображается заголовок и taskList (если задач в нем много, сокращенный до нескольких пунктов; при наведении на строку списка в таком случае необходимо во всплывающей подсказке выводить список всех задач задания, отмечая выполненные значком «√» ли «v» (галочка)) без возможности отмечать выполненные задачи. Действия на главной:

* перейти к созданию нового задания
* перейти к изменению задания
* удалить (необходим запрос подтверждения удаления)

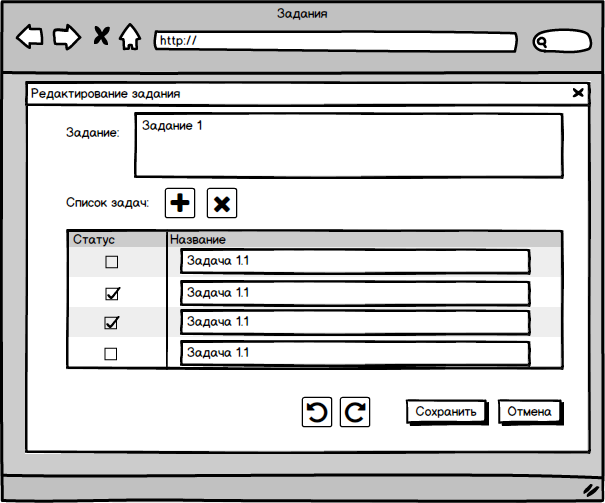


Модальное **окно изменения задания** позволяет определенное задание отредактировать, отметить пункты taskList, а после сохранить изменения. Действия с заданием:

* сохранить изменения
* отменить редактирование (необходимо подтверждение)
* удалить (необходимо подтверждение) – допускается этот пункт оставить только в списке задач, а не в редакторе задания
* отредактировать название задания
* отменить последнее внесенное изменение
* повторить последнее отмененное изменение

Действия с пунктами taskList (задачами текущего задания):

* добавить
* удалить
* отредактировать текст (без открытия отдельного окна редактирования текста задачи)
* отметить как выполненную (проставить чекбокс)



Требования к функционалу:

* Все действия в приложении должны происходить без перезагрузки страницы.
* Подтверждение действий (удалить задание) выполняется с помощью диалогового окна.
* Интерфейс должен отвечать требованиям usability и реализован на основе vuetify.
* После перезагрузки страницы состояние списка заданий должно сохраняться.
* Можно пренебречь несоответствием редактирования текста с помощью кнопок отменить/повторить и аналогичным действиям с помощью комбинация клавиш (Ctrl+Z, Command+Z, etc.).

Технические требования:

* В качестве языка разработки допускается использовать JavaScript или TypeScript.
* В качестве сборщика, если это необходимо, используйте Webpack.
* Верстка должна быть выполнена с использования UI библиотек (Vuetify).
* Адаптивность не обязательна, но приветствуется.
* Логика приложения должна быть разбита на разумное количество самодостаточных Vue-компонентов.
* Серверная часть может вообще отсутствовать (работа только на клиентской стороне, хранение в сессии, в localStorage или в Cookies браузера).
* Возможен вариант реализации с полноценной серверной частью – либо без БД (например, MongoDB), либо с СУБД (тогда это должна быть СУБД PostgreSQL). В случае наличия серверной части она должна быть реализована на основе языка программирования Java 8, желательно использование фреймворка Spring и библиотеки hibernate. Необходимо разработка ER-диаграммы базы данных (выделение основных сущностей и связей между ними).

Особое внимание стоит обратить на следующие моменты:

* Код должен быть написан понятно и аккуратно, с соблюдением табуляции и прочих элементов написания, без лишних элементов и функций, не имеющих отношения к функционалу тестового задания, снабжен понятными комментариями.
* Читабельность и наличие элементарной архитектуры.
* Чистота и оформление кода — не менее важный фактор. Код должен быть написан в едином стиле (желательно в рекомендуемом для конкретного языка). Также к чистоте относятся отсутствие копипаста и дублирования логики.

Тестовое задание должно быть предоставлено в следующем виде:

* Ссылка на публичный репозиторий (GitHub, BitBucket, GitLab) с исходным кодом.
* Должна быть инструкция по сборке и разворачиванию

## Дополнительный функционал

### Работа с пользователями

Данный функционал может реализовываться при наличии серверной части.

Должна быть реализована база данных в части ведения пользователей, формы регистрации пользователей (предлагается минимальный функционал, см. другие задачи) и аутентификации по логину и паролю (после авторизации использовать токен при работе).

1. При создании задания нужно сохранять пользователя, который его создал. Если не выбирался пользователь, которому задание назначено, он также должен быть проставлен автоматически и совпадать с создавшим пользователем.
2. В списке заданий нужно выводить пользователя, который создал задание, а также пользователя, которому оно назначено.
3. В списке заданий возможность удаления задания должна быть только у пользователя, который его создал.
4. В карточке задания нужно добавить возможность назначения задания пользователю (выбирать из всех зарегистрированных пользователей), это должны иметь возможность делать только тот, кто создал задание, и тот, кому оно в настоящий момент назначено.
5. В карточке задания в списке задач проставлять или снимать чекбоксы имеет право только тот пользователь, которому в данный момент назначено задание.
6. В карточке задания пользователь, создавший задание, должен иметь возможность изменять название задания, редактировать список задач в полном объеме, назначать исполнителя в любой момент времени.
7. Все остальные пользователи (кроме создавшего и того, кому оно назначено) не должны иметь возможности изменения данных задания (компоненты должны быть заблокированы).