Техническое задание на программу для ЭВМ «Выбор элемента дерева» версия 1.

1. Описание

Программа представляет собой приложение, выбирающее элемент из дерева, представленного в виде:

* структуры в базе данных,
* структуры в файле (в формате json),
* структуры, переданной в API (в формате json),

на основе нескольких параметров:

* вес элемента,
* счетчик предыдущих выборов,
* число, сгенерированное генератором случайных чисел,
* либо любых других параметров, определенных в структуре элемента дерева и в формуле выбора.

Приложение состоит из следующих частей:

* + - * сервер, реализующий логику работы с БД, предоставляющий веб-приложение и API. Поставляется в вариантах для Windows и Linux;
      * web-приложение ;
      * десктопное приложение.

Для всех частей приложения прилагаются программы установки (кроме web-приложения).

1. Требования

Исходное дерево может храниться в базе данных на сервере, в локальной базе данных, либо в файле.

Web-клиент работает только с сервером, десктопный клиент может работать автономно, с локальной базой данных (SQLite).

Формулы выбора элемента хранятся в базе данных, одна из формул - по умолчанию, устанавливается автоматически для нового дерева.

Должен быть предусмотрен механизм синхронизации локальной и серверной баз данных.

Авторизация не предусмотрена.

1. Логика работы

Пользователь работает либо в web-интерфейсе, либо в клиентском приложении. Логика работы в этих двух интерфейсах за небольшим исключением не отличается.

Основные действия пользователя:

1. Создание дерева, редактирование (в том числе заполнение элементами), удаление.
2. Добавление, редактирование, удаление формул получения следующего элемента
3. Получение следующего элемента
4. В интерфейсе десктопного клиента: разрешение конфликтов синхронизации, построение дерева из ФС
5. Вызов API: вычисление следующего элемента без добавления дерева в БД (вызов принимает в качестве параметров дерево и формулу)

В системе существуют следующие требования:

- уникальность наименований элементов дерева, имеющих общего родителя (непосредственного)

- Уникальность наименований деревьев и формул

- формула должна соответствовать одному из двух шаблонов :

1) AGR(Expression), где

* AGR – одна из агрегатных функций Max, Min, Avg
* Expression – выражение, возвращающее числовое значение, состоящее из математических операторов библиотеки Ncalc, чисел и полей элемента дерева (Weight, SelectCount, поля, указанные в поле AddFields)

\* Avg ищет элемент соответствия начиная от среднего значения (округленного) и постепенно увеличивая погрешность по единице в обе стороны, пока н будет найден хотя бы один элемент

2) Expression, где Expression – выражение, возвращающее булево значение, состоящее из математических операторов библиотеки Ncalc, чисел и полей элемента дерева (Weight, SelectCount, поля, указанные в поле AddFields)

В результате возвращается первый из элементов, удовлетворяющий выражению, либо null, если ни один из элементов не соответствует выражению.

1. Механизм синхронизации баз данных.

Синхронизация баз данных работает на стороне клиента в фоновом режиме. Логика работы сервиса синхронизации:

- При запуске клиента из таблицы конфигурации получается дата последней синхронизации.

- При каждой итерации сервиса с сервера вызывается список изменений по каждой из таблиц.

- Делается попытка сохранения изменений в локальную базу данных. Если сохранить изменения не получается (из-за конфликта версий), сохранение данных на сервер блокируется до разрешения конфликтов, список конфликтов записывается в таблицу конфликтов, на главном окне появляется уведомление о необходимости разрешения конфликтов синхронизации, сервис итерации не выполняет дальнейшую синхронизацию.

- В случае успешного прохождения предыдущего шага делается попытка записи несинхронизированных записей из локальной БД на серверную.

- Если есть записи, которые невозможно сохранить на сервер (конфликт версий), сохранение данных на сервер блокируется до разрешения конфликтов, список конфликтов записывается в таблицу конфликтов, на главном окне появляется уведомление о необходимости разрешения конфликтов синхронизации, сервис итерации не выполняет дальнейшую синхронизацию.

- Если все шаги завершены успешно, в таблицу конфигурации записывается дата последней синхронизации.

1. Технологии
   1. Общие компоненты

* Newtonsoft.Json (для работы с Json)
* NCalc (парсинг формул)
* .Net (Framework, Core)
  1. Сервер
* EntityFramework
* ASP .NetCore
* .NetCore 3.x
  1. Клиент
* SqLite (+ EntityFramework?)
* .Net Framework 4.7
* WPF
* HttpClient
  1. Web-client
* ASP .NetCore
* ReactJS или AngularJS
* Html
* Css
* JQuery
  1. Базы данных
* Сервер: PostgreSQL
* Клиент: SQLite
  1. Тестирование
* XUnit
* Moq
  1. Установщик
* WixInstaller

1. Модели данных
   1. Клиентские модели
      1. Базовый класс

Базовый класс Entity содержит общие поля (свойства) для всех моделей данных

***Entity***

***{***

***public Guid Id;***

***}***

* + 1. Дерево
       - Описание

***Tree: Entity***

***{***

***public string Name;***

***public string Description;***

***public Guid FormulaId;***

***}***

* + - * Элементы дерева

***TreeItem: Entity***

***{***

***public string Name;***

***public Guid TreeId;***

***public Guid? ParentId;***

***public bool IsLeaf;***

***public int SelectCount;***

***public int Weight;***

***public json AddFields;***

***}***

* + - * История изменений дерева

***Tree: EntityHistory***

***{***

***public string Name;***

***public string Description;***

***public Guid FormulaId;***

***}***

* + - * История изменений элементов дерева

***TreeItem: EntityHistory***

***{***

***public string Name;***

***public Guid TreeId;***

***public Guid? ParentId;***

***public bool IsLeaf;***

***public int SelectCount;***

***public int Weight;***

***public json AddFields;***

***}***

* + 1. Формула

***Formula: Entity***

***{***

***public string Name;***

***public string Text;***

***public bool IsDefault;***

***}***

* + 1. История изменений формул

***Formula: EntityHistory***

***{***

***public string Name;***

***public string Text;***

***public bool IsDefault;***

***}***

* + 1. Запрос выборки элемента

***SelectRequest***

***{***

***Public Guid? TreeId;***

***public int Count = 1;***

***Public bool LeafOnly = true;***

***}***

* + 1. Запрос выборки элемента дерева (дерево в запросе)

***SelectCustomRequest***

***{***

***Public IEnumerable<TreeItem> Tree;***

***Public string Formula;***

***public int Count = 1;***

***Public bool LeafOnly = true;***

***}***

* + 1. Результат выборки элемента

***SelectResponse***

***{***

***Public IEnumerable<SelectResponseElement> Result;***

***}***

***SelectResponseElement***

***{***

***Public Guid Id;***

***Public string Name;***

***Public string FullPath;***

***}***

* 1. Серверные (внутренние) модели
     1. Базовый класс

Базовый класс Entity содержит общие поля (свойства) для всех моделей данных

***Entity***

***{***

***public Guid Id;***

***public Guid VersionId;***

***public int VersionNum;***

***public DateTime VersionDate;***

***public bool IsDeleted;***

***}***

* + 1. Дерево
       - Описание

***Tree: Entity***

***{***

***public string Name;***

***public string Description;***

***public Guid FormulaId;***

***}***

* + - * Элементы дерева

***TreeItem: Entity***

***{***

***public string Name;***

***public Guid TreeId;***

***public Guid? ParentId;***

***public bool IsLeaf;***

***public int SelectCount;***

***public int Weight;***

***public json AddFields;***

***}***

* + - * История изменений дерева

***Tree: EntityHistory***

***{***

***public string Name;***

***public string Description;***

***public Guid FormulaId;***

***}***

* + - * История изменений элементов дерева

***TreeItem: EntityHistory***

***{***

***public string Name;***

***public Guid TreeId;***

***public Guid? ParentId;***

***public bool IsLeaf;***

***public int SelectCount;***

***public int Weight;***

***public json AddFields;***

***}***

* + 1. Формулы

***Formula: Entity***

***{***

***public string Name;***

***public string Text;***

***public bool IsDefault;***

***}***

* + 1. Фильтр
  1. Модели базы данных

1. Сервер
   1. Описание

Сервер представляет собой web-сервис, выполняющий функции работы с базой данных через ORM (Entity Framework). В первой версии программы реализуется коннектор только к PostgreSql.

Бизнес-логика приложения вынесена в компонент, доступный всем частям приложения для разных кейсов работы приложения.

Для работы с внешними клиентами сервер предоставляет API, хост которого располагается в той же сборке, что и Web клиент.

Сервер написан на C# под .Net Core 3.x.

* 1. Работа с базой данных

Работа с базой данных сделана через обертку (Repository).

Реализуется паттерн SoftDeleted (поле IsDeleted).

Поддерживается версионность записей для синхронизации с клиентом.

Реализуются следующие общие методы:

* + 1. Методы работы с данными
       - Get

Получение списка сущностей Entity согласно заданного фильтра. Фильтр содержит значения полей для фильтрации, пейджинг, сортировку, метод обогащения

Сигнатура:

***public Task<IEnumerable<Entity>> GetAsync(Filter<Entity> filter, CancellationToken token)***

* + - * GetItem

Получение сущности Entity по Id с обогащением (с использованием переданного метода). В случае отсутствия сущности в базе возвращается null

***public Task<Entity> GetItemAsync(Guid id, Action<Entity> addAction, CancellationToken token)***

* + - * Add

Добавление сущности Entity c генерированием Id по переданным полям. Если список полей для генерирования Id пустой или null, генерируется случайный Id. При добавлении используется оптимистичная блокировка по Id. Сигнатура:

***public Task<Entity> AddAsync(Entity entity, string[] keyFields, CancellationToken token)***

* + - * Update

Обновление сущности Entity. При обновлении используется оптимистичная блокировка по полям Version… . Сигнатура:

***public Task<Entity> UpdateAsync(Entity entity, CancellationToken token)***

* + - * Delete

Удаление сущности Entity по Id. Используется паттерн SoftDeleted. Сигнатура:

***public Task<Entity> DeleteAsync(Guid id, CancellationToken token)***

* 1. API
     1. Дерево
        + Метод получения списка деревьев

Метод принимает следующие параметры:

* Фильтр по имени (поиск по вхождению)
* Номер страницы
* Число элементов
* Поле сортировки

Возвращает список деревьев (Tree – клиентская модель), соответствующих фильтрам, в формате json. Сигнатура:

GET /api/v1/tree/?name=name&page=1&size=10&sort=name

* + - * Метод получения дерева по Id

Метод принимает следующие параметры:

* id

Возвращает дерево (Tree – клиентская модель) в формате json. Сигнатура:

GET /api/v1/tree/{id}

* + - * Метод получения всех элементов дерева

Метод принимает следующие параметры:

* id

Возвращает список элементов дерева (TreeItem – клиентская модель) в формате json. Сигнатура:

GET /api/v1/tree/{id}/items

* + - * Метод добавления дерева

Метод принимает следующие параметры:

* TreeCreator

Возвращает добавленное дерево (Tree – клиентская модель) в формате json. Сигнатура:

POST /api/v1/tree/

* + - * Метод изменения дерева

Метод принимает следующие параметры:

* TreeUpdater

Возвращает измененное дерево (Tree – клиентская модель) в формате json. Сигнатура:

PUT /api/v1/tree/

* + - * Метод изменения элементов дерева

Метод принимает следующие параметры:

* TreeItemsUpdater

Возвращает измененный список (TreeItem – клиентская модель) в формате json. Сигнатура:

PUT /api/v1/tree/{id}/items

* + - * Метод удаления дерева

Метод принимает следующие параметры:

* id

Возвращает удаленное дерево (Tree – клиентская модель) в формате json. Элементы дерева удаляются так же. Сигнатура:

DELETE /api/v1/tree/{id}

* + - * Метод получения списка изменений дерева

Метод принимает следующие параметры:

* id
* Фильтр по имени (поиск по вхождению)
* Дата изменения от
* Дата изменения до
* Номер страницы
* Число элементов
* Поле сортировки

Возвращает список изменений деревьев (TreeHistory – клиентская модель), соответствующих фильтрам, в формате json. Сигнатура:

GET /api/v1/tree/changes/?id=id&name=name&page=1&size=10&sort=name

* + - * Метод получения списка изменений элементов дерева

Метод принимает следующие параметры:

* Id (дерева)
* Фильтр по имени дерева (поиск по вхождению)
* Фильтр по имени элемента (поиск по вхождению)
* Дата изменения от
* Дата изменения до
* Номер страницы
* Число элементов
* Поле сортировки

Возвращает список изменений элементов дерева (TreeItemHistory – клиентская модель), соответствующих фильтрам, в формате json. Сигнатура:

GET /api/v1/tree/items/changes/?id=id&treename=treename&name=name&page=1&size=10&sort=name

* + 1. Формулы
       - Метод получения списка формул

Метод принимает следующие параметры:

* Фильтр по имени (поиск по вхождению)
* Номер страницы
* Число элементов
* Поле сортировки

Возвращает список формул (Formula – клиентская модель), соответствующих фильтрам, в формате json. Сигнатура:

GET /api/v1/formula/?name=name&page=1&size=10&sort=name

* + - * Метод получения формулы по Id

Метод принимает следующие параметры:

* id

Возвращает формулу (Formula – клиентская модель) в формате json. Сигнатура:

GET /api/v1/ formula /{id}

* + - * Метод добавления формулы

Метод принимает следующие параметры:

* Formula (клиентская модель)

Возвращает добавленную формулу (Formula – клиентская модель) в формате json. Сигнатура:

POST /api/v1/ formula /

* + - * Метод изменения формулы

Метод принимает следующие параметры:

* Formula (клиентская модель)

Возвращает измененную формулу (Formula – клиентская модель) в формате json. Сигнатура:

PUT /api/v1/ formula /

* + - * Метод удаления формулы

Метод принимает следующие параметры:

* id

Возвращает удаленную формулу (Formula – клиентская модель) в формате json. Сигнатура:

DELETE /api/v1/ formula /{id}

* + - * Метод получения списка изменений

Метод принимает следующие параметры:

* id
* Фильтр по имени (поиск по вхождению)
* Дата изменения от
* Дата изменения до
* Номер страницы
* Число элементов
* Поле сортировки

Возвращает список изменений формул (FormulaHistory – клиентская модель), соответствующих фильтрам, в формате json. Сигнатура:

GET /api/v1/formula/changes/?id=id&name=name&page=1&size=10&sort=name

* + 1. Синхронизация
       - Метод синхронизации SyncInit

Метод не принимает параметров

В ответ выгружаются таблицы Tree, TreeItem и Formula (без удаленных записей) в формате json. Сигнатура:

GET /api/v1/sync/init

* + - * Метод синхронизации Sync

Метод принимает таблицы Tree, TreeItem и Formula с удаленными записями в формате json. В ответ выгружаются таблицы Tree, TreeItem и Formula (без удаленных записей) в формате json, а также список конфликтов при вставке. Сигнатура:

POST /api/v1/sync

* 1. Web-приложение (см. Web клиент)

1. Клиент
   1. Главное окно
      1. Меню
         * Файл

* Подключиться к серверу… (*подключение к серверу*)
* Синхронизация с сервером (*синхронизация локальной базы данных с серверной*)
* ----
* Добавить дерево (*добавить дерево вручную*)
* Импорт из файла… (*импорт дерева из файла*)
* Экспортировать в файл… (*сохранение выбранных деревьев в файлы*)
* ----
* Выход (*закрытие программы*)
  + - * Окно
* Дерево
* Формулы
  + - * Помощь
* О программе
* Сообщение об ошибке (*сообщение на почту, позднее – сервис сбора ошибок*)
  + 1. Управляющие элементы

Управляющие элементы представлены в виде панели инструментов, частично дублирующих меню:

* Синхронизация с сервером (*синхронизация локальной базы данных с серверной*)
* Добавить таблицу (*добавить дерево вручную*)
* Импорт из файла… (*импорт дерева из файла*)
* Редактировать выбранное дерево
* Удалить выбранное дерево
* Переключиться на окно редактирования формул
* Получить следующий элемент выбранного дерева
* Выход
  + 1. Дизайн
    2. Таблица

Табличный список деревьев с постраничной отдачей. Колонки таблицы:

* + - * + - идентификатор (ID)
        + - наименование (Наименование)
        + - управляющие элементы (кнопки):

- редактировать описание. Открывается страница редактирования описания дерева

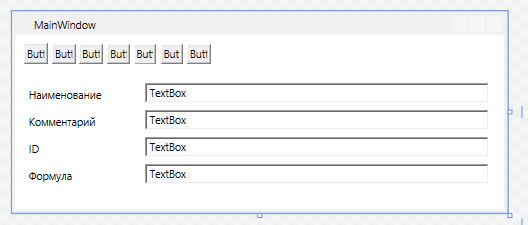
- редактировать дерево. Открывается страница редактирования дерева

- изменить формулу выбора для данного дерева. Открывается окно выбора формулы

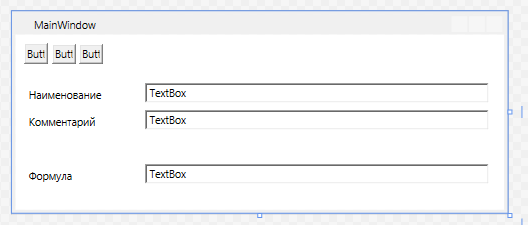
- получить следующий элемент дерева. Открывается модальное окно элемента

- удалить дерево. Открывается диалоговое окно с подтверждением удаления

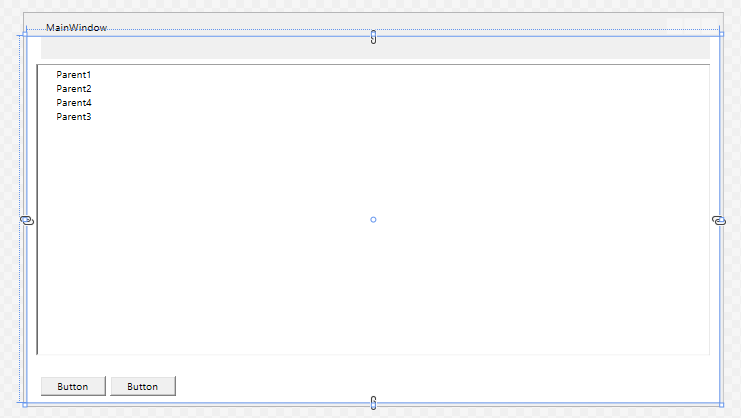
* 1. Окно редактирования описания дерева
     1. Управляющие элементы
* Кнопка «Сохранить»
* Кнопка «Копировать в новое дерево»
* Кнопка «Изменить формулу»
* Кнопка «Просмотреть историю изменений»
* Кнопка «Просмотр/Редактирование дерева»
* Кнопка «Удалить»
* Кнопка «Отменить»
  + 1. Дизайн



* + 1. Поля
* Идентификатор (readonly)
* Наименование
* Комментарий
* Формула (readonly)
  1. Окно добавления описания дерева
     1. Управляющие элементы
* Кнопка «Сохранить»
* Кнопка «Выбрать формулу»
* Кнопка «Отменить»
  + 1. Дизайн



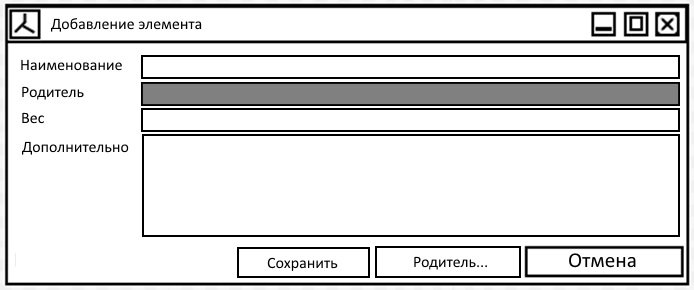
* + 1. Поля
* Наименование
* Комментарий
* Формула (readonly)
  1. Окно редактирования элементов дерева
     1. Управляющие элементы
* Кнопка «Сохранить»
* Кнопка «Отменить»
  + 1. Дизайн



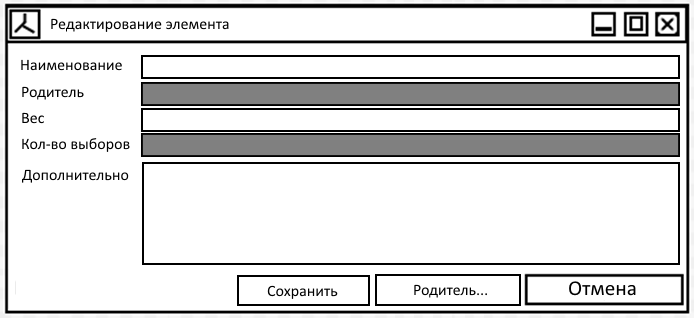
* + 1. Отображение дерева

Дерево отображается наподобие дерева файловой системы, слева от каждого элемента значок отображения/скрытия дочерних элементов, справа управляющие элементы: добавить дочерний элемент, удалить, редактировать

* 1. Окно добавления элемента дерева
     1. Управляющие элементы
* Кнопка «Сохранить»
* Кнопка «Отменить»
  + 1. Дизайн



* + 1. Поля
* Наименование
* Родитель (readonly)
* Вес
* Дополнительно (textarea, для хранения дополнительных полей)
  1. Окно редактирования элемента дерева
     1. Управляющие элементы
* Кнопка «Сохранить»
* Кнопка «Сменить родителя»
* Кнопка «Отменить»
  + 1. Дизайн



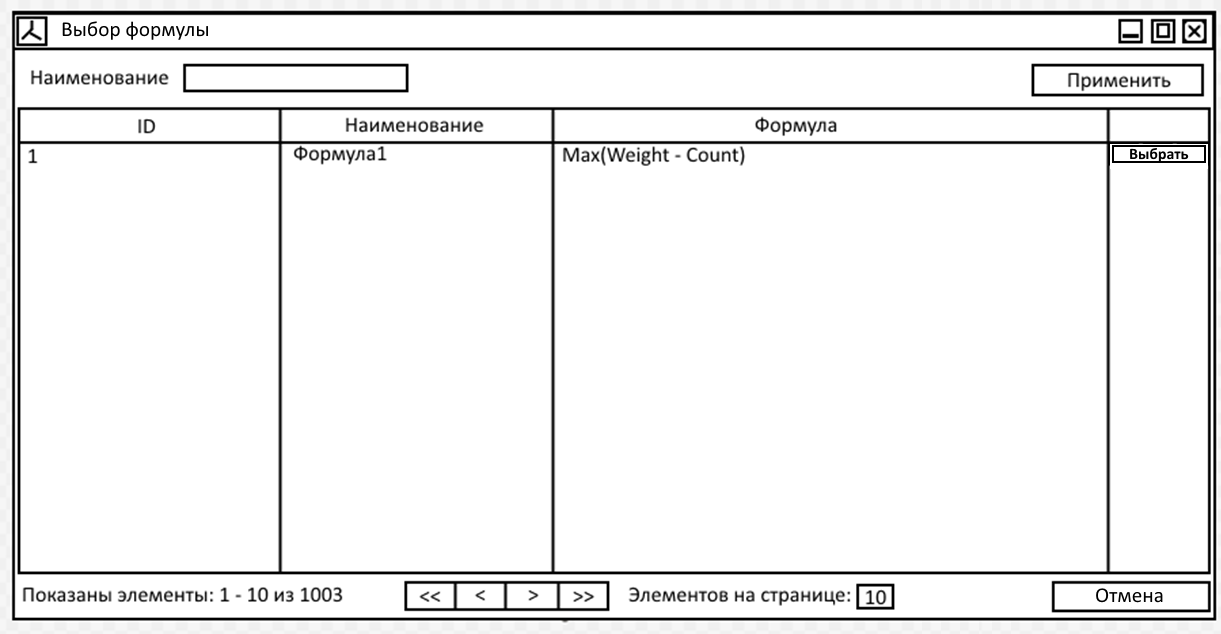
* + 1. Поля
* Наименование
* Родитель (readonly)
* Кол-во выборов (readonly)
* Вес
* Дополнительно (textarea, для хранения дополнительных полей)
  + 1. Логика

Поля «родитель» и «кол-во» выборов readonly (родителя можно изменить по кнопке «Изменить родителя»). Наименование должно быть уникальным для всех дочерних элементов родителя. В поле «Дополнительно» данные хранятся в формате json. Вес - целое положительное число (больше 0).

По кнопке «Сохранить» данные сохраняются в структуру дерева (без сохранения в БД), форма закрывается.

По кнопке «Отменить» изменения не сохраняются, форма закрывается (с подтверждением).

* 1. Окно выбора формулы
     1. Управляющие элементы
* Кнопка «Выбрать» напротив каждой формулы
* Кнопка «Применить» в блоке фильтров
* Кнопка «Отмена»
* Элементы пагинатора
  + 1. Дизайн



* + 1. Поля
* Таблица: список формул
* Фильтры:
  + Текстовое поле: «Наименование»
* Пагинатор:
  + Числовое поле: кол-во элементов на странице
    1. Логика

Форма предназначена для выбора формулы, вызывается с формы редактирования описания дерева. По нажатия кнопки «Выбрать» возвращает в вызывающую форму Id выбранной формулы.

* 1. Окно списка формул
     1. Управляющие элементы

- Кнопка «Добавить»

- Напротив каждой формулы кнопки:

- Редактировать

- Удалить

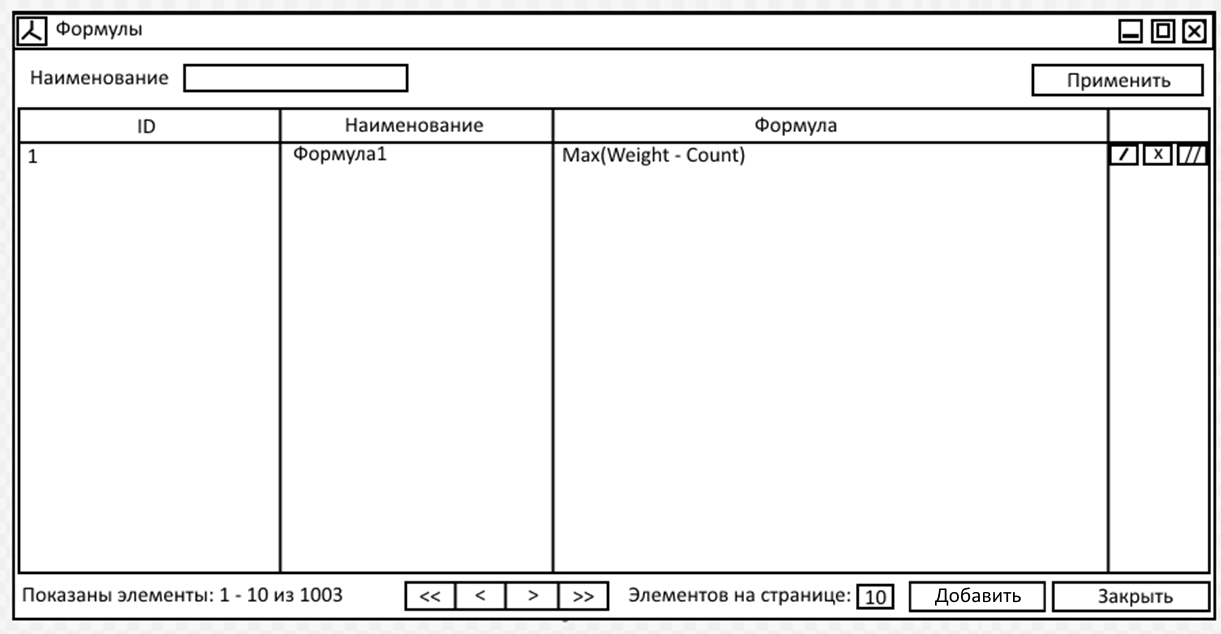
- Копировать

- Кнопка «Применить» на панели фильтров

- Элементы управления пагинатора

- Кнопка «Закрыть»

* + 1. Дизайн



* + 1. Поля
* Таблица: Список формул
* Фильтры:
  + Текстовое поле: Наименование
* Пагинатор:
  + Числовое поле: Кол-во элементов на странице
    1. Логика

Окно представляет собой список формул с блоком фильтров и пагинатором. Напротив каждой записи три кнопки – «Редактировать», «Копировать», «Удалить».

Общий элемент управления – кнопка Добавить.

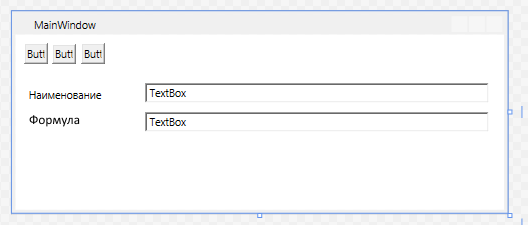
По нажатию на «Добавить» открывается страница добавления формулы.

По нажатию на «Редактировать» открывается страница редактирования выбранной формулы.

По нажатию на «Удалить» открывается окно подтверждения удаления формулы.

По нажатию на «Копировать» открывается окно добавления формулы с заполненными полями Наименование и Формула, скопированными из выбранной записи.

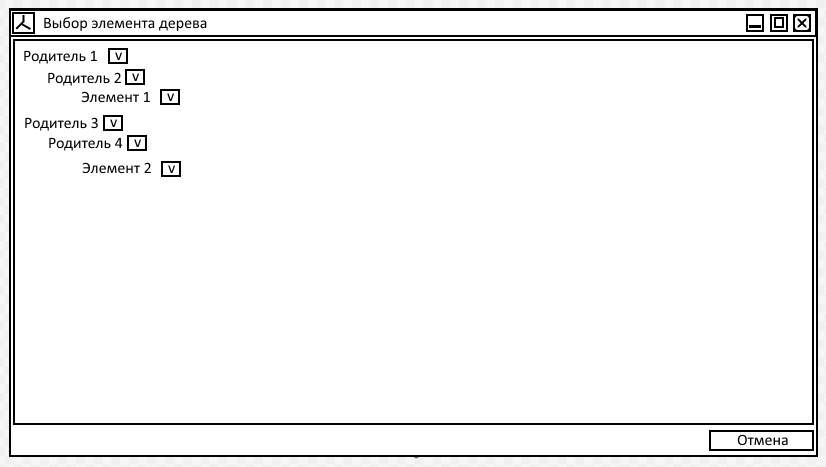
* 1. Окно редактирования формулы
     1. Управляющие элементы
* Кнопка «Сохранить»
* Кнопка «Отменить»
  + 1. Дизайн



* + 1. Поля
* Наименование
* Формула
* Флаг «Установить по умолчанию»
  + 1. Логика

Окно предназначено для редактирования формулы, по кнопке «Сохранить» формула сохраняется в базу данных, по кнопке «Отмена» форма закрывается без сохранения, по кнопке «Копировать» форма закрывается и открывается новая форма со скопированными из текущей формы данными.

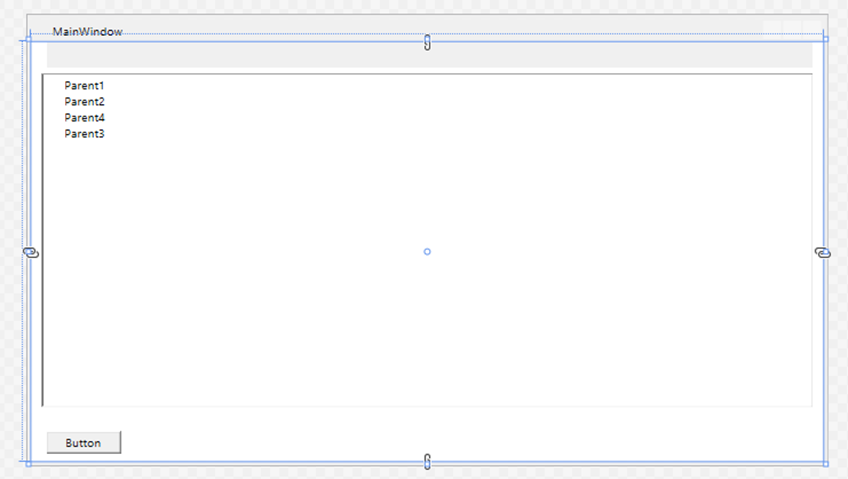
* 1. Окно выбора элемента дерева
     1. Управляющие элементы
* Кнопка «Выбрать» напротив каждого из элементов
* Кнопка «Отмена»
  + 1. Дизайн



* + 1. Поля
* Tree-компонент: список элементов дерева
  + 1. Логика

Окно содержит список элементов, кнопки «Выбрать» и «Отменить». По нажатию на кнопку «Выбрать» окно закрывается, в вызывающее окно возвращается id выбранного элемента. По нажатию на «Отменить» окно возвращается, не возвращая ничего.

* 1. Окно синхронизации
     1. Управляющие элементы
* Кнопка «Синхронизировать»
* Кнопка «Отмена»
  + 1. Дизайн



* + 1. Поля

- Таблица конфликтов синхронизации

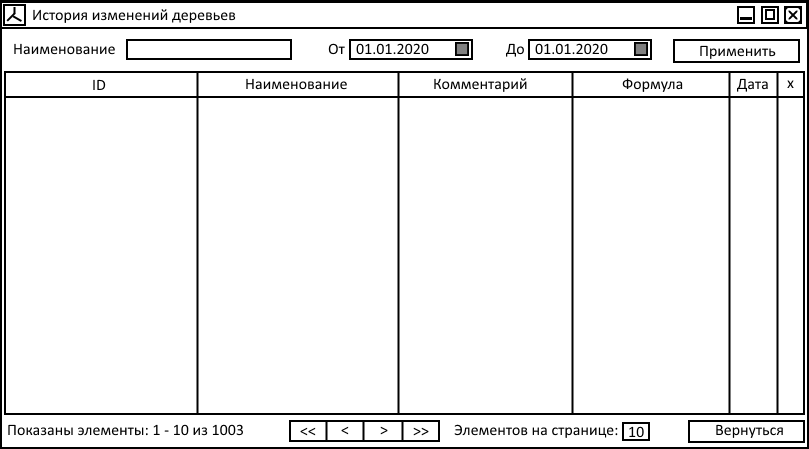
* + 1. Логика

Окно открывается только в случае наличия конфликтов синхронизации с сервером. Конфликт возникает, если на сервере есть записи младше либо равные сохраняемой по номеру версии.

По кнопке «Синхронизация» выполняется попытка синхронизации с сервером, в таблице отображаются конфликты в виде двух версий (клиентской и серверной) и галочек выбора. Кнопка доступна только если во всех строках проставлены галочки выбора. Записи, для которых имеются конфликты синхронизации, будут сохраняться только локально.

По кнопке «Отмена» окно закрывается, на главном окне отображается напоминание о наличии в системе конфликтов синхронизации.

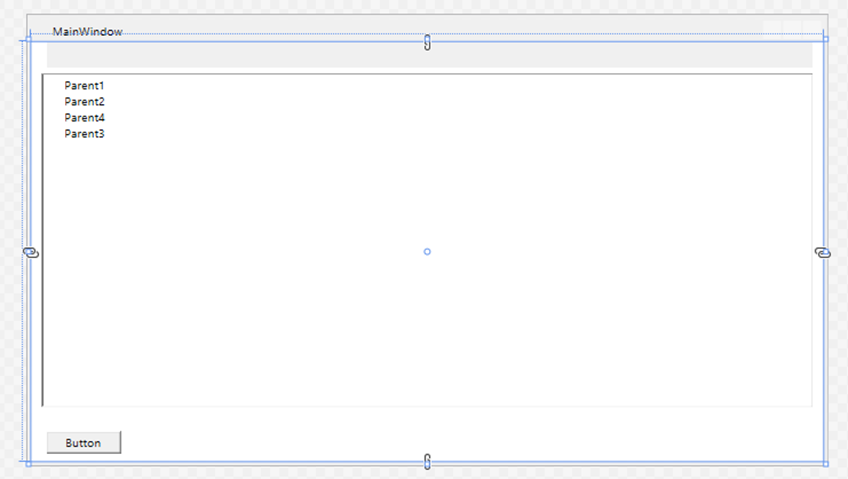
* 1. Окно истории изменений деревьев
     1. Управляющие элементы
* Кнопка «Показать»
* Кнопка «Вернуться»
  + 1. Дизайн



* + 1. Поля
* Таблица истории изменений
  + 1. Логика

По открытию страницы отображается история изменений элементов по выбранному дереву за весь период по убыванию даты, в фильтрах возможно изменить эти параметры. По кнопке «Вернуться» форма закрывается и открывается предыдущая страница.

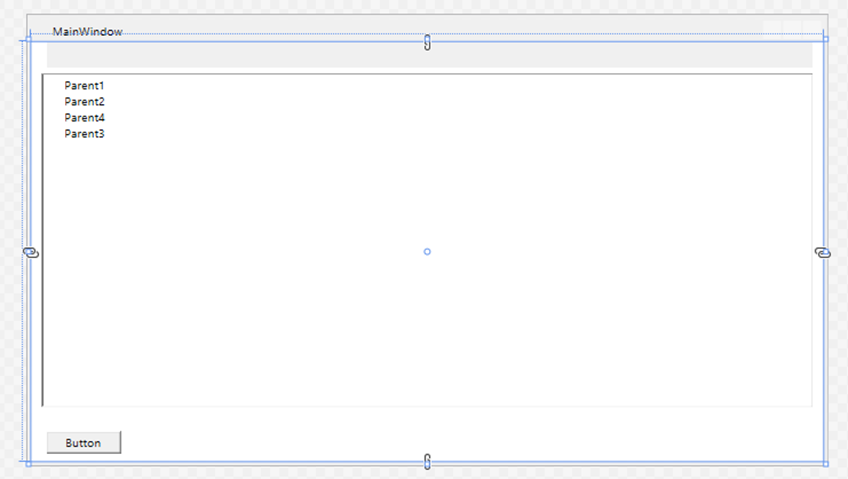
* 1. Окно истории изменений элементов деревьев
     1. Управляющие элементы
* Кнопка «Показать»
* Кнопка «Вернуться»
  + 1. Дизайн



* + 1. Поля
* Таблица истории изменений
  + 1. Логика

По открытию страницы отображается история изменений элементов по выбранному дереву за весь период по убыванию даты, в фильтрах возможно изменить эти параметры. По кнопке «Вернуться» форма закрывается и открывается предыдущая страница.

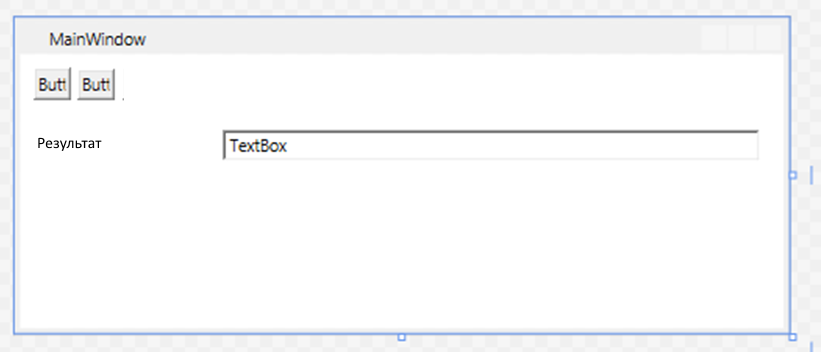
* 1. Окно истории изменений формул
     1. Управляющие элементы
* Кнопка «Показать»
* Кнопка «Вернуться»
  + 1. Дизайн



* + 1. Поля
* Таблица истории изменений
  + 1. Логика

По открытию страницы отображается история изменений формул за весь период по убыванию даты, в фильтрах возможно изменить эти параметры. По кнопке «Вернуться» форма закрывается и открывается предыдущая страница.

* 1. Окно выборки следующего элемента дерева
     1. Управляющие элементы
* Кнопка «Следующий»
* Кнопка «Закрыть»
  + 1. Дизайн



* + 1. Поля

Поле результата (readonly)

* + 1. Логика

При открытии окна делается выборка следующего элемента дерева и увеличивается счетчик кол-ва выборов для выбранного элемента. Выбранный элемент отображается с полным путем от вершины дерева.

По кнопке «Следующий» делается выборка следующего элемента, по кнопке «Вернуться» окно закрывается.

1. Web клиент
   1. Главная страница
      1. Управляющие элементы

- “Добавить” - добавить дерево. Открывается страница добавления описания дерева

- «Формулы» - список формул выбора элемента. Открывается окно редактирования списка формул

* + 1. Фильтры таблицы:

- по наименованию, фильтрация от трех символов по вхождению (ILIKE)

* + 1. Сортировка таблицы – по заголовкам таблицы (только наименование)
    2. Главная таблица

Табличный список деревьев с постраничной отдачей. Колонки таблицы:

* + - * + - идентификатор (ID)
        + - наименование (Наименование)
        + - управляющие элементы (в виде иконок с ToolTip):

- редактировать описание. Открывается страница редактирования описания дерева

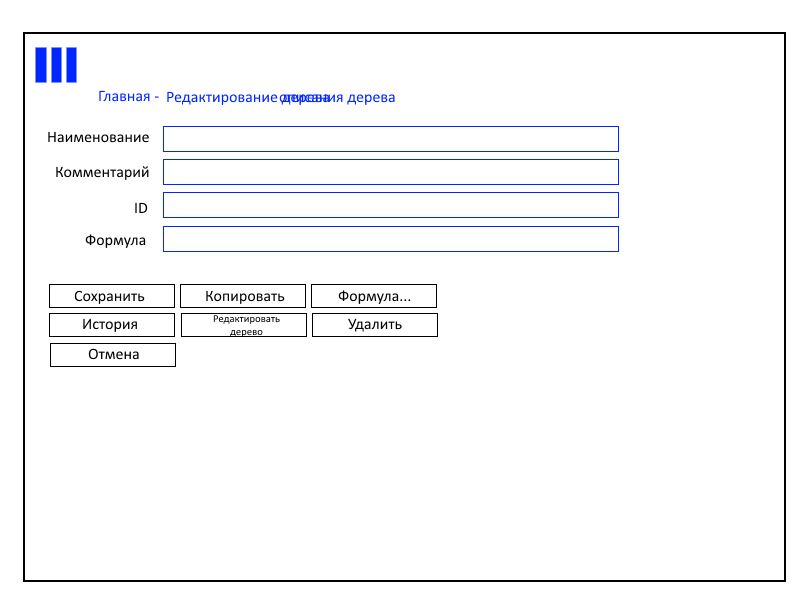
- редактировать дерево. Открывается страница редактирования дерева

- изменить формулу выбора для данного дерева. Открывается окно выбора формулы

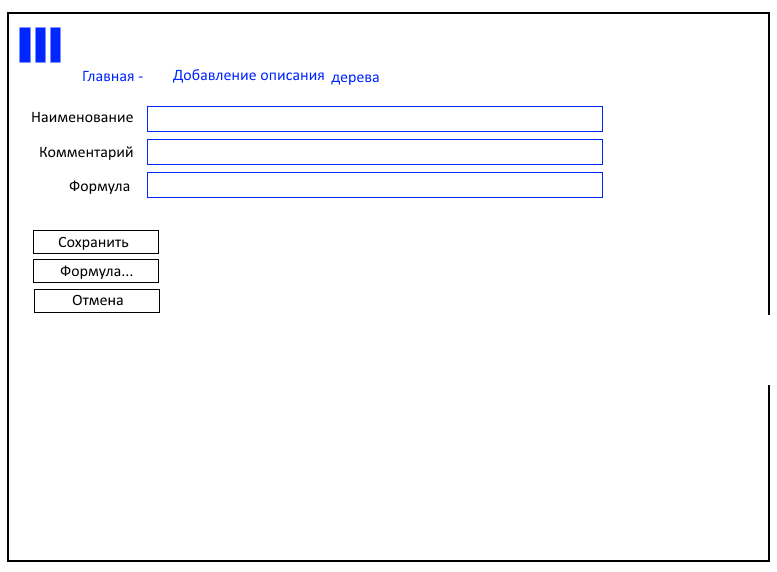
- получить следующий элемент дерева. Открывается модальное окно элемента

- удалить дерево. Открывается диалоговое окно с подтверждением удаления

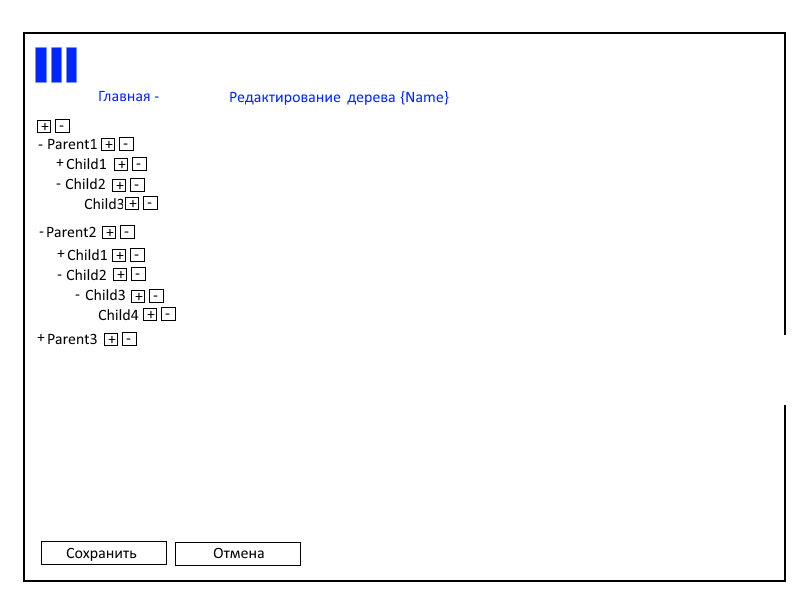
* + 1. Пагинатор главной страницы – общий элемент для всех таблиц
  1. Страница редактирования описания дерева
     1. Управляющие элементы
* Кнопка «Сохранить»
* Кнопка «Копировать в новое дерево»
* Кнопка «Изменить формулу»
* Кнопка «Просмотреть историю изменений»
* Кнопка «Просмотр/Редактирование дерева»
* Кнопка «Удалить»
* Кнопка «Отменить»
  + 1. Поля
* Идентификатор (readonly)
* Наименование
* Комментарий
* Формула (readonly)
  + 1. Меню – общий элемент
    2. Дизайн



* 1. Страница добавления описания дерева
     1. Управляющие элементы
* Кнопка «Сохранить»
* Кнопка «Выбрать формулу»
* Кнопка «Отменить»
  + 1. Поля
* Наименование
* Комментарий
* Формула (readonly)
  + 1. Меню – общий элемент
    2. Дизайн



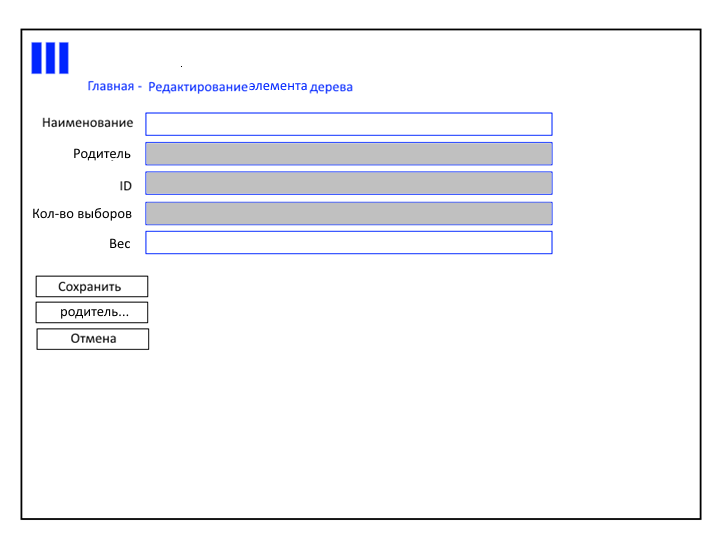
* 1. Страница редактирования дерева
     1. Управляющие элементы
* Кнопка «Сохранить»
* Кнопка «Отменить»
  + 1. Меню – общий элемент
    2. Дизайн



* + 1. Отображение дерева

Дерево отображается наподобие дерева файловой системы, слева от каждого элемента значок отображения/скрытия дочерних элементов, справа управляющие элементы: добавить дочерний элемент, удалить, редактировать

* 1. Страница редактирования элемента дерева
     1. Управляющие элементы
* Кнопка «Сохранить»
* Кнопка «Изменить родителя»
* Кнопка «Отменить»
  + 1. Меню – общий элемент
    2. Дизайн



* + 1. Поля
* Наименование
* Родитель (readonly)
* ID (readonly)
* Кол-во выборов (readonly)
* Вес
* Дополнительно
  1. Страница добавления элемента дерева
     1. Управляющие элементы
* Кнопка «Сохранить»
* Кнопка «Изменить родителя»
* Кнопка «Отменить»
  + 1. Меню – общий элемент
    2. Дизайн
    3. Поля
* Наименование
* Родитель (readonly)
* Вес
* Дополнительно
  + 1. Логика

На форме можно задать наименование, вес элемента и дополнительные поля (в формате json). Изменить родительский элемент можно по кнопке «Изменить родителя» (открывается окно выбора родительского элемента). По кнопке «Сохранить» элемент сохраняется на форме дерева, форма добавления закрывается. По кнопке «Отменить» элемент не добавляется, форма добавления закрывается. Вес элемента – целое положительное число, минимум 1.

* 1. Пагинатор

Представляет собой общий элемент для всех страниц со списками.

По изменению страницы или размера делается запрос обновления данных.

Информационные элементы:

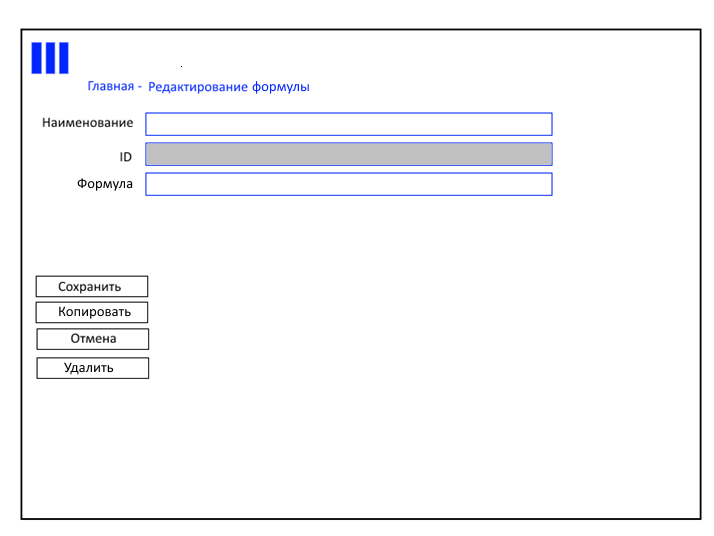
* Номер страницы (либо порядковые номера элементов от - до)
* Общее количество страниц

Управляющие элементы

* В начало (кнопка)
* В конец (кнопка)
* Предыдущая страница (кнопка)
* Следующая страница (кнопка)
* Количество на странице (выпадающий список)

Элементы расположены в одну строку

* 1. Общий дизайн (см. раздел «Дизайн» проекта)
  2. Страница редактирования формулы
     1. Управляющие элементы
* Кнопка «Сохранить»
* Кнопка «Копировать в новую формулу»
* Кнопка «Отменить»
* Кнопка «Удалить»
  + 1. Меню – общий элемент
    2. Дизайн



* + 1. Поля
* Текстовое поле: Наименование
* Текстовое поле: Формула
* Логическое значение (флаг): Сделать по умолчанию
  + 1. Логика

Поле ID – только для чтения, генерируется на сервере. Поле «Формула» - textarea (многострочное текстовое поле).

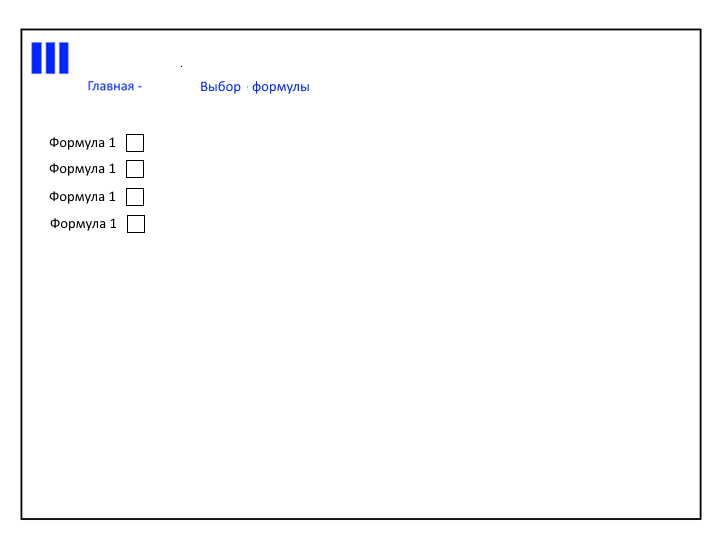
По нажатию на кнопку «Сохранить» формула сохраняется в базу, страница закрывается.

По нажатию на кнопку «Копировать» страница закрывается, открывается форма добавления формулы с копиями полей (Наименование генерируется новое как «наименование» + « копия»).

По нажатию на кнопку «Отмена» страница закрывается без сохранения, если были изменения – открывается форма подтверждения отмены.

По нажатию на кнопку «Удалить» формула удаляется из базы, страница закрывается, открывается форма подтверждения удаления. Если формула привязана к дереву, генерируется ошибка.

* 1. Страница выбора формулы
     1. Дизайн



* + 1. Поля
* Single-select список формул
  + 1. Логика

Форма предназначена для выбора формулы, вызывается с формы редактирования описания дерева. По нажатия кнопки выбора напротив выбранной формулы возвращает в вызывающую форму Id выбранной формулы.

* + 1. Управляющие элементы

Кнопки выбора формулы напротив каждой формулы

* 1. Страница редактирования списка формул
     1. Управляющие элементы

- Кнопка «Добавить»

- Напротив каждой формулы кнопки:

- Редактировать

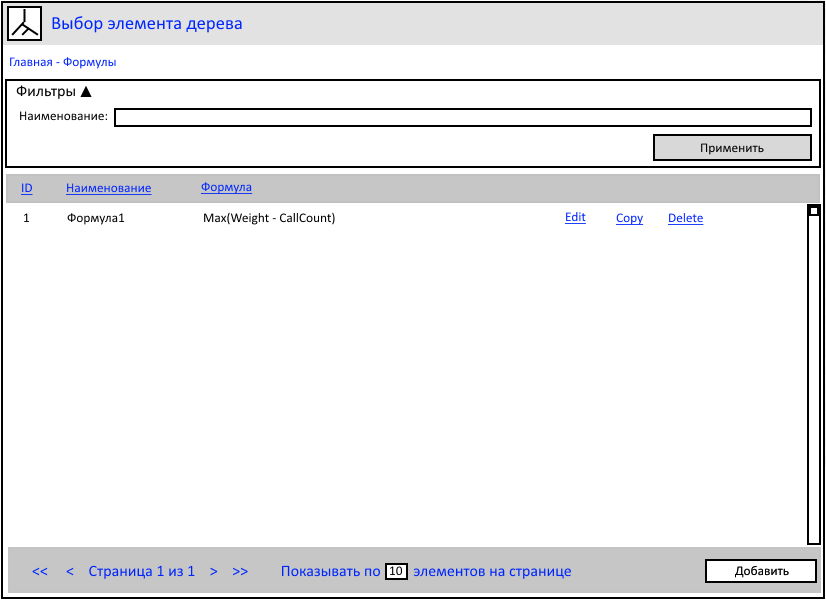
- Удалить

- Копировать

- Кнопка «Применить» в блоке фильтров

Элементы пагинатора (общий элемент)

* + 1. Поля
* Таблица: список формул
* Фильтры:
  + Текстовое поле: Наименование
* Текстовое поле: Кол-во записей на странице
  + 1. Дизайн



* + 1. Логика

Страница представляет собой список формул с блоком фильтров и пагинатором. Напротив каждой записи три кнопки – «Редактировать», «Копировать», «Удалить».

Общий элемент управления – кнопка Добавить.

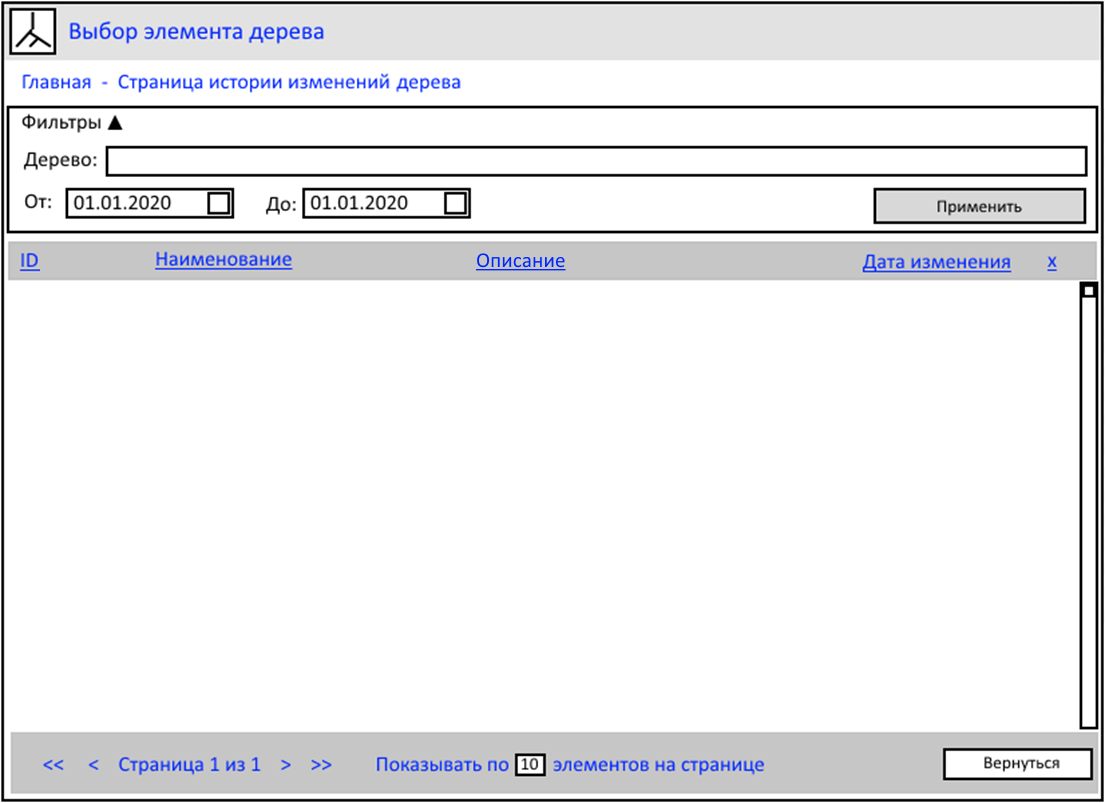
По нажатию на «Добавить» открывается страница добавления формулы.

По нажатию на «Редактировать» открывается страница редактирования выбранной формулы.

По нажатию на «Удалить» открывается окно подтверждения удаления формулы.

По нажатию на «Копировать» открывается окно добавления формулы с заполненными полями «Наименование» и «Формула», скопированными из выбранной записи.

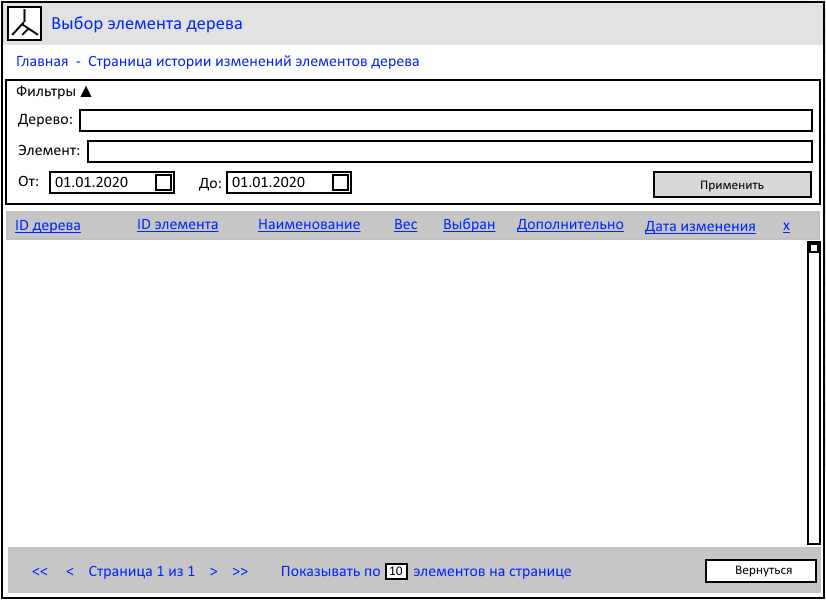
* 1. Страница истории изменений деревьев
     1. Управляющие элементы
* Кнопка «Применить фильтр»
* Кнопка «Вернуться»
* Управляющие элементы пагинатора
  + 1. Поля
       - Таблица: Список истории изменений
       - Поля фильтров:
         * Текстовое поле: Дерево
         * Поле даты: От
         * Поле даты: До
       - Числовое поле: Количества отображаемых элементов
    2. Дизайн



* + 1. Логика

По открытию страницы отображается история по выбранному дереву за весь период по убыванию даты, в фильтрах возможно изменить эти параметры. По кнопке «Вернуться» форма закрывается и открывается предыдущая страница.

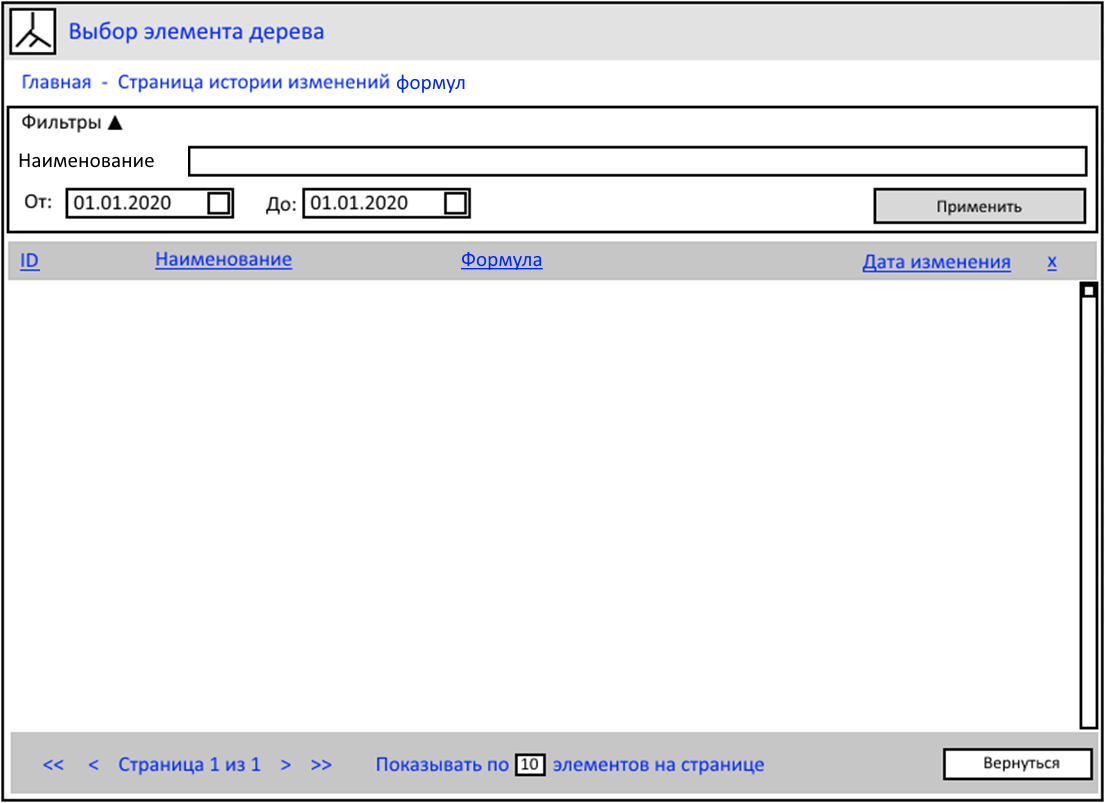
* 1. Страница истории изменений элементов деревьев
     1. Управляющие элементы
* Фильтры: Кнопка «Применить фильтр»
* Кнопка «Вернуться»
* Управляющие элементы пагинатора
  + 1. Поля
       - Таблица: Список истории изменений
       - Поля фильтров:
         * Текстовое поле: Дерево
         * Текстовое поле: Элемент
         * Поле даты: От
         * Поле даты: До
       - Пагинатор:
         * Числовое поле: кол-во элементов на странице
    2. Дизайн



* + 1. Логика

По открытию страницы отображается история изменений элементов по выбранному дереву за весь период по убыванию даты, в фильтрах возможно изменить эти параметры. По кнопке «Вернуться» форма закрывается и открывается предыдущая страница.

* 1. Страница истории изменений формул
     1. Управляющие элементы
* Кнопка «Вернуться»
* Блок фильтров: Кнопка «Применить»
* Управляющие элементы пагинатора
  + 1. Поля
       - Таблица: Список истории изменений
       - Фильтры
         * Текстовое поле: Наименование
         * Поле даты: От
         * Поле даты: До
       - Пагинатор
         * Числовое поле: кол-во элементов на странице
    2. Дизайн



* + 1. Логика

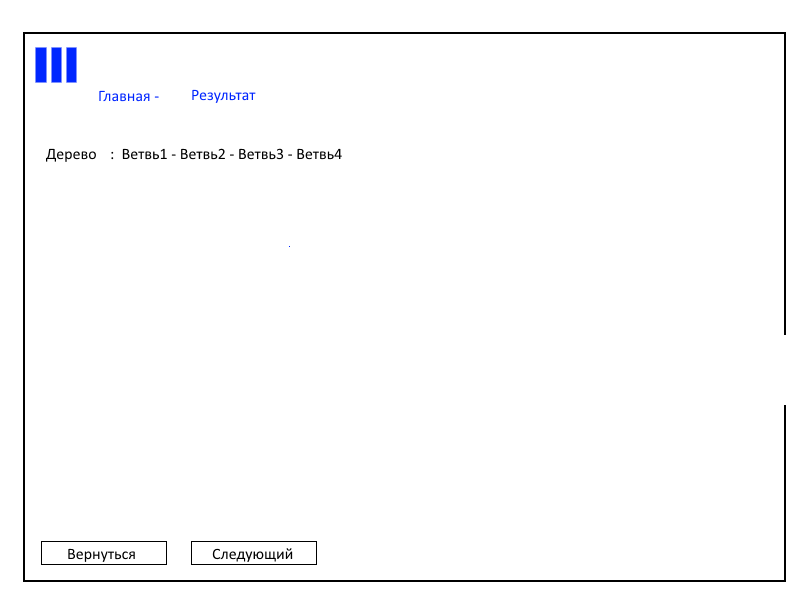
По открытию страницы отображается история изменений формул за весь период по убыванию даты, в фильтрах возможно изменить эти параметры. По кнопке «Вернуться» форма закрывается и открывается предыдущая страница.

* 1. Страница выборки следующего элемента дерева
     1. Управляющие элементы

- Кнопка «Вернуться»

- Кнопка «Следующий»

* + 1. Поля
       - Текстовое поле результата выборки элемента (полный путь от вершины дерева)
    2. Дизайн



* + 1. Логика

При открытии окна делается выборка следующего элемента дерева и увеличивается счетчик кол-ва выборов для выбранного элемента. Выбранный элемент отображается с полным путем от вершины дерева.

По кнопке «Следующий» делается выборка следующего элемента, по кнопке «Вернуться» окно закрывается.

1. Установщик
   1. Сервер Windows

Для построения установщика используется WixInstaller. Содержит следующие окна:

* + 1. Описание
    2. Лицензионное соглашение
    3. Выбор папки установки
    4. Выбор поставщика базы данных
    5. Подключение к базе данных
    6. Выбор режима работы (служба/приложение)
    7. Установка
    8. Выход
  1. Сервер Linux

Приложение поставляется в виде архива tar.gz. Конфигурируется вручную, описание настройки и лицензионное соглашение располагаются в файле readme.

* 1. Десктоп Windows

Для построения установщика используется WixInstaller. Содержит следующие окна:

* + 1. Описание
    2. Лицензионное соглашение
    3. Выбор папки установки
    4. Установка пароля базы данных (опционально)
    5. Установка параметров подключения к серверу (опционально)
    6. Установка
    7. Выход
  1. Сервер Docker

Приложение поставляется в виде контенйнера. Конфигурируется вручную, описание настройки и лицензионное соглашение располагаются в файле readme.

1. Тестирование
   1. Модульное тестирование
      1. Сервер
         * Тесты репозиториев
           + Метод Add

Проверяется работа метода Add репозитория для любой из моделей. Последовательность теста:

* Создание окружения
* Добавление сущности
* Получение сущности по Id
* Проверка на not null
* Проверка на идентичность полей базовой и сохраненной моделей
  + - * + Метод Get

Проверяется работа метода Get репозитория для любой из моделей. Последовательность теста:

* Создание окружения
* Добавление нескольких сущностей
* Получение сущностей по фильтру
* Проверка на соответствие полученных сущностей фильтру
  + - * + Метод GetItem

Проверяется работа метода GetItem репозитория для любой из моделей. Последовательность теста:

* Создание окружения
* Добавление сущности
* Получение сущности по Id
* Проверка на Not null
  + - * + Метод Update

Проверяется работа метода Update репозитория для любой из моделей. Последовательность теста:

* Создание окружения
* Добавление сущности
* Получение сущности по Id
* Проверка на not null
* Проверка на идентичность полей базовой и сохраненной моделей
* Изменение сущности
* Получение сущности по Id
* Проверка на not null
* Проверка на идентичность ожидаемых (измененных) полей и полей сохраненной модели
  + - * + Метод Delete

Проверяется работа метода Delete репозитория для любой из моделей. Последовательность теста:

* Создание окружения
* Добавление сущности
* Получение сущности по Id
* Проверка на not null
* Проверка на идентичность полей базовой и сохраненной моделей
* Удаление сущности
* Получение сущности по Id
* Проверка на null
  + - * Тесты API
        + Дерево

Метод получения списка деревьев

Проверяется работа метода получения списка деревьев согласно переданным фильтрам. Последовательность теста:

* Создание окружения
* Добавление нескольких деревьев
* Получение деревьев по фильтру через API
* Проверка на соответствие полученных деревьев фильтру

Метод получения дерева по Id

Проверяется работа метода получения дерева по идентификатору. Последовательность теста:

* Создание окружения
* Добавление дерева
* Получение дерева по Id через API
* Проверка на not null

Метод получения всех элементов дерева

Проверяется работа метода получения всех элементов дерева по идентификатору дерева. Последовательность теста:

* Создание окружения
* Добавление дерева
* Добавление нескольких элементов дерева
* Получение дерева по Id через API
* Проверка на not null
* Получение всех элементов дерева по Id через API
* Проверка на соответствие элементов (по количеству)

Метод добавления дерева

Проверяется работа метода добавления дерева. Последовательность теста:

* Создание окружения
* Добавление дерева с несколькими элементами через метод API
* Получение дерева и его элементов через репозиторий
* Проверка на соответствие дерева и его элементов базовому

Метод изменения дерева

Проверяется работа метода изменения дерева. Последовательность теста:

* Создание окружения
* Добавление дерева с несколькими элементами через метод API
* Получение дерева и его элементов через репозиторий
* Проверка на соответствие дерева и его элементов базовому
* Изменение дерева (в т.ч. элементов)
* Вызов метода изменения дерева через API
* Получение дерева и его элементов через репозиторий
* Проверка на соответствие дерева и его элементов измененному

Метод изменения элементов дерева

Проверяется работа метода изменения элементов дерева. Последовательность теста:

* Создание окружения
* Добавление дерева с несколькими элементами через метод API
* Получение дерева и его элементов через репозиторий
* Проверка на соответствие дерева и его элементов базовому
* Изменение элементов дерева (перенос в другую ветвь, переименование, добавление нового, удаление)
* Вызов метода изменения элемента дерева через API
* Получение элементов дерева через репозиторий
* Проверка что элементы были изменены

Метод удаления дерева

Проверяется работа метода удаления дерева. Последовательность теста:

* Создание окружения
* Добавление дерева с несколькими элементами через метод API
* Получение дерева и его элементов через репозиторий
* Проверка на соответствие дерева и его элементов базовому
* Вызов метода удаления дерева через API
* Получение дерева через репозиторий
* Проверка на null

Методы получения списка изменений

Проверяется работа метода получения списка изменений. Последовательность теста:

* Создание окружения
* Добавление дерева с несколькими элементами через репозиторий
* Получение дерева, его элементов и списка изменений дерева и элементов через API
* Проверка на соответствие дерева и его элементов базовому, а также соответствие изменений произведенным изменениям
* Изменение элементов дерева (перенос в другую ветвь, переименование, добавление нового, удаление)
* Получение дерева, его элементов и списка изменений дерева и элементов через API
* Проверка на соответствие дерева и его элементов базовому, а также соответствие изменений произведенным изменениям
* Вызов метода удаления дерева через API
* Получение дерева, его элементов и списка изменений дерева и элементов через API
* Проверка на соответствие дерева и его элементов базовому, а также соответствие изменений произведенным изменениям
  + - * + Формула

Метод получения списка формул

Проверяется работа метода получения списка формул согласно переданным фильтрам. Последовательность теста:

* Создание окружения
* Добавление нескольких формул
* Получение формул по фильтру через API
* Проверка на соответствие полученных формул фильтру

Метод получения формулы по id

Проверяется работа метода получения формулы. Последовательность теста:

* Создание окружения
* Добавление формулы через репозиторий
* Получение формулы через API
* Проверка на not null

Метод добавления формулы

Проверяется работа метода добавления формулы. Последовательность теста:

* Создание окружения
* Добавление формулы через метод API
* Получение формулы через репозиторий
* Проверка на соответствие формулы базовой

Метод изменения формулы

Проверяется работа метода изменения формулы. Последовательность теста:

* Создание окружения
* Добавление формулы через метод API
* Получение формулы через репозиторий
* Проверка на соответствие формулы базовой
* Изменение формулы
* Вызов метода изменения формулы через API
* Получение формулы через репозиторий
* Проверка на соответствие формулы измененной

Метод удаления формулы

Проверяется работа метода удаления формулы. Последовательность теста:

* Создание окружения
* Добавление формулы через метод API
* Получение формулы через репозиторий
* Проверка на соответствие формулы базовой
* Вызов метода удаления формулы через API
* Получение формулы через репозиторий
* Проверка на null

Методы получения списка изменений

Проверяется работа метода получения списка изменений. Последовательность теста:

* Создание окружения
* Добавление формулы через репозиторий
* Получение формулы и списка изменений формулы через API
* Проверка на соответствие формулы базовому, а также соответствие изменений произведенным изменениям
* Изменение формулы
* Получение формулы и списка изменений формулы через API
* Проверка на соответствие формулы базовой, а также соответствие изменений произведенным изменениям
* Вызов метода удаления формулы через API
* Получение формулы и списка изменений формулы через API
* Проверка на null, а также соответствие изменений произведенным изменениям
  + - * + Синхронизация

Методы синхронизации

Проверяется работа методов синхронизации. Последовательность теста:

* Создание окружения
* Добавление дерева с несколькими элементами через репозиторий
* Получение дерева и его элементов
* Проверка на соответствие дерева и его элементов базовому
* Получение дерева через метод сихронизации SyncInit
* Проверка на соответствие дерева и его элементов базовому
* Изменение нескольких элементов дерева с изменением VersionId
* Изменение нескольких элементов дерева без изменения VersionId и с повышением версии
* Сохранение дерева через метод Sync
* Проверка ответа: записи с измененным VersionId упали в конфликт, остальные нет
  + - * + Библиотека расчета по формуле

Метод расчета

Проверяется работа метода расчета. Последовательность теста:

* Создание окружения
* Отправка на расчет нескольких формул с однозначным результатом
* Проверка правильности расчета
  + 1. Клиент
    2. Web-клиент
  1. Интеграционное тестирование
     1. Клиент
        + Сценарий 1
* Создание окружения (деплой БД, запуск сервера, клиента)
* Добавление формулы (однозначный результат, без рандома)
* Добавление дерева и элементов с привязкой к формуле
* Выборка нескольких значений, сравнение с контрольными
  + - * Сценарий 2
* Создание окружения (деплой БД, запуск сервера, клиента)
* Добавление формулы (с рандомом)
* Добавление дерева и элементов с привязкой к формуле
* Выборка нескольких значений (статистически значимое количество)
* Проверка равномерного распределения (с погрешностью)
  + - * Сценарий 3
* Создание окружения (деплой БД, запуск сервера, клиента)
* Добавление нерабочей формулы
* Добавление дерева и элементов с привязкой к формуле
* Выборка значения
* Проверка правильного исключения
  + - * Сценарий 4
* Создание окружения (деплой БД, запуск сервера, клиента)
* Добавление формулы
* Добавление дерева и элементов с привязкой к формуле
* Попытка удаления формулы
* Проверка правильного исключения
  + - * Сценарий 5
* Создание окружения (деплой БД, запуск сервера, клиента)
* Добавление формулы
* Добавление дерева и элементов с привязкой к формуле
* Отключение сервера
* Изменение дерева и элементов на клиенте и сервере (через базу данных либо через сервер с другим адресом)
* Включение сервера
* Ожидание предложения синхронизации
* Синхронизация
* Проверка правильности данных на сервере
  + 1. Web-клиент.
       - Сценарий 1
* Создание окружения (деплой БД, запуск сервера, клиента)
* Добавление формулы (однозначный результат, без рандома)
* Добавление дерева и элементов с привязкой к формуле
* Выборка нескольких значений, сравнение с контрольными
  + - * Сценарий 2
* Создание окружения (деплой БД, запуск сервера, клиента)
* Добавление формулы (с рандомом)
* Добавление дерева и элементов с привязкой к формуле
* Выборка нескольких значений (статистически значимое количество)
* Проверка равномерного распределения (с погрешностью)
  + - * Сценарий 3
* Создание окружения (деплой БД, запуск сервера, клиента)
* Добавление нерабочей формулы
* Добавление дерева и элементов с привязкой к формуле
* Выборка значения по формуле
* Проверка правильного исключения
  + - * Сценарий 4
* Создание окружения (деплой БД, запуск сервера, клиента)
* Добавление формулы
* Добавление дерева и элементов с привязкой к формуле
* Попытка удаления формулы
* Проверка правильного исключения
  1. Ручное тестирование

Тесты из интеграционного тестирования

* 1. Регрессионное тестирование

Тесты из интеграционного тестирования