ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ И ЕСТЕСТВЕННЫХ НАУК

КАФЕДРА ПРИКЛАДНОЙ ИНФОРМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ИНФОРМАЦИОННОГО СОПРОВОЖДЕНИЯ РЕТРОСПЕКТИВНЫХ ДАННЫХ (НА ПРИМЕРЕ ГРУППЫ КОМПАНИЙ «ЭФКО»)

Выпускная квалификационная работа
обучающегося по направлению подготовки 09.03.03 Прикладная информатика
очной формы обучения, группы 07001404
Моисеева Михаила Андреевича

Научный руководитель
к.т.н.
Гахова Н.Н.

БЕЛГОРОД 2018

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Менее ста лет назад судьба человека во многом зависела от природных факторов, а сельхозпроизводители подстраивались под диктуемые природой законы. Современные технологии позволяют повысить эффективность сельского хозяйства за счет улучшения качества посевных материалов и генетических характеристик сельскохозяйственных растений. Это приводит к необходимости учитывать различные качественные и количественные характеристики. В результате в селекционном отделе необходимо хранить информацию о каждом виде растений, например сои, о ее мутациях, о том, когда и кем было сделано секвенирование и т.п., а также необходимо выполнять быстрый ее поиск, требуемую обработку и обновление. Эти данные занимают большой объем, а процедуры их хранения, изменения и поиска являются трудоемкими и рутинными. В настоящее время применение компьютерных технологий позволяет провести автоматизацию практически всех сфер деятельности человека. При этом упрощаются все перечисленные процедуры взаимодействия с данными.

Таким образом, проблема структурированного хранения генетических и фенотипических данных растений является актуальной. Чтобы решить данную проблему была выбрана тема по проектированию и разработке информационного сопровождения ретроспективных данных на примере группы компаний «ЭФКО». Необходимо автоматизировать процесс внесения, извлечения и хранения данных, снизить время поиска нужной информации, обеспечить удобный доступ к хранящемся данным

Объектом исследования в данной работе является селекционный отдел группы компаний «ЭФКО».

Предмет исследования – информационное сопровождение процессов хранения и обработки генетических данных растений.

Цель выпускной квалификационной работы заключается в повышении эффективности работы отдела за счет сокращения временных затрат на

выполнение операций хранения и обработки данных путем использования информационного сопровождения.

Исходя из поставленной цели, необходимо спроектировать и разработать информационное сопровождение ретроспективной информации в виде количественных и качественных показателей растений.

Для выполнения поставленной цели необходимо выполнить следующие задачи:

- исследовать предметную область;
- сформулировать основные требования к разрабатываемому информационному сопровождению;
- провести анализ и обоснование выбора инструментальных средств;
- спроектировать модель базы данных;
- разработать информационное сопровождение ретроспективной информации;
- протестировать разработанное приложение;
- выполнить оценку временных затрат.

Пояснительная записка к выпускной квалификационной работе структурно состоит из: введения, трех разделов, заключения, списка использованных источников и приложения.

Во введении рассмотрена актуальность работы, выбран предмет и объект исследования, поставлена цель, а также определены задачи и описана структура выпускной квалификационной работы.

В первом разделе приводится характеристика предметной области группы компаний «ЭФКО», описана структура селекционного отдела, приведена характеристика данных, используемых в информационном обеспечении, разрабатывается и декомпозируется модель «Как есть» и определены недостатки. Также были сформулированы основные требования к разрабатываемому информационному сопровождению и проведен анализ существующих разработок.

Во втором разделе рассматривается информационное обеспечение, используемое при разработке информационного сопровождения, а также производится структурно-функциональный анализ деятельности селекционного отдела «Как должно быть». Разработана модель базы данных и описана ее структура.

В третьем разделе представлен процесс разработки серверной и клиентской частей информационного сопровождения, описан контрольный пример использования разработанного приложения, также выполнена оценка эффективности проекта.

В заключении приведены основные результаты, полученные в ходе выполнения ВКР, сделаны выводы и даны рекомендации.

В приложении представлены материалы, характеризующие ход выполнения выпускной квалификационной работы. Также приведены рисунки и программный код разработанного информационного сопровождения.

Выпускная квалификационная работа содержит 64 страницы, 49 рисунков, 3 таблицы, 41 литературный источник и 3 приложений.

1 Аналитическая часть

1.1 Технико-экономическая характеристика предметной области

1.1.1 Характеристика группы компаний «ЭФКО»

«ЭФКО» – российская компания, управляющая несколькими предприятиями масложировой промышленности, входит в тройку крупнейших компаний российского агропромышленного комплекса. Головной офис расположен в Воронеже, представительства – в Москве, Алексеевке и Тамани, производственные площадки – в Белгородской, Воронежской, Московской и Свердловской областях, в Краснодарском крае и Казахстане. Основана в 1994 году на базе приватизированного комбината по выпуску парфюмерно-косметических изделий, синтетических душистых веществ и эфирных масел – производственного объединения «Эфирное» из Алексеевки. Двумя годами позже производство было переориентировано на выпуск подсолнечного масла, бутилированная продукция получила торговую марку «Слобода» [12].

Группа компаний (ГК) «ЭФКО» является крупнейшим российским вертикально-интегрированным производителем жиров специального назначения, используемых в кондитерской, хлебопекарной и других отраслях пищевой промышленности. Компания также является ведущим производителем майонеза, растительного масла и кетчупа в России, выпуская эту продукцию под такими широко известными брендами, как «Слобода» и Altero. ГК «ЭФКО» занимает лидирующие позиции в своей отрасли и зарекомендовала себя как надежный партнер, при этом компания продолжает успешное развитие во всех сегментах и видит широкие возможности для дальнейшей экспансии на другие рынки. Все это достигнуто благодаря выбранной стратегии компании: усиление лидирующих позиций за счет разработки новых, более совершенных видов жиров и повышения степени лояльности клиентов, в том числе благодаря расширению спектра оказываемых услуг.

Компания занимает ведущие позиции в России по объему продаж майонеза. Увеличение степени влияния на данном рынке стало возможным с помощью:

- запуска новых продуктов, в том числе удовлетворяющих потребности покупателей в правильном питании, и проведения широких маркетинговых кампаний;
- модернизации производственных мощностей и оптимизации затрат;
- обновление имеющихся и строительство новых производственных объектов для увеличения выпуска и ассортимента продукции и улучшения операционной эффективности производства.

Основные черты компании – это инновационный подход к производству, высокое качество продукции и бережное отношение к окружающей среде. Инновационная деятельность ГК «ЭФКО» включает в себя обширную программу по разработке новых продуктов и технологий производства, внедрению и совершенствованию системы управления качеством, а также автоматизации бизнес-процессов [12]. Инновации являются частью философии «ЭФКО», и во многом именно собственные инновационные проекты позволили Компании занять ведущее положение в различных направлениях бизнеса: в области переработки сырья, производства высокотехнологичных жиров и брендовой продукции. В каждом дивизионе Компании созданы исследовательские лаборатории и центры прикладных исследований: в брендовом дивизионе – служба по разработке новых продуктов, в масложировом дивизионе – Центр Прикладных Исследований и Цех пилотных установок, которые занимаются разработкой и тестированием жировых продуктов для линий здорового питания и инновационных рецептур с учетом потребностей производителей пищевой отрасли. До конца года Группа планирует открыть Центр прикладных исследований масложировой продукции в Казахстане [12].

Структура ГК представлена на рисунке 1.1.

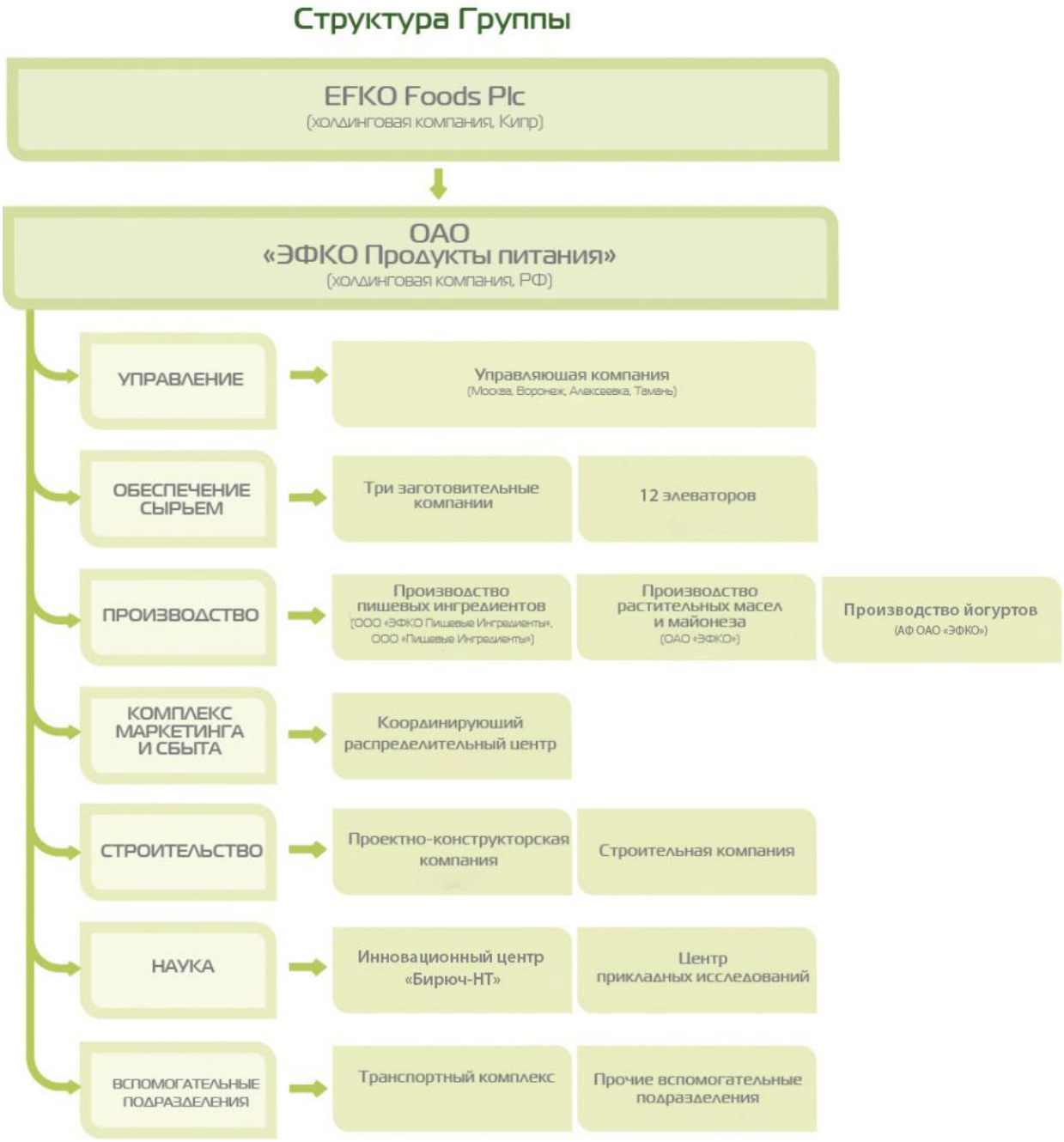


Рисунок 1.1 – Структура ГК «ЭФКО»

ГК «ЭФКО» уделяет особое внимание сотрудничеству с ведущими научно-исследовательскими институтами, а также ведет активную работу с профильными вузами, акселераторами стартап-проектов и поддерживает конкурсы научных идей. Компания всегда ищет новые драйверы роста и заинтересована в поддержке талантливых специалистов с активной жизненной позицией, которым готова предоставить уникальные возможности для самореализации в качестве разработчиков новых технологий и инновационных

проектов и будущих руководителей бизнесов на основе этих проектов. В результате в 2013 году создан Инновационный Центр «Бирюч» – российская инновационная компания, являющаяся результатом равноправного партнерства акционеров ГК «ЭФКО» и коллектива ведущих отечественных ученых. Инновационный Центр «Бирюч – НТ» расположен в Белгородской области, село Малобыково Красногвардейского района, возле лимана реки Тихая сосна.

Фокус исследований R&D (научно-исследовательские и опытно-конструкторские работы) центра сосредоточен на инновационном приборостроении. В рамках ИЦ «Бирюч» сформирован уникальный коллектив специалистов – математиков, физиков, химиков, механиков, конструкторов, электронщиков и программистов [11].

Среди ключевых проектов наукоемкого приборостроения, созданных коллективом инновационного центра «Бирюч», стали разработки:

– погружного георадара для горизонтальной проводки скважин;

– дифференциальных спектрометров ионной подвижности;

– обладающих рекордной разрешающей способностью;

– промышленного дуального томографа для обнаружения алмазов в кимберлитовой руде;

– спектрометров динамического рассеяния, позволяющих определять размер и форму наночастиц.

В сфере научных исследований ЗАО «ИЦ Бирюч» выполняет исследования по проектам приборостроения:

– спектрометр ионной подвижности: разработка новых типов приборов определения малых количеств веществ в газовых и жидкостных средах сложного состава на основе спектроскопии ионной подвижности и создание приборов с рекордными параметрами;

– подземный георадар: разработка новых методов в области навигационного каротажа (контрастного расчленения разреза грунта или стратификации, в плоскости, перпендикулярной бурению, и определения

расстояния до границ продуктивного пласта в процессе бурения) для обеспечения точной и достоверной маршрутизации горизонтальных скважин;

– промышленный дуальный томограф: рентгеновский томограф, позволяющий проводить исследование внутренней структуры самых разных объектов;

– спектрометр динамического рассеяния: создание приборов определения размеров и формы наночастиц на основе динамического рассеяния;

– аппарат для размораживания и подогрева криоконсервированных продуктов крови, аппарат для размораживания и нагрева плазмы;

– проект робота «Полоз»: создание прототипа змеевидного робота для решения задач поиска и простых неотложных действий в зонах стихийных бедствий и техногенных катастроф.

На рисунке 1.2 представлены основные направления исследований и разработок Инновационного центра «Бирюч-НТ».

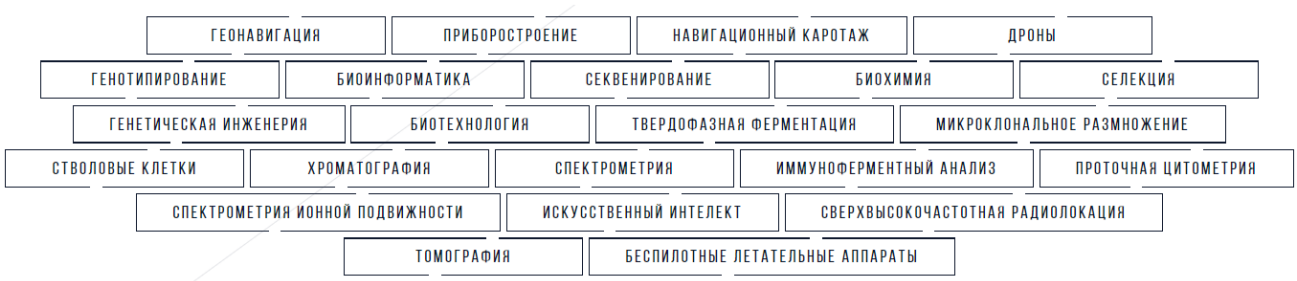


Рисунок 1.2 – Направления исследований и разработок

Партнерские отношения акционеров ГК «ЭФКО» и ученых ИЦ «Бирюч» основаны на корпоративной культуре, в основе которой лежат заинтересованность в достижении результатов, атмосфера комфортной творческой деятельности, четкое разграничение зон ответственности, взаимное уважение и ответственность, осознание значимости всех этапов проекта от идеи и НИОКР до вывода продукта на рынок.

1.1.2 Характеристика и структура селекционного отдела

Селекционный отдел относится к проекту геномики и биотехнологий. Находится отдел в Инновационном центре «Бирюч – НТ». Основные направления данного проекта – биотехнология ферментов, биотехнология белка, геномика и селекция. Селекционно-генетические проекты Инновационного центра помогут выйти отечественному сельскому хозяйству на новый уровень. Для этого используются методы геномной селекции для повышения продуктивности молочного животноводства и создания высокопродуктивных сортов сои, адаптированных к различным климатическим условиям. Для ускорения селекционного процесса внедряются технологии ЭКО и трансфер эмбрионов, а также микроклональное размножение растений [11]. На рисунке 1.3 представлена структура сотрудников отдела.

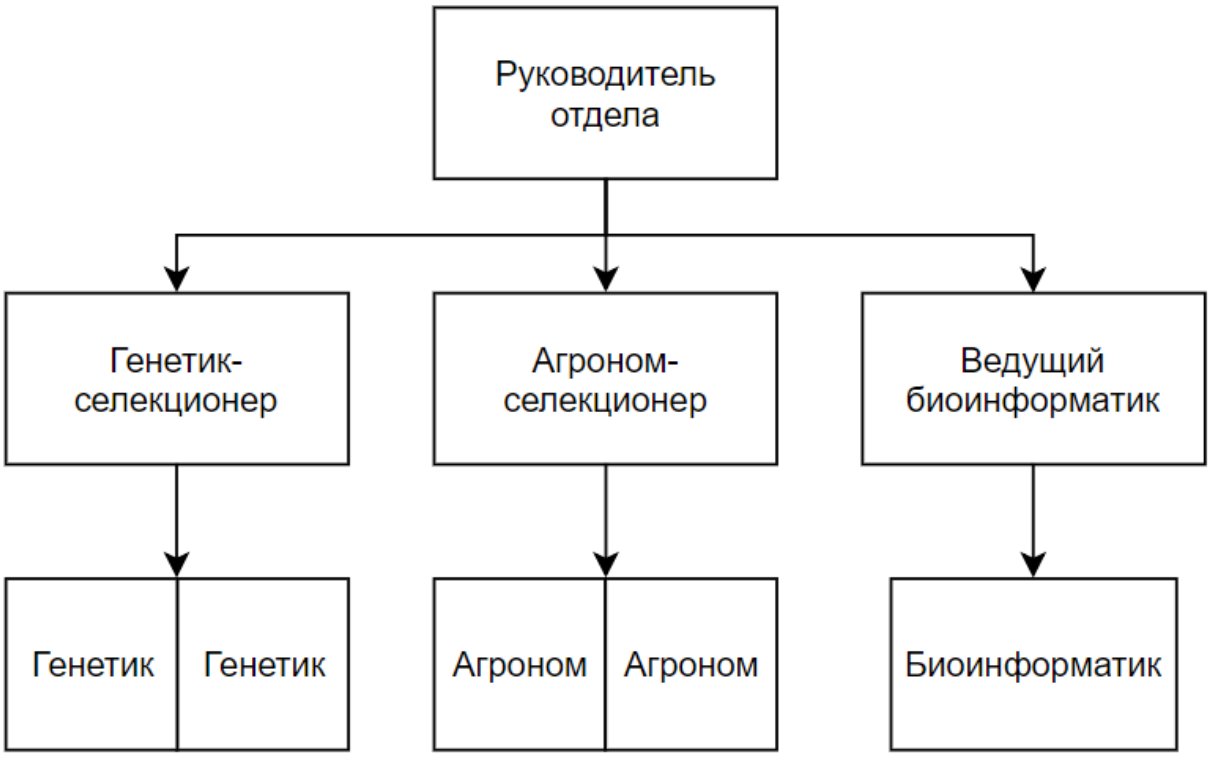


Рисунок 1.3 – Структура сотрудников отдела

Должностные обязанности ведущего биоинформатика:

– осуществлять работу с базами данных биологической информации;

– осуществлять обработку больших объемов информации (последовательности геномов, генов, фрагментов ДНК);

– заниматься расчетом структуры и функций биологических молекул (аннотацией генов, регуляторных элементов и других структур);

– заниматься множественным выравниванием последовательностей и поиском гомологических генов с помощью прикладных компьютерных программ;

– заниматься систематизацией биологической информации и разработкой алгоритмов работы с целью создания собственных баз данных;

– заниматься составлением программы скрещивания растений с известными признаками и генами совместно с селекционером для получения новых сортов сои с заранее заданными признаками;

– осуществлять разработку программ для анализа данных о структуре геномов, а также для автоматического составления генетических карт скрещивания;

– заниматься поиском, анализом, идентификацией генетических маркеров на основании данных о строении геномов;

– заниматься разработкой и реализацией сложных алгоритмов проектирования геномов, а также созданием удобных инструментов для их использования;

– систематически изучать отечественные и зарубежные научно-генетические достижения и передовой опыт в области генной инженерии и биоинформатики;

– участвовать в совещаниях и комитетах в рамках своей компетенции;

– осуществлять написание статей, подготовку докладов по темам, актуальным для компании;

– своевременно предоставлять информацию в системе электронного документооборота, по соответствующим формам;

– формировать КПГ работ на месяц и долгосрочный период;

– подготавливать отчетную документацию о выполненных работах для предоставления на Комитеты, рассмотрение руководителю проекта, другим сотрудником ООО «ИЦ «Бирюч-НТ» и других подразделений компании по согласованию с руководителем проекта;

– соблюдать требования инструкций по охране труда, норм и правил промышленной и пожарной безопасности;

– проходить обучение по охране труда и проверки знаний в комиссии предприятия не реже 1 раза в 3 года;

– обеспечивать конфиденциальность информации, сохранность коммерческой тайны. Последнее включает в себя весь комплекс обязанностей, связанных с информацией ограниченного доступа;

– обязан знать структуру предприятия, направления деятельности ООО «ИЦ «Бирюч-НТ», в том числе финансовое и экономическое, содержание и методику заполнения личной карточки, а также существующие на предприятии механизмы оплаты труда и дополнительной мотивации работников. Для подтверждения квалификации обязан проходить аттестацию по вышеуказанным предметам, а также по дисциплинам, отраженным в личной карточке в качестве необходимых для значимой должности.

Полная должностная инструкция приведена в приложении А.

1.2 Характеристика ретроспективных данных, используемых в информационном обеспечении

Селекция – наука о создании новых и улучшении существующих сортов растений. В основе селекции лежат такие методы, как гибридизация и отбор. Теоретической основой селекции является генетика. В селекции и генетики

используются такие понятия как генотип, фенотип, хромосомы и другие. В этой работе, в том числе и при разработке информационного обеспечения, используются эти понятия. Ниже представлены определения этих понятий [9].

Генотип – это совокупность всех генов организма, являющихся его наследственной основой. Ген – структурная и функциональная единица наследственности живых организмов.

Фенотип – совокупность всех признаков и свойств организма, которые выявляются в процессе индивидуального развития в данных условиях и являются результатом взаимодействия генотипа с комплексом факторов внутренней и внешней среды.

Секвенирование – это общее название методов, которые позволяют установить последовательность нуклеотидов в молекуле ДНК. В результате секвенирования получают формальное описание первичной структуры линейной макромолекулы в виде последовательности мономеров в текстовом виде.

Мутации – это изменения в ДНК клетки. Возникают под действием ультрафиолета, радиации (рентгеновских лучей) и т.п. Передаются по наследству. Эффект мутаций может выражаться либо в утрате функции, либо в приобретении новой функции.

Хромосомы – нуклеопротеидные структуры в ядре эукариотической клетки, в которых сосредоточена большая часть наследственной информации и которые предназначены для её хранения, реализации и передачи.

Аллели – различные формы одного и того же гена, расположенные в одинаковых участках (локусах) гомологичных хромосом и определяющие альтернативные варианты развития одного и того же признака. Локус означает местоположение определённого гена на генетической или цитологической карте хромосомы.

1.3 Обоснование необходимости и цели использования вычислительной техники для решения задачи

Существует более 20-и технологий проектирования организационно-технических систем и несколько сотен инструментов, предназначенных для автоматизации этого процесса. Поэтому, с учетом временного фактора, сравнительный анализ был ограничен наиболее популярным на российском рынке продуктом - AllFusion Process Modeler (BPwin).

AllFusion Process Modeler – средство программной поддержки моделирования в таких методиках как DFD, IDEF0 и IDEF3.

BPwin позволяет строить функциональные модели в каждой из методик по отдельности, так и гибридные функциональные модели, состоящие из диаграмм двух или сразу всех трех методик [24].

AllFusion Process Modeler умеет проверять создаваемые модели с точки зрения синтаксиса выбранной методологии, проверяет ссылочную целостность между диаграммами, а также выполняет ряд других проверок, чтобы помочь создать правильную модель, а не просто рисунок. При этом сохраняются главные преимущества рисунка – простота создания и наглядность [24,25].

Для того чтобы документировать механизмы передачи и обработки информации в моделируемой системе, используются диаграммы потоков данных (Data Flow Diagrams). Диаграммы DFD обычно строятся для наглядного изображения текущей работы системы документооборота организации. Чаще всего диаграммы DFD используют в качестве дополнения модели бизнес-процессов, выполненной в IDEF0[28].

Перед тем как перейти к разработке необходимо рассмотреть деятельность отдела и выявить недостатки в работе.

Моделирование проведено с помощью стандарта IDEF0. Функциональная модель IDEF0 представляет собой набор блоков, каждый из которых представляет собой «черный ящик» со входами и выходами, управлением и механизмами, которые детализируются (декомпозируются) до

необходимого уровня. Наиболее важная функция расположена в верхнем левом углу. А соединяются функции между собой при помощи стрелок и описаний функциональных блоков. При этом каждый вид стрелки или активности имеет собственное значение. Данная модель позволяет описать все основные виды процессов, как административные, так и организационные.

Стрелки могут быть:

– входящие – вводные, которые ставят определенную задачу;

– исходящие – выводящие результат деятельности;

– управляющие (сверху вниз) – механизмы управления (положения, инструкции);

– механизмы (снизу вверх) – что используется для того, чтобы произвести необходимую работу.

Контекстная диаграмма – это модель, представляющая систему как набор иерархических действий, в которой каждое действие преобразует некоторый объект или набор объектов [27]. На рисунке 1.4 представлена контекстная диаграмма деятельности селекционного отдела.

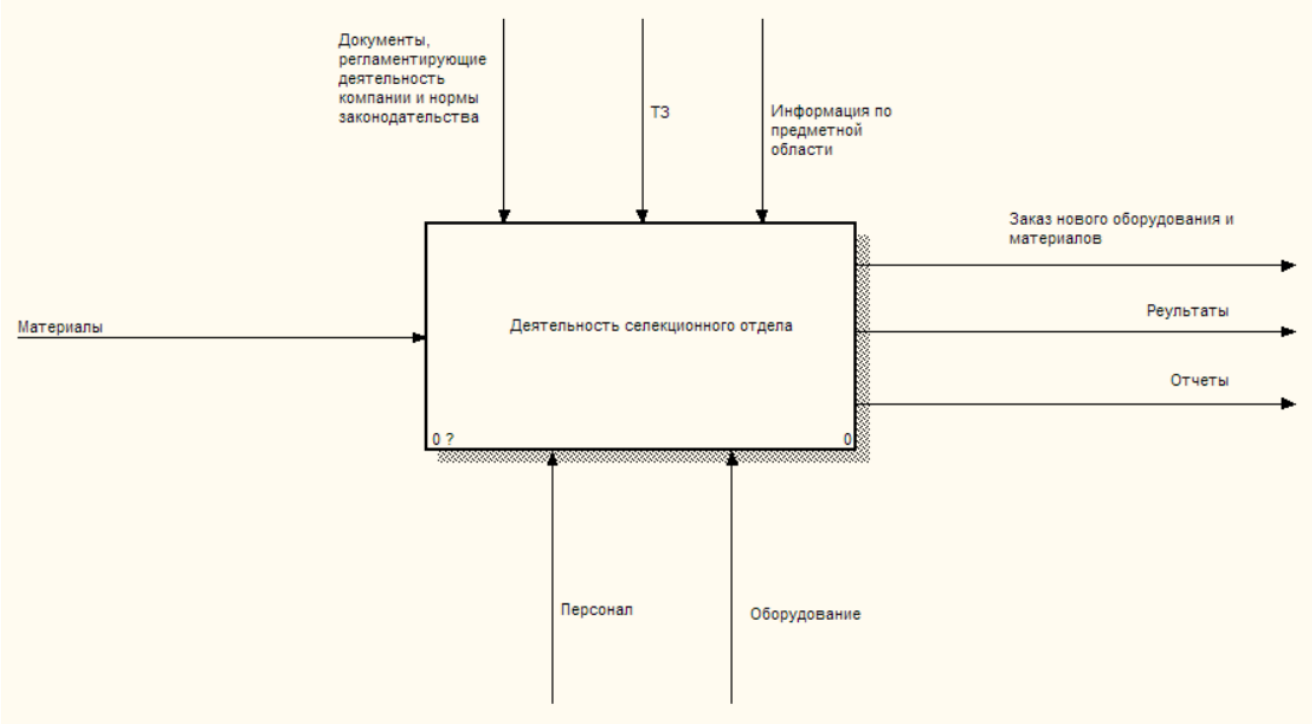


Рисунок 1.4 – Контекстная диаграмма «КАК ЕСТЬ»

На рисунке 1.5 изображена декомпозированная структурно-функциональная диаграмма первого уровня. На ней видно разделение деятельности предприятия на несколько функциональных подгрупп:

– управление отделом;
– сбор команды на проект;
– подготовка проекта;
– исполнение проекта.

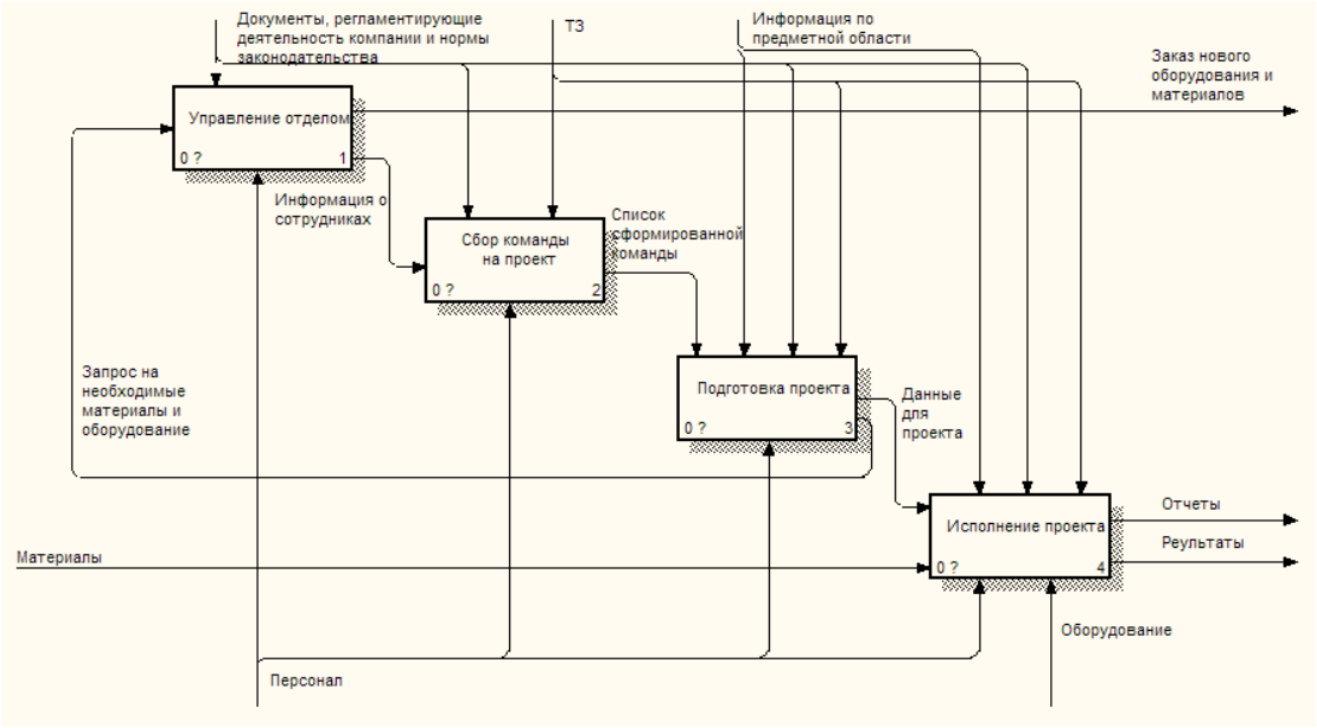


Рисунок 1.5 – Декомпозиция первого уровня

На рисунке 1.6 изображена декомпозиция блока первого уровня подготовка проекта. Декомпозиция включает следующие блоки:

– анализ предметной области;
– составление списка необходимых материалов и оборудования;
– составление запроса на необходимые материалы и оборудование;
– формирование данных по проекту.

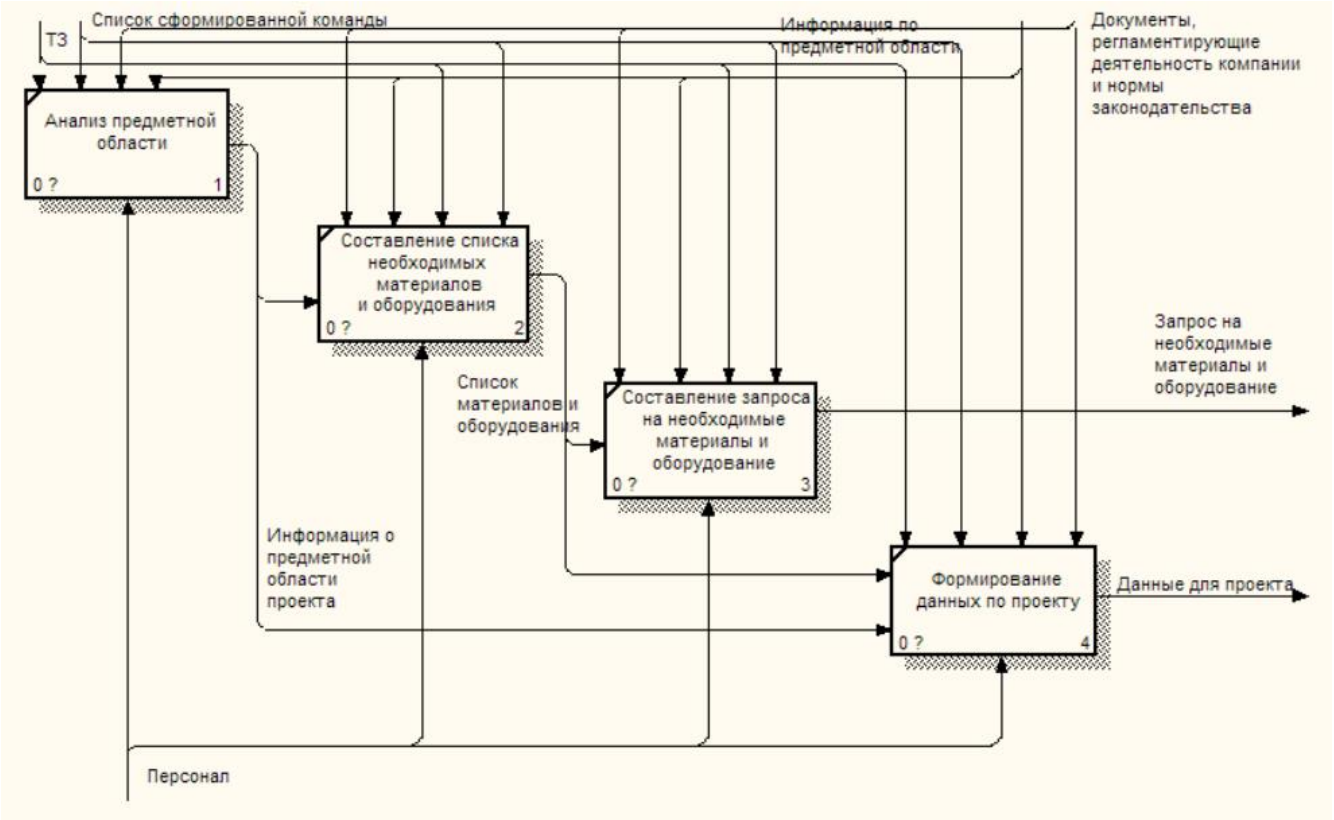


Рисунок 1.6 – Декомпозиция блока Подготовка проекта

На рисунке 1.7 изображена декомпозиция блока первого уровня исполнение проекта. Декомпозиция включает следующие блоки:

- распределение задач;
- подготовка оборудования и рабочих мест;
- проведение экспериментов;
- составление отчетов и результатов по проекту.

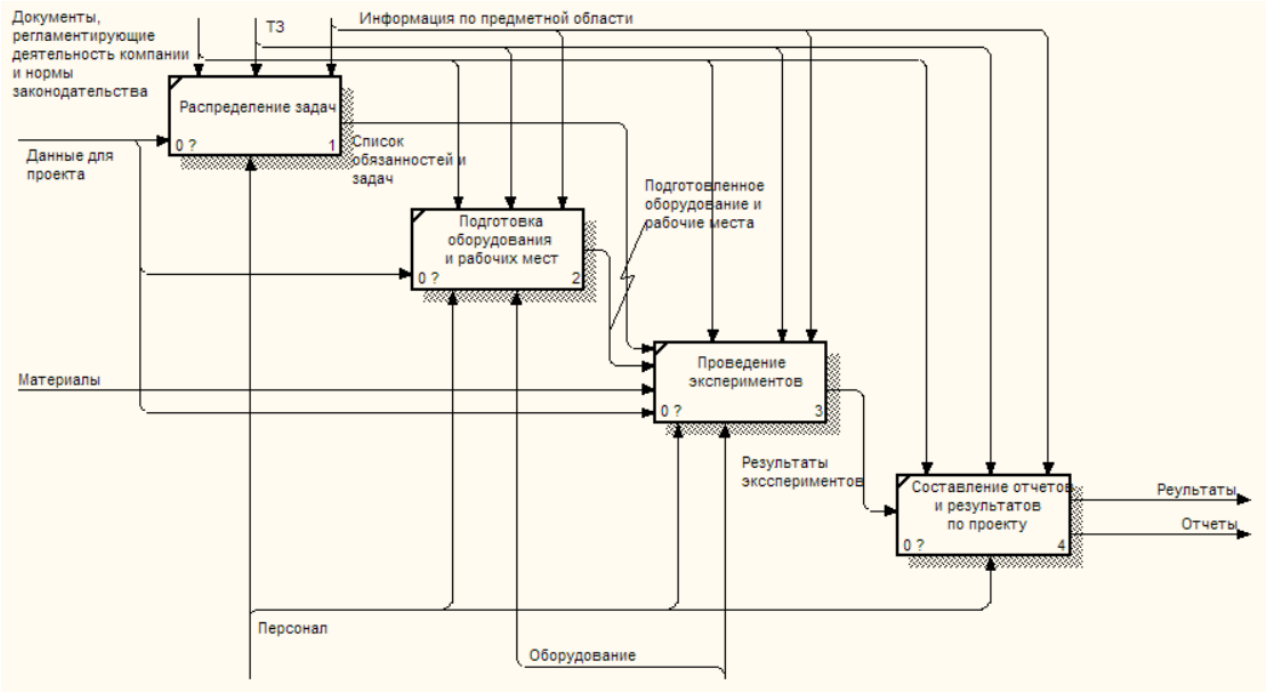


Рисунок 1.7 – Декомпозиция блока Исполнение проекта

Теперь, когда рассмотрена деятельность отдела, на основе структурно-функциональной диаграммы, можно выделить недостатки и определить по ним задачи. Главный недостаток – отсутствие базы данных. Необходимо хранить огромное количество различных данных от сортов, до их мутаций.

1.4 Постановка задачи

Задачей является разработка информационного сопровождения, целью которого являет хранение генетических и фенотипических данных сои. Основные требования к разрабатываемой системе:

– иметь подходящую для хранения данных структуру;

– содержать функции внесения и поиска данных;

– иметь возможность доработки системы в будущем и подключения других систем;

– предоставлять возможность использования системы нескольким пользователям с нескольких рабочих мест.

Информационное сопровождение – вариант информационного обеспечения, применяемый при формировании и реализации различного рода программ, научно-исследовательских и опытно-конструкторских работ [19].

Информационное обеспечение – это совокупность единой системы классификации и кодирования информации, унифицированных систем документации, схем информационных потоков, циркулирующих на предприятии, методология построения баз данных.

Источниками информации служат: документация, нормативно-справочная информация (устанавливаемая законодательными органами РБ), информация, поступающая от вышестоящих органов (например, казначейство, министерства), информация, поступающая от бухгалтерии с помощью локальной сети. Информационное обеспечение включает в себя внутримашинное и внемашинное информационное обеспечение.

Внемашинное информационное обеспечение включает различные документы на бумажных носителях (договора, приказы, распоряжения, отчеты, приходно-расходные ордера, ТТН, кассовые ордера и др.).

Внутримашинное информационное обеспечение включает информационную базу на машинном носителе и средства ее ведения. Данное обеспечение должно реализоваться в режиме реального масштаба времени, где изменения в данных, произведенные одним пользователем, сразу должны становиться доступными другим пользователям системы.

Разрабатываемая система будет являться веб-приложением, т.к. это обеспечит доступ к единой базе нескольким пользователям с нескольких рабочих мест. Для предоставления возможности доработки системы и подключения к ней других систем, информационное сопровождение будет разрабатываться с учетом архитектурного стиля REST.

1.5 Анализ существующих разработок и обоснование выбора технологии проектирования

При поиске существующих разработок были найдены два ресурса, которые могут подойти под требования селекционного отдела.

WheatPGE - база данных для анализа взаимоотношений генотипа, фенотипа и окружающей среды у пшеницы. База содержит информацию о количественных признаках опушения листа пшеницы, как примера одного из морфологических признаков растения, а также набору сортов и генетических маркеров как примеров описания генотипа растения. Пользователь может осуществлять поиск информации на основе запроса по фенотипическим характеристикам, сохранять имеющуюся информацию в формате CSV для дальнейшей статистической обработки.

База данных содержит три основных модуля, описывающие генотип, фенотип и условия окружающей среды. Информация, представленная этими блоками в базе данных связана между собой через таблицу «Растение». Растение описывается как набор фенотипических характеристик, параметров генотипа и условий произрастания.

Описание генотипа включает: сорт растения, линию, генотипы родителей, номер поколения для гибридных растений. Дополнительно генотип содержит информацию о генетических маркерах, связанных с геномными данными. Описание фенотипа включает: описание морфологии опушения (плотность трихом, распределение трихом по длине). Система позволяет создавать дополнительно различные типы отношений. Каждый тип может характеризовать определенный фенотипический признак. Описание окружающей среды включает: место произрастания (теплица или поле), средние значения температур для места произрастания, дату посева и т.п.

Данная база не подходит по ряду причин: модель базы подстроена исключительно под пшеницу, недостаточное количество таблиц и самих полей в таблицах для описания.

Другим ресурсом является SoyBase.

SoyBase – это генетическая база данных соевых бобов научно-исследовательского подразделения министерства сельского хозяйства США, которая является всеобъемлющим хранилищем для профессионально-кураторной генетики, геномики и соответствующих ресурсов данных для сои. SoyBase содержит самые современные генетические, физические и геномные карты последовательности, интегрированные с качественными и количественными признаками. Количественные ловушки признаков (QTL) представляют более 18 лет QTL-сопоставления более чем 90 уникальных признаков. SoyBase также содержит хорошо аннотированную геномную последовательность Williams 82 и связанные с ней инструменты для интеллектуального анализа данных. Генетические и последовательные виды хромосом сои и обширные данные о признаках и фенотипах тесно взаимосвязаны. Это позволяет входить в базу данных, используя практически любую доступную информацию, такую как символы генетической карты, имена генов сои или фенотипические признаки. SoyBase – это хранилище для контролируемых словарей для роста, развития и характеристик сои, которые также связаны с более общими онтологиями растений.

Данный ресурс близок к требованиям и находится в свободно доступе, но все же у него есть недостатки. Ресурс является полностью англоязычным, также загрузка информации (новых исследований и результатов) производится через администраторов ресурса.

В результате проведенного анализа и выявленных недостатков, было принято решение о целесообразности разработки собственного приложения, которое сможет удовлетворить все требуемые запросы.

В первом разделе были описаны предметная область и структура селекционного отдела, также была составлена модель «Как есть», поставлены задачи, проведен анализ существующих разработок и обоснована необходимость разработки информационного сопровождения.

2 Информационное обеспечение системы

2.1 Выбор средств разработки

При выборе средств разработки учитывались такие факторы как: их возможности, доступность, удобство использования и умение с ними работать. Для хранения данных была выбрана СУБД PostgreSQL – свободная объектно-реляционная система управления базами данных. Основными преимуществами, по сравнению с другими популярными СУБД, является возможность хранить большие объемы данных (таблица 2.1), высокопроизводительные и надёжные механизмы транзакций и репликации, легкая расширяемость, так же возможность хранить сложные и составные типы данных [7].

Таблица 2.1 – Размеры данных в PostgreSQL

Параметр	**Размер**
Максимальный размер базы данных	Нет ограничений
Максимальный размер таблицы	32 Тбайт
Максимальный размер записи	1,6 Тбайт
Максимальный размер поля	1 Гбайт
Максимум записей в таблице	Нет ограничений
Максимум полей в записи	250—1600, в зависимости от типов полей
Максимум индексов в таблице	Нет ограничений

Для написания кода был выбран ЯВУ Java. Программы на Java транслируются в байт-код Java, выполняемый виртуальной машиной Java (JVM) – программой, обрабатывающей байтовый код и передающей инструкции оборудованию как интерпретатор [1].

Достоинством подобного способа выполнения программ является полная независимость байт-кода от операционной системы и оборудования, что позволяет выполнять Java-приложения на любом устройстве, для которого существует соответствующая виртуальная машина. Другой важной особенностью технологии Java является гибкая система безопасности, в рамках которой исполнение программы полностью контролируется виртуальной машиной. Любые операции, которые превышают установленные полномочия программы (например, попытка несанкционированного доступа к данным или соединения с другим компьютером), вызывают немедленное прерывание [31,35].

Помимо этого, есть еще ряд преимуществ:

– автоматическое управление памятью;

– расширенные возможности обработки исключительных ситуаций;

– наличие классов, позволяющих выполнять HTTP-запросы и обрабатывать ответы;

– унифицированный доступ к базам данных:

– на уровне отдельных SQL-запросов – на основе JDBC, SQLJ;

– на уровне концепции объектов, обладающих способностью к хранению в базе данных – на основе Java Data Objects и Java Persistence API.

Java Persistence API (JPA) – спецификация API Java EE, предоставляет возможность сохранять в удобном виде Java-объекты в базе данных. Существует несколько реализаций этого интерфейса, одна из самых популярных использует для этого Hibernate. JPA реализует концепцию ORM.

Hibernate – самая популярная реализация спецификации JPA, предназначенная для решения задач объектно-реляционного отображения (ORM). Распространяется свободно на условиях GNU Lesser General Public License [16].

Spring Framework (или коротко Spring) – универсальный фреймворк с открытым исходным кодом для Java -платформы. Spring обеспечивает решения многих задач, с которыми сталкиваются Java-разработчики и организации,

которые хотят создать информационную систему, основанную на платформе Java. Из-за широкой функциональности трудно определить наиболее значимые структурные элементы, из которых он состоит. Spring не всецело связан с платформой Java Enterprise, несмотря на его масштабную интеграцию с ней, что является важной причиной его популярности [3,4].

Spring может быть рассмотрен как коллекция меньших фреймворков или фреймворков во фреймворке. Большинство этих фреймворков может работать независимо друг от друга, однако они обеспечивают большую функциональность при совместном их использовании. При разработке будут использованы следующие фреймворки:

- inversion of control-контейнер: конфигурирование компонентов приложений и управление жизненным циклом Java-объектов;
- фреймворк аспектно-ориентированного программирования: работает с функциональностью, которая не может быть реализована возможностями объектно-ориентированного программирования на Java без потерь;
- фреймворк доступа к данным: работает с системами управления реляционными базами данных на Java-платформе, используя JDBC- и ORM-средства и обеспечивая решения задач, которые повторяются в большом числе Java-based environments;
- фреймворк управления транзакциями: координация различных API управления транзакциями и инструментарий настраиваемого управления транзакциями для объектов Java;
- фреймворк MVC: каркас, основанный на HTTP и сервлетах, предоставляющий множество возможностей для расширения и настройки (customization);
- фреймворк аутентификации и авторизации: конфигурируемый инструментарий процессов аутентификации и авторизации, поддерживающий много популярных и ставших индустриальными стандартами протоколов, инструментов, практик через дочерний проект Spring Security.

IntelliJ IDEA – интегрированная среда разработки программного обеспечения для многих языков программирования, в частности Java, JavaScript, Python, разработанная компанией JetBrains. Дизайн среды ориентирован на продуктивность работы программистов, позволяя сконцентрироваться на функциональных задачах, в то время как IntelliJ IDEA берёт на себя выполнение рутинных операций. Начиная с версии 9.0, среда доступна в двух редакциях: Community Edition и Ultimate Edition. Community Edition является полностью свободной версией, доступной под лицензией Apache 2.0, в ней реализована полная поддержка Java SE, Groove, Scala, а также интеграция с наиболее популярными системами управления версиями [17].

Angular (версия 2 и выше) – это открытая и свободная платформа для разработки веб-приложений, написанная на языке TypeScript, разрабатываемая командой из компании Google, а также сообществом разработчиков из различных компании. Angular - это полностью переписанный фреймворк от той же команды, который назывался AngularJS. Прежде всего он нацелен на разработку SPA-решений (Single Page Application), то есть одностраничных приложений [13].

TypeScript – язык программирования, представленный Microsoft в 2012 году и позиционируемый как средство разработки веб-приложений, расширяющее возможности JavaScript/ TypeScript является обратно совместимым с JavaScript и компилируется в последний. Фактически, после компиляции программу на TypeScript можно выполнять в любом современном браузере или использовать совместно с серверной платформой Node.js. Код экспериментального компилятора, транслирующего TypeScript в JavaScript, распространяется под лицензией Apache. Его разработка ведётся в публичном репозитории через сервис GitHub [15].

TypeScript отличается от JavaScript возможностью явного статического назначения типов, поддержкой использования полноценных классов (как в традиционных объектно-ориентированных языках), а также поддержкой подключения модулей, что призвано повысить скорость разработки, облегчить

читаемость, рефакторинг и повторное использования кода, помочь осуществлять поиск ошибок на этапе разработки и компиляции, и, возможно, ускорить выполнение программ.

2.2 Информационная модель и ее описание

Была составлена функциональная модель «Как должно быть». Как правило, данная модель создается на основе «Как есть», с устранением недостатков в существующей организации бизнес-процессов, а также с их совершенствованием и оптимизацией. Это достигается за счет устранения выявленных на базе анализа «Как есть» узких мест. На рисунке 2.1 представлена диаграмма модели «Как должно быть», а именно декомпозиция первого уровня.

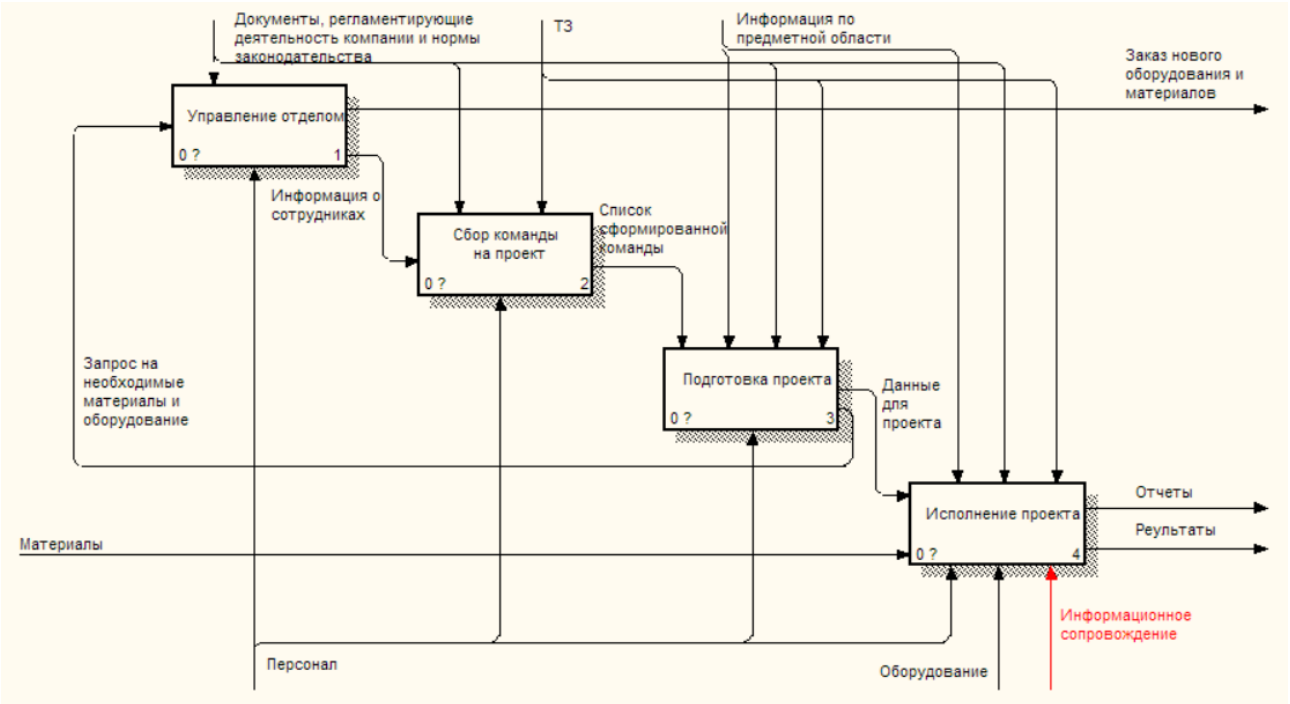


Рисунок 2.1 – Декомпозиция первого уровня

Как видно на рисунке, теперь появилась стрелка механизма «Информационное сопровождение». На рисунке 2.2 изображена декомпозиция блока «Исполнение проекта».

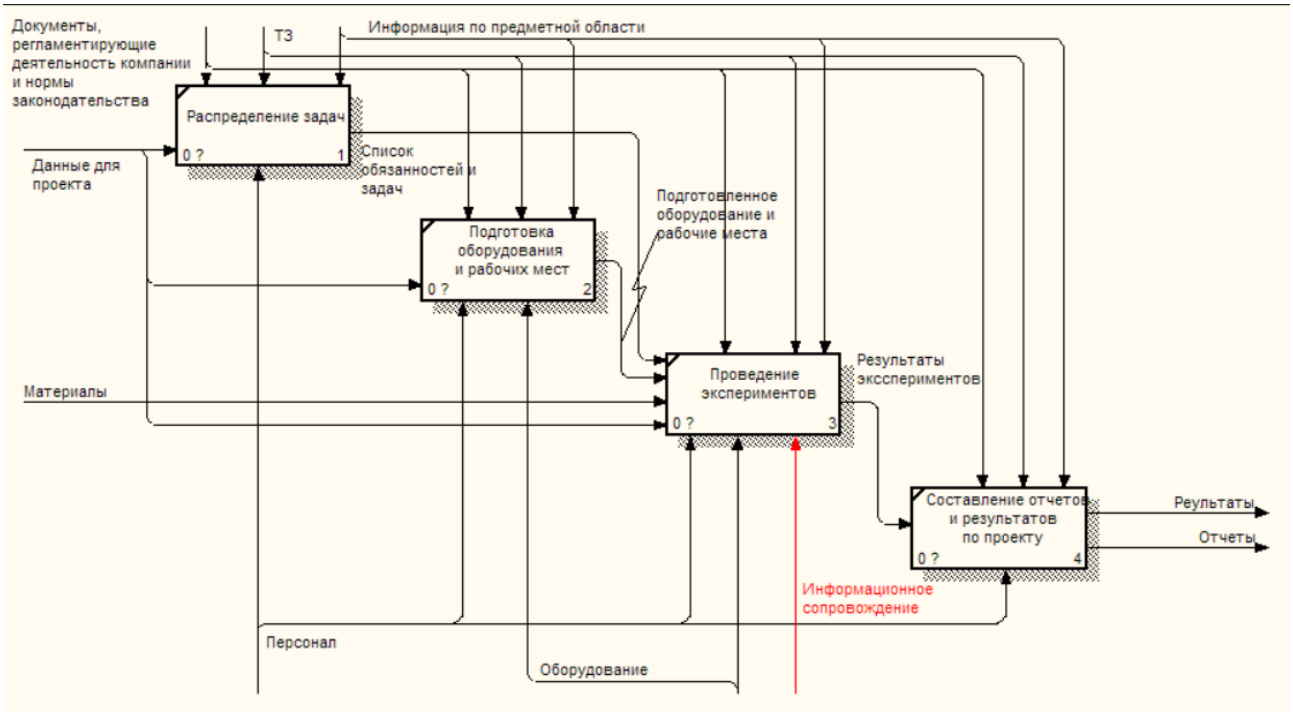


Рисунок 2.2 – Декомпозиция блока «Исполнение проекта»

Механизм «Информационное сопровождение» входит в блок «Проведение экспериментов». Это необходимо для хранения данных, полученных в ходе проведения экспериментов и использования их в дальнейшем.

2.3 Моделирование базы данных

В ходе моделирования была построена модель базы данных «сущность-связь», наглядно показывающая таблицы и их взаимодействие между собой. Для описания схемы базы данных применяется диаграмма “сущности-связи” (Entity-Relationship diagram или ER diagram). Существуют различные варианты диаграмм “сущности-связи”. Способы изображения элементов ER-диаграмм называют нотациями. На них одни и те же элементы графически изображаются по-разному. Известны нотация Мартина, нотация IDEF1X и др. Кроме того, различные программные средства, реализующие одну и ту же нотацию, могут отличаться своими возможностями [16].

Все варианты диаграмм “сущность-связь” исходят из одной идеи – рисунок всегда нагляднее текстового описания. Все такие диаграммы используют графическое изображение сущностей предметной области, их свойств (атрибутов), и взаимосвязей между сущностями [18].

Для информационного сопровождения поставленной задачи была разработана база данных. Модель базы включает порядка 40 взаимосвязанных таблиц, поэтому для наглядности она была разделена на три части. На рисунке 2.3 показана структура модели базы данных.

Рисунок 2.3 – Модель базы

Базу можно разделить на таблицы-словари, таблицы, связанные с анализами материала и хранением их, а также таблицы, связанные с нахождением мутаций, закономерностей и изменений.

На рисунке 2.4 представлена физическая модель базы.

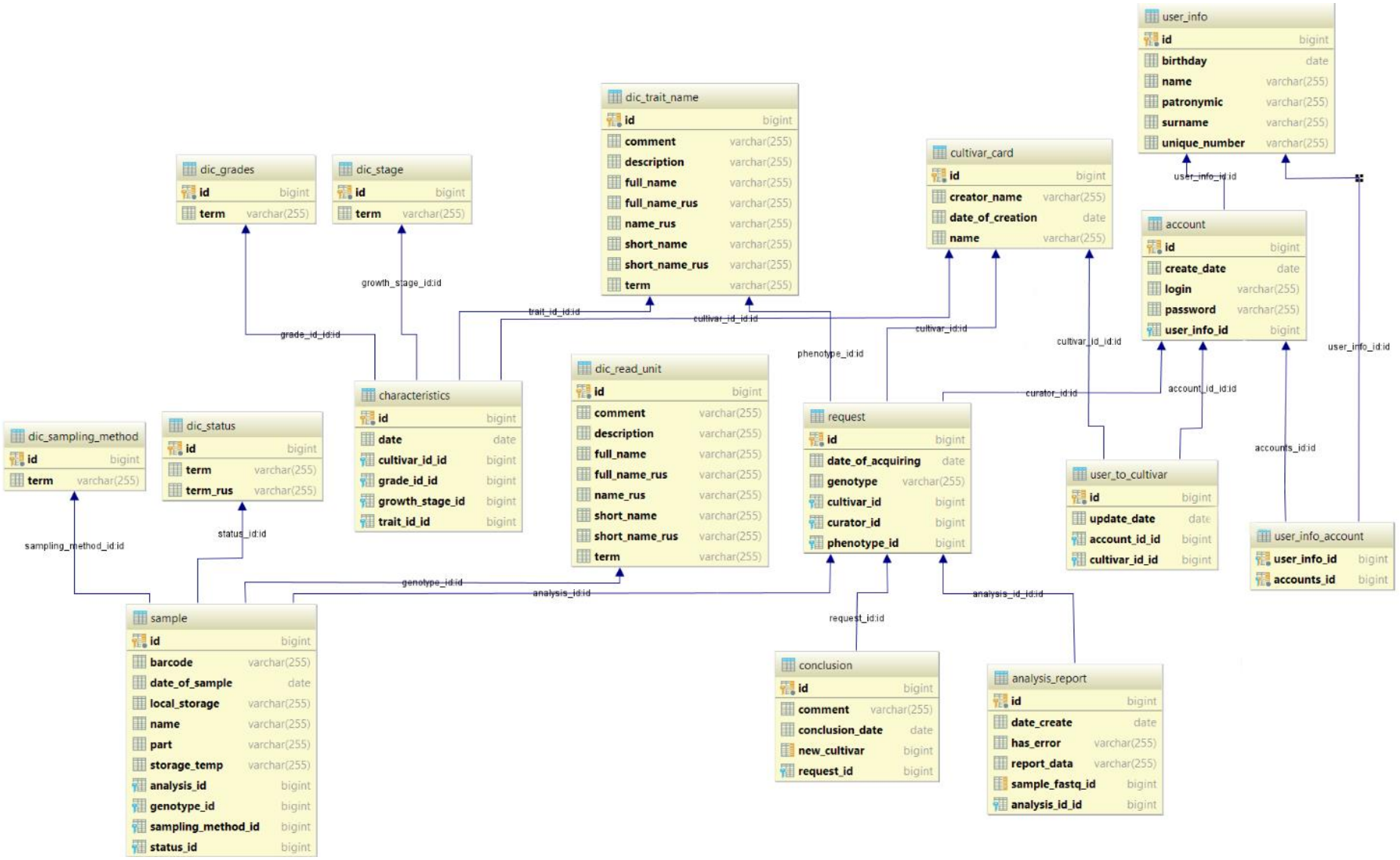


Рисунок 2.4 – Первая часть модели базы данных

В первой части модели таблицы имеют следующие назначения:

- User_info – информация о пользователе;
- Account – пользователи;
- Cultivar_card – информация о сорте сои;
- User_to_cultivar – информация о том, кто является куратором сорта сои;
- Request – информация о направлении на генетический анализ;
- Characteristics – данные о характеристиках/признаках сорта сои;
- Sample – таблица данных о пробе, взятой для анализа;
- Conclusion – информация о совпадении фенотипа с фенотипом, прогнозированным на основе генотипического анализа пробы;
- Analysis_report – информация о файле с данными результатов анализа, его названии, типе и платформе;
- Dic_trait_name – список названий всех известных характеристик/признаков сои;
- Dic_sampling_method – список методов забора генетического материала;
- Dic_status – список статусов анализируемых образцов;
- Dic_read_unit – список единиц измерения (ридов);
- Dic_stage – список этапов роста сои;
- Dic_grades – список степеней развития боба сои.

Датологическая модель первой части БД находится в приложении Б.

Во второй части модели показаны таблицы:

- Account – пользователи;
- Cultivar_card – информация о сорте сои;
- Sample – таблица данных о пробе, взятой для анализа;
- Processing_status – список статусов обработки анализа;
- File_repository – информация о файле с данными результатов анализа, его названии, типе и платформе;

– Processing – информация обо всех обработках, которые запускались в системе (какой пользователь инициировал обработку, какой сорт сои, дата запуска обработки, статус обработки);

– Processing_stage – информация об описаниях этапов обработки данных;

– Processing_data – информация о результатах обработки анализа;

– Dic_platform – список платформ (SSR панели, тип секвенатора и т.п.) для анализа ДНК;

– Dic_file_type – список типов файлов.

Модель таблиц показана на рисунке 2.5

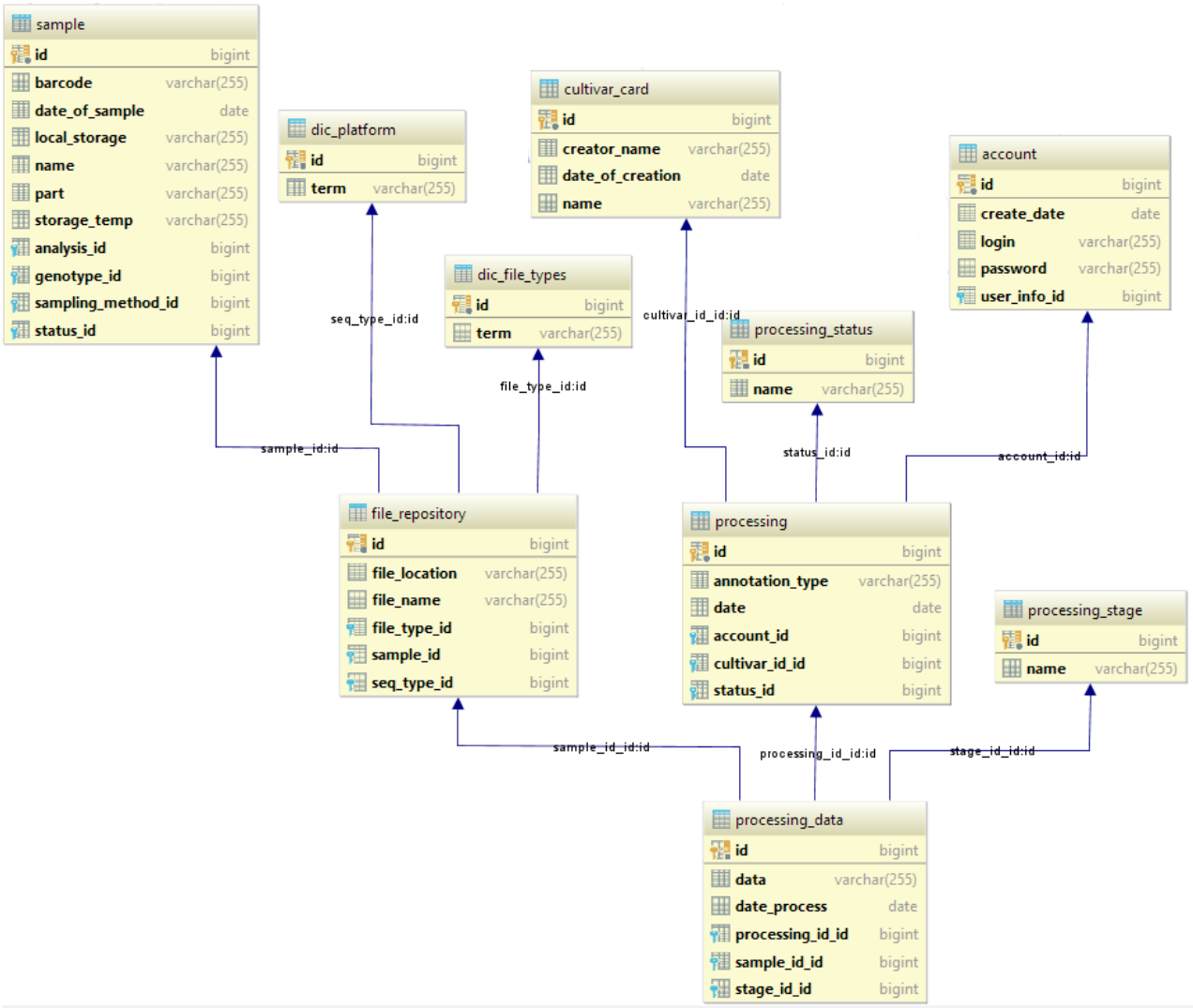


Рисунок 2.5 – Вторая часть модели базы данных

Таблица 2.2 – Датологическая модель второй части БД

Сущность	Идентификатор таблицы	Атрибут	Идентификатор поля	Тип поля
1	2	3	4	5
Данные о пробе	Sample	Id	id (pk)	bigint
		Дата взятия образца	date_of_sample	date
		Имя образца	name	varchar
Данные о пробе	Sample	Часть растения	part	varchar
		Место хранение	local_place	varchar
		Штрихкод	barcode	varchar
		Температура	storage_temp	varchar
		Направление	analysis_id (fk)	bigint
		Метод взятия пробы	sampling_method_id (fk)	bigint
		Статус пробы	status_id (fk)	bigint
		Количество ридов	genotype_id (fk)	bigint
Информация о сорте	Cultivar_card	id	id (pk)	bigint
		Название сорта	name	varchar
		Имя производителя	creator_name	varchar
		Дата получения	date_of_creation	date
Пользователь	Account	id	id (pk)	bigint
		Логин	login	varchar
		Пароль	password	varchar
		Дата создания	create_date	date
		Информация о пользователе	user_info_id (fk)	bigint
Платформа для анализа	Dic_platform	id	id (pk)	bigint
		Название	term	varchar
Тип файла	Dic_file_types	id	id (pk)	bigint
		Название	term	varchar
Информация о файле с данными результатов анализа	File_repository	id	id (pk)	bigint
		Название файла	file_name	varchar
		URL файла	file_location	varchar
		Тип файла	file_type_id	bigint
		Платформа	seq_type_id	bigint
		Проба	sample_id	bigint
Статус обработки	Processing_ status	id	id (pk)	bigint
		Название	name	varchar
Информация обо всех обработках	Processing	id	id (pk)	bigint
		Дата запуска	date	varchar
		Тип аннотации	annotation_type	varchar
		Сорт сои	cultivar_id	bigint
		Пользователь	account_id	bigint
		Статус обработки	status_id	bigint

Таблица 2.2 – Продолжение

1	2	3	4	5
Информация о результатах обработки	Processing_data	id	id (pk)	bigint
		Получаемые данные	data	varchar
		Дата обработки	date_process	varchar
		Файл с результатами анализа	sample_id	bigint
		Этап обработки	stage_id	bigint
		Обработка	processing_id	bigint
Этапы обработки	Processing_ stage	id	id (pk)	bigint
		Название	name	varchar

Третья часть модели, представленная на рисунке 2.6, состоит из таблиц:

- Sample – таблица данных о пробе, взятой для анализа;
- Sample_storage – способ и место хранения образца растения;
- Analysis_results – информация о результатах анализа пробы;
- Db_comparison – информация о сравнении результатов анализа пробы с базой данных;
- Mutation – информация об изменениях в покрытии хромосом, которые были найдены у исследуемых сортов сои по которым делаются выводы о характеристике/признаке данного сорта;
- Source – информация об источнике данных, статье, патенте и т.п;
- Dic_storage_place – список мест хранения образца в лаборатории;
- Dic_read_unit – список единиц измерения (ридов);
- Dic_experiment_type – список типов постановки экспериментов;
- Dic_mut_type – список типов мутации;
- Dic_chromosome – список хромосом генома сои;
- Dic_zygocity – список эффектов зиготности мутации влияющей на характеристику/признак сои;
- Dic_mut_effect – список эффектов мутации/полиморфизма.

Таким образом, во втором разделе был проведен выбор программных средств и построена модель «Как должно быть». Также была разработана модель базы данных для разрабатываемого приложения.

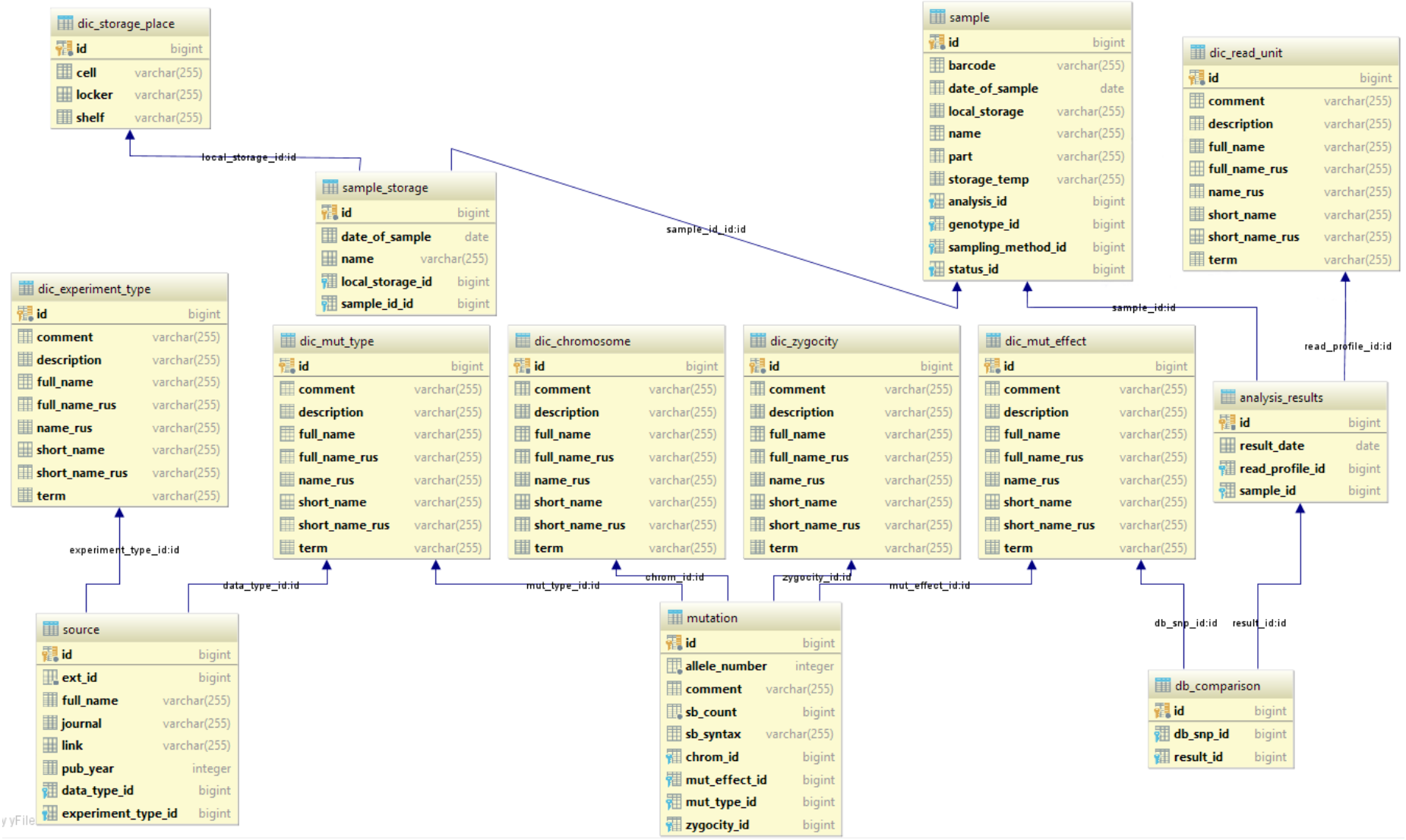


Рисунок 2.6 – Третья часть модели базы данных

3 Разработка информационного обеспечения

3.1 Разработка серверной части

Разработку серверной части можно разделить на две части: создание базы данных и написание репозиториев к базе. Репозитории будут созданы по архитектуре REST. Проще говоря, архитектура REST – это комплекс решений, связанных с передачей информации о состоянии ресурса в некоторой форме от сервера клиенту (или наоборот) [8].

Для разработки базы будут использовано следующее ПО: СУБД PostgreSQL, среда разработки IntelliJ IDEA, язык высокого уровня Java, фреймворк Spring. Перед написанием кода настраивается приложение. Для этого в файле application.yaml были добавлены перечисленные на рисунке 3.1 свойства.

```
spring:
  datasource:
    url: jdbc:postgresql://localhost:5432/EFKO
    username: postgres
    password: 123
    driver-class-name: org.postgresql.Driver
  jpa:
    hibernate:
      ddl-auto: update
    show-sql: true
    database-platform: org.hibernate.dialect.PostgreSQLDialect
```

Рисунок 3.1 – Свойства приложения

Свойства имеют иерархическую структуру, т.е. в каталоге spring, есть два подкаталога: datasource и jpa. В свойстве url указан путь к базе данных, в username и password – логин и пароль для подключения к базе данных и driver-class-name – драйвер, для подключения к базе. Все указанное позволяет взаимодействовать со многими СУБД, но так как используется СУБД PostgreSQL, то и подключаются драйверы для этой СУБД. Свойство hibernate ddl-auto: update обновляет существующую схему базы. Также, помимо update,

есть другие свойства: create – создает схему, create-drop – создает схему при запуске и удаляет ее при завершении. Свойство Show-sql включает, либо отключает логирование базы.

На рисунке 3.2 показан пример кода создания таблицы.

```
/**
Список эффектов мутации/полиморфизма на функцию гена (подобъект "Фенотип")

 */
package com.db.Tables.Dictionary;

import lombok.Data;
import lombok.NoArgsConstructor;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;

@Entity
@Data
@NoArgsConstructor
public class DicMutEffect {
    @Id
    @GeneratedValue (strategy = GenerationType.AUTO)
    private long id;
    private String term; // Название мутации на английском
    private String fullName; // Полное название мутации на английском языке
    private String shortName;
    private String nameRus; // Название мутации на русском
    private String fullNameRus;
    private String shortNameRus;
    private String comment; // Внутренний  комментарий
    private String description; // Описание
}
```

Рисунок 3.2 – Код создания таблицы Dic_mut_effect

Аннотация Entity указывает, что данный класс является сущностью, в скобках после аннотации можно указать, как будет называться таблица в базе данных, если этого не сделать, то таблица по умолчанию называется именем класса. Аннотация Id ставится над полем, которое будет являться первичным ключом в таблице, а аннотация GeneratedValue указывает на стратегию автоинкремента. Также используются аннотации библиотеки Lombok – Data и NoArgsConstructor, которые автоматически создают для класса геттер, сеттер, конструктор и переопределяют методы toString, hashCode и equals.

В результате выполнения этого кода Spring Boot находит аннотацию Entity и опознает ее как класс-сущность. Далее с помощью Hibernate была создана в базе данных таблица с названием класса Dic_mut_effect, в которой находятся все перечисленные поля. Результат выполнения кода был проверен в приложении pgAdmin (GUI для PostgreSQL) (рис. 3.3).

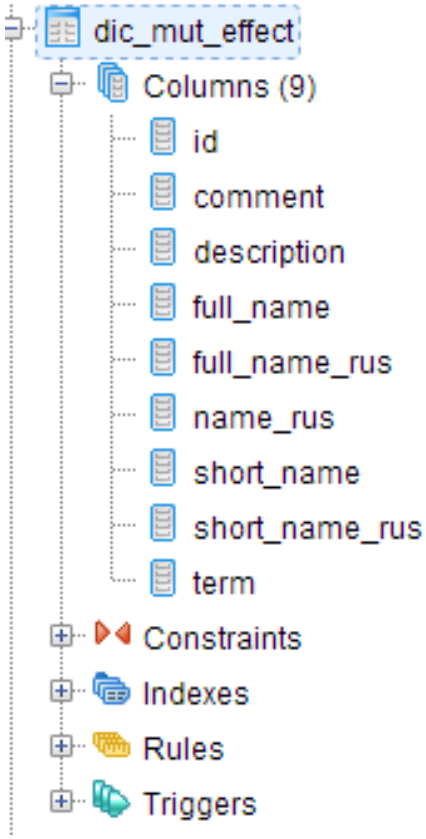


Рисунок 3.3 – Созданная таблица

На рисунке 3.4 представлен разработанный код создания таблицы селекционных учреждений. В отличии от кода предыдущей таблицы, здесь над полем region стоит аннотация ManyToOne, а тип этого поля – другая таблица. С помощью этой аннотации создается связь многие к одному с таблицей Dic_regions. Также есть аннотации OneToOne, OneToMany и ManyToMany, которые устанавливают связь один к одному, один ко многим и многие ко многим соответственно.

```java
/**...*/
package com.db.Tables;

import com.db.Tables.Dictionary.DicRegions;
import lombok.Data;
import lombok.NoArgsConstructor;

import javax.persistence.*;

@Entity
@Data
@NoArgsConstructor
public class Institution {
    @Id
    @GeneratedValue (strategy = GenerationType.AUTO)
    private long id;
    private String name; // Название учреждения
    private String shortName;
    private String zipCode; // Почтовый индекс

    @ManyToOne
    private DicRegions region;

    private String city; //
    private String houseNum;
    private String bldgNum; // Номер корпуса или строения
    private String other; // Другие данные

}
```

Рисунок 3.4 – Код таблицы Institution

Остальные 40 таблиц были созданы в базе аналогичным образом, поэтому не целесообразно приводить код каждой таблицы.

После создания всех таблиц для них были написаны репозитории, которые позволяют взаимодействовать с таблицами, т.е. просматривать их записи, добавлять новые, редактировать и удалять данные и так далее.

На рисунке 3.5 представлен код репозитория для таблицы Dic_mut_effect.

```java
package com.db.Repository.DicRepository;

import com.db.Tables.Dictionary.DicMutEffect;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.repository.query.Param;
import org.springframework.data.rest.core.annotation.RepositoryRestResource;
import org.springframework.web.bind.annotation.CrossOrigin;

import java.util.List;

@CrossOrigin
@RepositoryRestResource (path = "dicMutEffect")
public interface DicMutEffectRepository extends JpaRepository<DicMutEffect, Long>{

    List<DicMutEffect> findByTerm(@Param("term")String term);
    List<DicMutEffect> findByFullName(@Param("fullName")String fullName);
    List<DicMutEffect> findByShortName(@Param("shortName")String shortName);
    List<DicMutEffect> findByNameRus(@Param("nameRus") String nameRus);
    List<DicMutEffect> findByFullNameRus(@Param("fullNameRus") String fullNameRus);
    List<DicMutEffect> findByShortNameRus(@Param("shortNameRus") String shortNameRus);
```

Рисунок 3.5 – Код репозитория таблицы Dic_mut_effect

Для репозитория был создан интерфейс, который наследуется от класса JpaRepository и принимает два параметра: класс таблицы, для которой создается репозиторий и оберточный тип уникального поля (id). Для опознания репозитория в Spring была создана над интерфейсом аннотация RepositoryRestResource. В скобках аннотации указывается путь, по которому можно получить доступ к данной таблице. Аннотация CrossOrigin нужна для разрешения получения доступа к таблице из других приложений, иначе будет ошибка доступа.

В теле интерфейса были созданы методы для поиска по каждому полю таблицы. Например, метод findByTerm ищет записи в таблице по полю term и возвращает список объектов Dic_mut_effect. Аналогично работают и другие созданные методы: findByNameRus позволяет искать записи по полю nameRus, т.е. по полю «название на русском», findByShortName производит поиск по полю shortName, в котором указывается краткое наименование эффекта мутации на английском.

При переходе на URL localhost:8080, на странице отображается JSON, в котором указаны ссылки на все созданные таблицы (рис. 3.6).

Рисунок 3.6 – JSON

JSON (JavaScript Object Notation) – простой формат обмена данными, удобный для чтения и написания как человеком, так и компьютером. Он основан на подмножестве языка программирования JavaScript, определенного в стандарте ECMA-262 3rd Edition - December 1999. JSON – текстовый формат, полностью независимый от языка реализации, но он использует соглашения, знакомые программистам C-подобных языков, таких как C, C++, C#, Java, JavaScript, Perl, Python и многих других. Эти свойства делают JSON идеальным языком обмена данными. [6]

При переходе по какому-либо адресу, отображается JSON этой таблицы с ее данными и другими ссылками. При отсутствии данных в таблицах будут выведены только ссылки, как на рисунке 3.7.

Рисунок 3.7 – Данные по таблице source

Здесь указана ссылка http://localhost:8080/source/search, при переходе на которую отображаются методы поиска, имеющиеся для данной таблицы (рис. 3.8). В фигурных скобках указан параметр, который нужно написать для поиска. Например, для поиска объектов 2011 года публикации формат URL: http://localhost:8080/source/search/findByPubYear?year=2011/

Рисунок 3.8 – Методы поиска

Репозитории для других таблиц создаются подобным образом, поэтому показывать здесь их код нецелесообразно.

После создания базы и доступа к ней есть, была разработана клиентская часть приложения.

3.2 Разработка клиентской части

Для разработки клиентской части приложения был выбран фреймворк Angular 2. Для написания кода, в котором используются язык TypeScript и язык разметки HTML. Так как в ответе с сервера приходят данные в формате JSON, то для начала необходимо написать модель принимаемых данных. Для примера можно создать модель данных таблицы эффектов мутации (рис. 3.9).

```
/*...*/
export class MutEffect {
  term: string;
  fullName: string;
  shortName: string;
  nameRus: string;
  fullNameRus: string;
  shortNameRus: string;
  comment: string;
  description: string;
  href: string;

  constructor(obj?: any) {
    this.term = obj.term;
    this.fullName = obj.fullName;
    this.shortName = obj.shortName;
    this.nameRus = obj.nameRus;
    this.fullNameRus = obj.fullNameRus;
    this.shortNameRus = obj.shortNameRus;
    this.comment = obj.comment;
    this.description = obj.description;
    this.href = obj.href;

  }
}
```

Рисунок 3.9 – Модель принимаемых данных

Название полей соответствует названию полей в базе данных. Был создан сервис для данной таблицы. В сервисе находятся методы для взаимодействия с данными таблицы. На рисунке 3.10 представлен такой сервис. Созданная переменная http типа HttpClient. HttpClietn – компонент, предоставляющий методы для работы с протоколом HTTP. В дальнейшем потребуется для методов сервиса.

```
import ...

@Injectable()
export class MutEffectService {

  constructor(private http: HttpClient) {}

  url = 'http://localhost:8080/dicMutEffect';
  searchUrl = 'http://localhost:8080/dicMutEffect/search/';

  public getMutEffect(): Observable<MutEffect[]> {
    return this.http.get<MutEffect[]>(this.url)
      .pipe(map( project: response => {
        return <MutEffect[]>response['_embedded']['dicMutEffects'].map(mutEffect => {
          return new MutEffect({
            term: mutEffect.term,
            fullName: mutEffect.fullName,
            shortName: mutEffect.shortName,
            nameRus: mutEffect.nameRus,
            fullNameRus: mutEffect.fullNameRus,
            shortNameRus: mutEffect.shortNameRus,
            comment: mutEffect.comment,
            description: mutEffect.description,
            href: mutEffect['_links']['self'].href

          });
        });
      }));
  }
```

Рисунок 3.10 – Класс MutEffectService

В теле сервиса созданы переменные с адресами и метод getMutEffect(). Метод не принимает никаких параметров, но возвращает список объектов типа MutEffect, модель которого была написана ранее. Метод обрабатывает JSON-объект, оставляя только нужную информацию, а именно данные полей таблицы. На рисунке 3.11 показаны остальные методы сервиса. Метод addMutEffect добавляет в таблицу новую запись, принимая объект типа MutEffect. Метод deleteMutEffect принимает ссылку на объект и удаляет его из базы. Метод searchByTerm принимает строку, содержащую название объекта и отправляет get-запрос на поиск объектов по названию, возвращая при этом обработанный ответ от сервера.

```
public addMutEffect(me: MutEffect): Observable<MutEffect> {
  return this.http.post<MutEffect>(this.url, me)
    .pipe(map( project: mutEffect => {
      return new MutEffect({
        term: mutEffect.term,
        fullName: mutEffect.fullName,
        shortName: mutEffect.shortName,
        nameRus: mutEffect.nameRus,
        fullNameRus: mutEffect.fullNameRus,
        shortNameRus: mutEffect.shortNameRus,
        comment: mutEffect.comment,
        description: mutEffect.description,
        href: mutEffect['_links']['self'].href
      });
    }));
}

public deleteMutEffect(href: string): Observable<any> {
  return this.http.delete(href);
}

public searchByTerm(term: string): Observable<MutEffect[]> {
  return this.http.get<MutEffect[]>( url: this.searchUrl + 'findByTerm?term=' + term)
    .pipe(map( project: response => {
      return <MutEffect[]> response['_embedded']['dicMutEffects'].map(mutEffect => {
      return new MutEffect({
        term: mutEffect.term,
        fullName: mutEffect.fullName,
        shortName: mutEffect.shortName,
        nameRus: mutEffect.nameRus,
        fullNameRus: mutEffect.fullNameRus,
        shortNameRus: mutEffect.shortNameRus,
        comment: mutEffect.comment,
        description: mutEffect.description,
        href: mutEffect['_links']['self'].href
      });
```

Рисунок 3.1 – Методы сервиса

В сервисе была описана логика методов. Для их использования был создан компонент, при котором генерируется html-файл, css, и ts-файл, в котором и создаются методы, использующие методы сервиса. На рисунке 3.12 представлен компонент mut-effect.component.ts

```
import { Component, OnInit } from '@angular/core';
import {MutEffectService} from '../service/mut-effect.service';
import {MutEffect} from '../models/mut-effect';

@Component({
  selector: 'app-mut-effect',
  templateUrl: './mut-effect.component.html',
  styleUrls: ['./mut-effect.component.css']
})
export class MutEffectComponent implements OnInit {
  mutEffect: MutEffect[];

  constructor(private mutEffectService: MutEffectService) { }

  ngOnInit() {
    this.getAll();
  }

  getAll(): void {
    this.mutEffectService.getMutEffect()
      .subscribe( next: response => {
        this.mutEffect = response;
      });
  }

  getByTerm(term: string): void {
    this.mutEffectService.searchByTerm(term)
      .subscribe( next: response => {
        this.mutEffect = response;
      });
  }
```

Рисунок 3.12 – Компонент mut-effect

В компоненте указаны: selector – имя для обращения к компоненту, templateUrl – путь к представлению компонента и styleUrls – массив css-файлов. Далее, в самом классе была создана переменная типа MutEffect, которая принимает значения объектов с сервера. В конструкторе создана переменная типа сервиса, которая получает все методы, написанные выше. Метод ngOnInit() применяется для какой-то комплексной инициализации компонента. Здесь можно выполнять загрузку данных с сервера или из других источников данных. Далее уже идут методы. Метод getAll() использует логику сервиса для получения всех данных таблицы с сервера и присваивает их ранее созданной переменной mutEffect. Так же работает и метод getByTerm. Еще в компоненте есть методы, представленные на рисунке 3.13. Метод del() использует метод

для удаления записи из сервиса, передает ссылку и в этот метод и снова вызывает метод getAll() для обновления данных на странице. Метод add принимает все поля таблицы и создает новый объект типа MutEffect, который передает в метод сервиса, для создания новой записи в таблице и обновляет данные на странице.

```
  del(href: string): void {
    this.mutEffectService.deleteMutEffect(href).subscribe( next: () => {
      this.getAll();
    });
  }

  add(term: string, fullName: string, shortName: string, nameRus: string,
      fullNameRus: string, shortNameRus: string, comment: string, description: string): void {
    this.mutEffectService.addMutEffect(new MutEffect({
      term: term,
      fullName: fullName,
      shortName: shortName,
      nameRus: nameRus,
      fullNameRus: fullNameRus,
      shortNameRus: shortNameRus,
      comment: comment,
      description: description
    })).subscribe( next: response => {
        this.getAll();
      }
    );
  }
}
```

Рисунок 3.13 – Методы компонента

Для компонента mut-effect сверстана страница, что бы через интерфейс пользователь мог использовать ранее созданные методы. На рисунке 3.14 находится фрагмент html-кода компонента.

Рисунок 3.14 – Html-код страницы компонента

На строке 47 была добавлена кнопка в которой есть строка «(click)», это angular’овская функция, которая выполняется, при нажатии на кнопку. Т.е. при нажатии на эту кнопку вызывается метод add() из компонента, в который передаются значения из полей ввода, после чего будет создана новая запись в таблице с этими значениями. В строке 83 также присутствует специальная функция фреймворка, она подобна циклу for each. В данном случае, переменная “me” будет принимать все значения переменной mutEffect и записывать их в строки таблицы. Кнопка на 92 строке при нажатии удаляет эту запись, вызывая метод del из компонента.

На рисунке 3.15 представлена страница компонента.

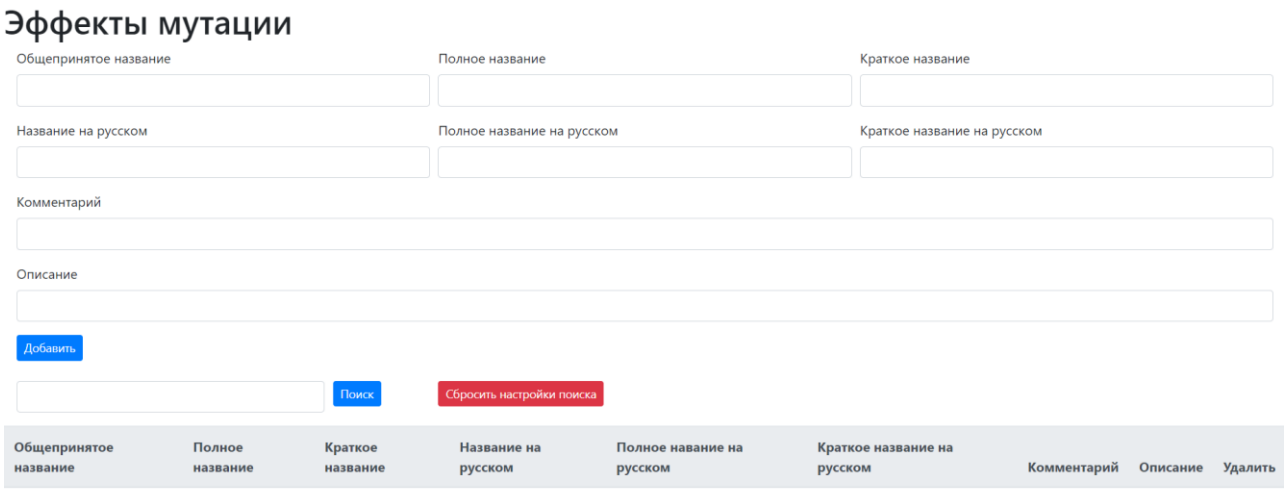


Рисунок 3.15 – Html-страница компонента

Остальные сервисы, компоненты и страницы выглядят практически аналогично, с разницей лишь в том, что у таблиц другие поля.

3.3 Тестирование разработанного приложения

После создания всех модулей приложения, было выполнено его тестирование. Так как для выполнения операций на клиентской части используется серверная часть, то в тестировании приведены примеры работы только клиентской части, потому что если серверная часть не работает, то и на клиентской части не будут выполняться операции.

На странице для перехода между всеми таблицами есть панель навигации, на которой есть выпадающие списки: словари и таблицы. В списках находятся ссылки для перехода к таблицам. На рисунке 3.16 представлена список словарей.

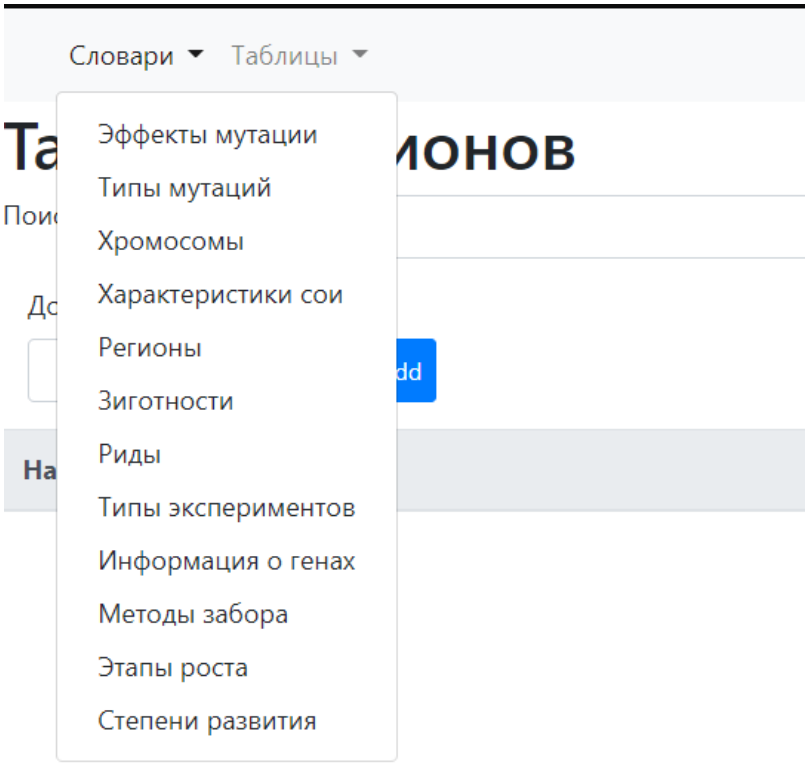


Рисунок 3.16 – Список словарей

На рисунке 3.17 показан список таблиц

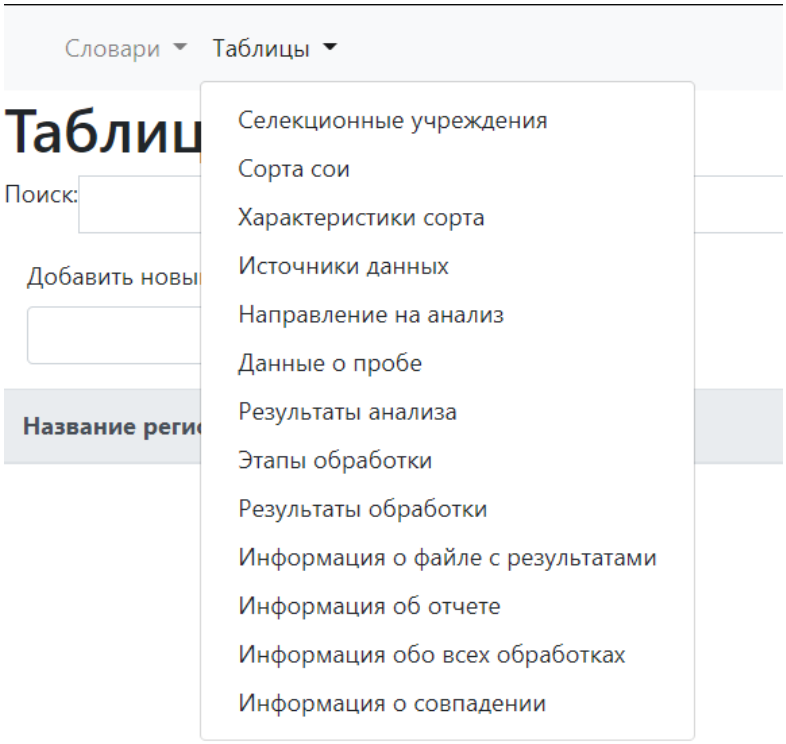


Рисунок 3.17 – Список таблиц

Для начал была выбрана таблица эффектов мутации из списка словарей. Данная страница представлена на рисунке 3.18.

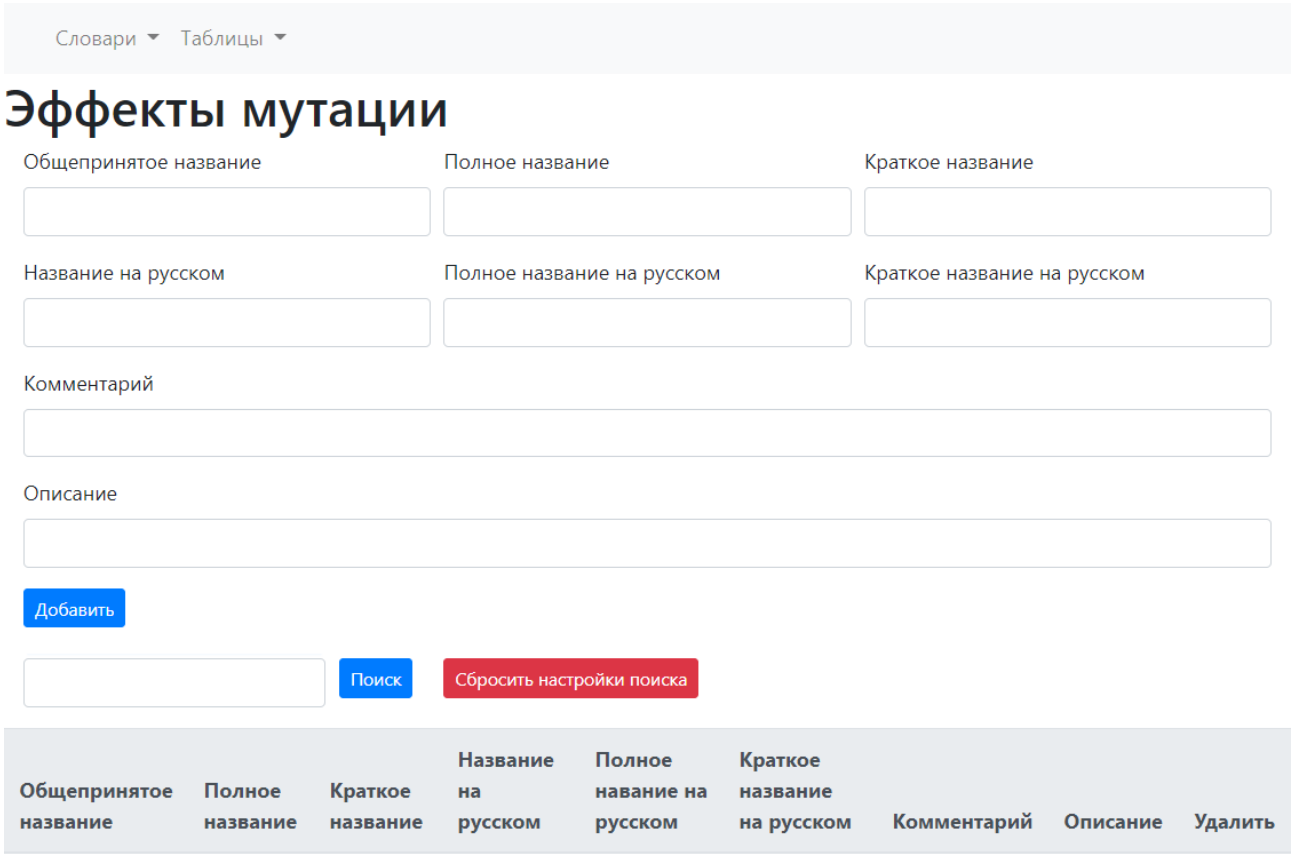


Рисунок 3.18 – Страница словаря «Эффекты мутации»

На странице есть поля для введения данных. Пример заполнения полей представлен на рисунке 3.19.

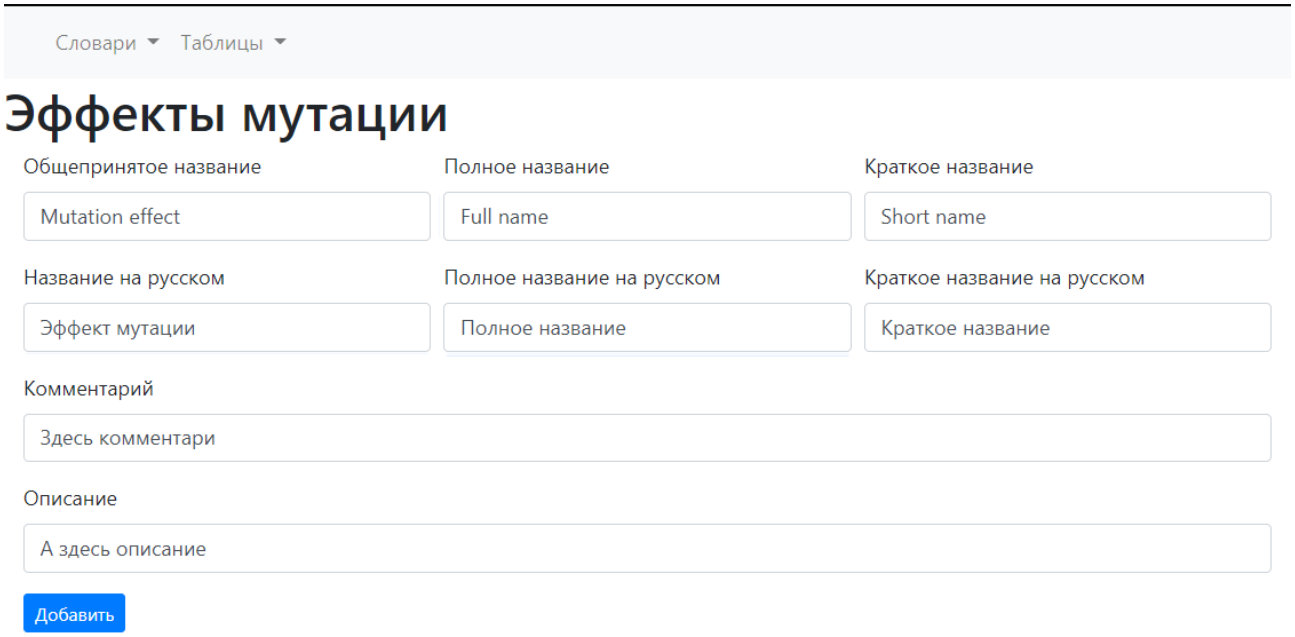
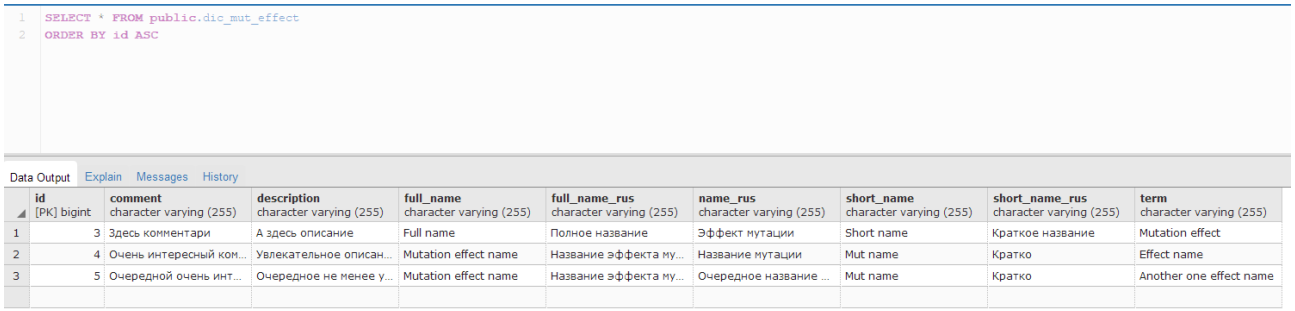


Рисунок 3.19 – Пример заполнения полей

После введения новых данных и нажатия кнопки «Добавить», эти данные были внесены в таблицу. Для наглядности выполнения операции в приложении pgAdmin можно посмотреть, есть ли в таблицы какие-либо записи (рис. 3.20).



```
SELECT * FROM public.dic_mut_effect
ORDER BY id ASC
```

Data Output | Explain | Messages | History

	id [PK] bigint	comment character varying (255)	description character varying (255)	full_name character varying (255)	full_name_rus character varying (255)	name_rus character varying (255)	short_name character varying (255)	short_name_rus character varying (255)	term character varying (255)
1	3	Здесь комментари	А здесь описание	Full name	Полное название	Эффект мутации	Short name	Краткое название	Mutation effect
2	4	Очень интересный ком...	Увлекательное описан...	Mutation effect name	Название эффекта му...	Название мутации	Mut name	Кратко	Effect name
3	5	Очередной очень инт...	Очередное не менее у...	Mutation effect name	Название эффекта му...	Очередное название ...	Mut name	Кратко	Another one effect name

Рисунок 3.20 – Записи в таблице

Отображение записей на странице представлено на рисунке 3.21

Поиск | Сбросить настройки поиска

Общепринятое название	Полное название	Краткое название	Название на русском	Полное навание на русском	Краткое название на русском	Комментарий	Описание	Удалить
Mutation effect	Full name	Short name	Эффект мутации	Полное название	Краткое название	Здесь комментари	А здесь описание	x
Another one effect name	Mutation effect name	Mut name	Очередное название мутации	Название эффекта мутации	Кратко	Очередной очень интересный комментарий	Очередное не менее увлекательное описание	x
Effect name	Mutation effect name	Mut name	Название мутации	Название эффекта мутации	Кратко	Очень интересный комментарий	Увлекательное описание	x

Рисунок 3.21 – Отображение записей на странице

Для удаления записи есть кнопка удаления, находящаяся в конце каждой записи. Для наглядности, что удаление выполняется, на рисунке 3.22 представлен отчет о совершенной операции удаления последней записи.

▼ General
Request URL: http://localhost:8080/dicMutEffect/4
Request Method: DELETE
Status Code: 204
Remote Address: [::1]:8080
Referrer Policy: no-referrer-when-downgrade

Рисунок 3.22 – Подтверждение операции удаления

Так же реализована операция поиска, пример работы которой представлен на рисунке 3.23.

Mutation effect | Поиск | Сбросить настройки поиска

Общепринятое название	Полное название	Краткое название	Название на русском	Полное навание на русском	Краткое название на русском	Комментарий	Описание	Удалить
Mutation effect	Full name	Short name	Эффект мутации	Полное название	Краткое название	Здесь комментари	А здесь описание	x

Рисунок 3.23 – Поиск по названию

Далее была выбрана таблица селекционных учреждений, которая представлена на рисунке 3.24

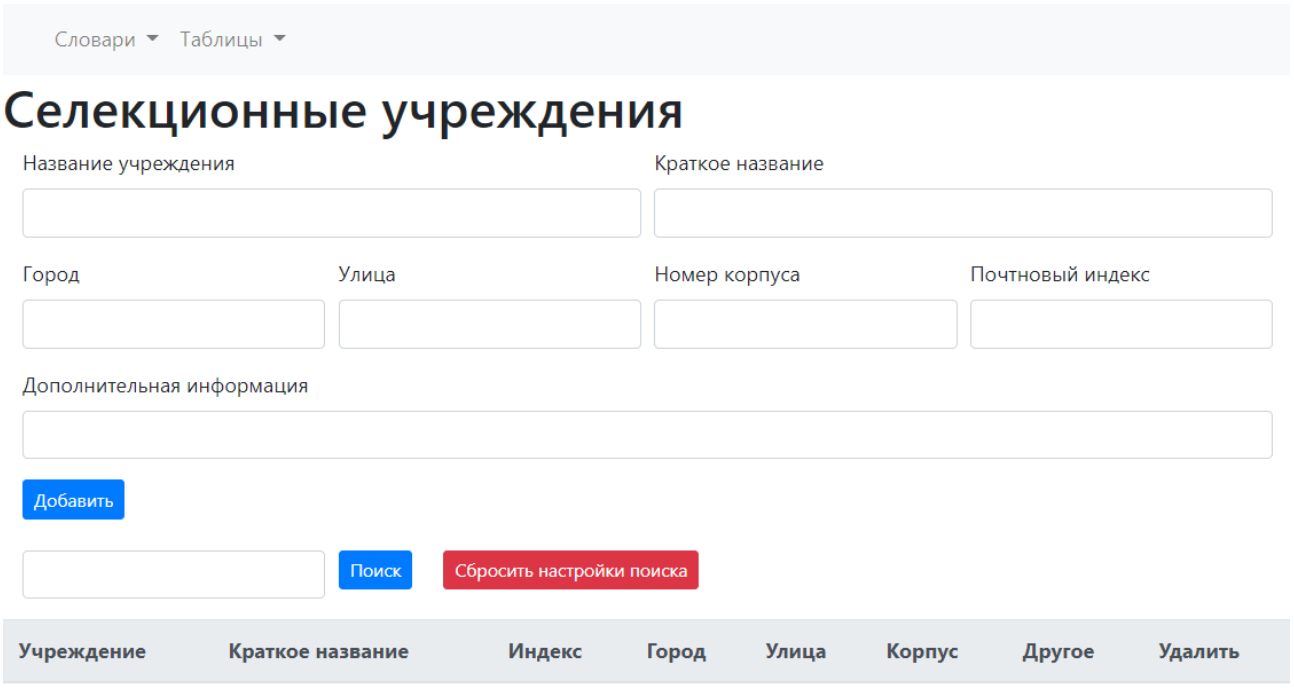


Рисунок 3.24 – Таблица «Селекционные учреждения»

Для данной таблице так же были сделаны поля для введения новых данных, на рисунке 3.25 показан пример введения новых данных.

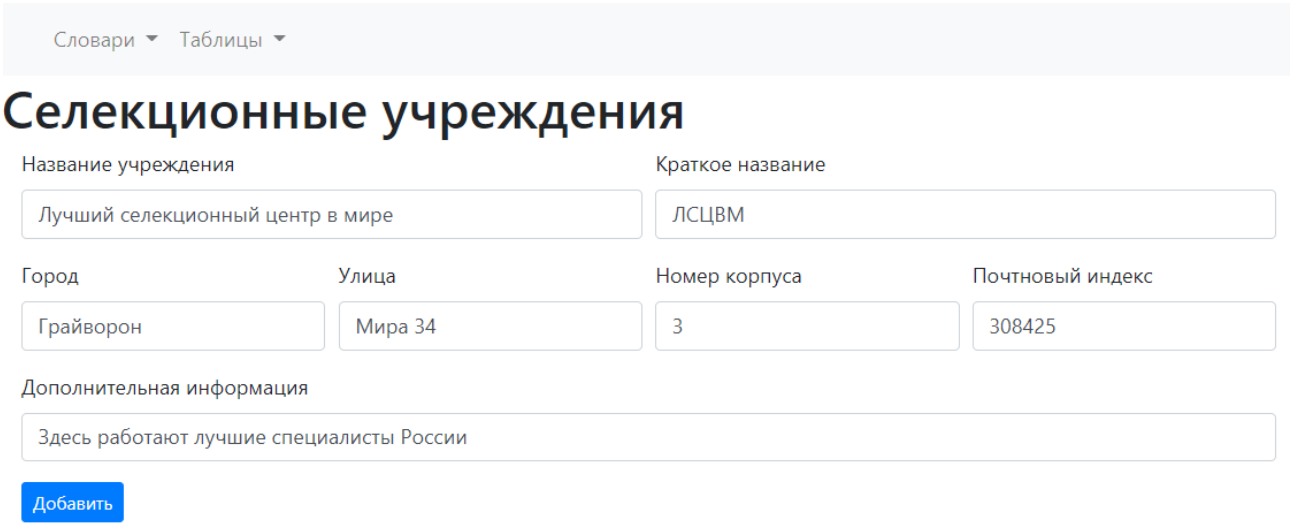


Рисунок 3.25 – Введения данных в таблицу

После добавления нескольких записей в таблицу, страница выглядит, как показано на рисунке 3.26

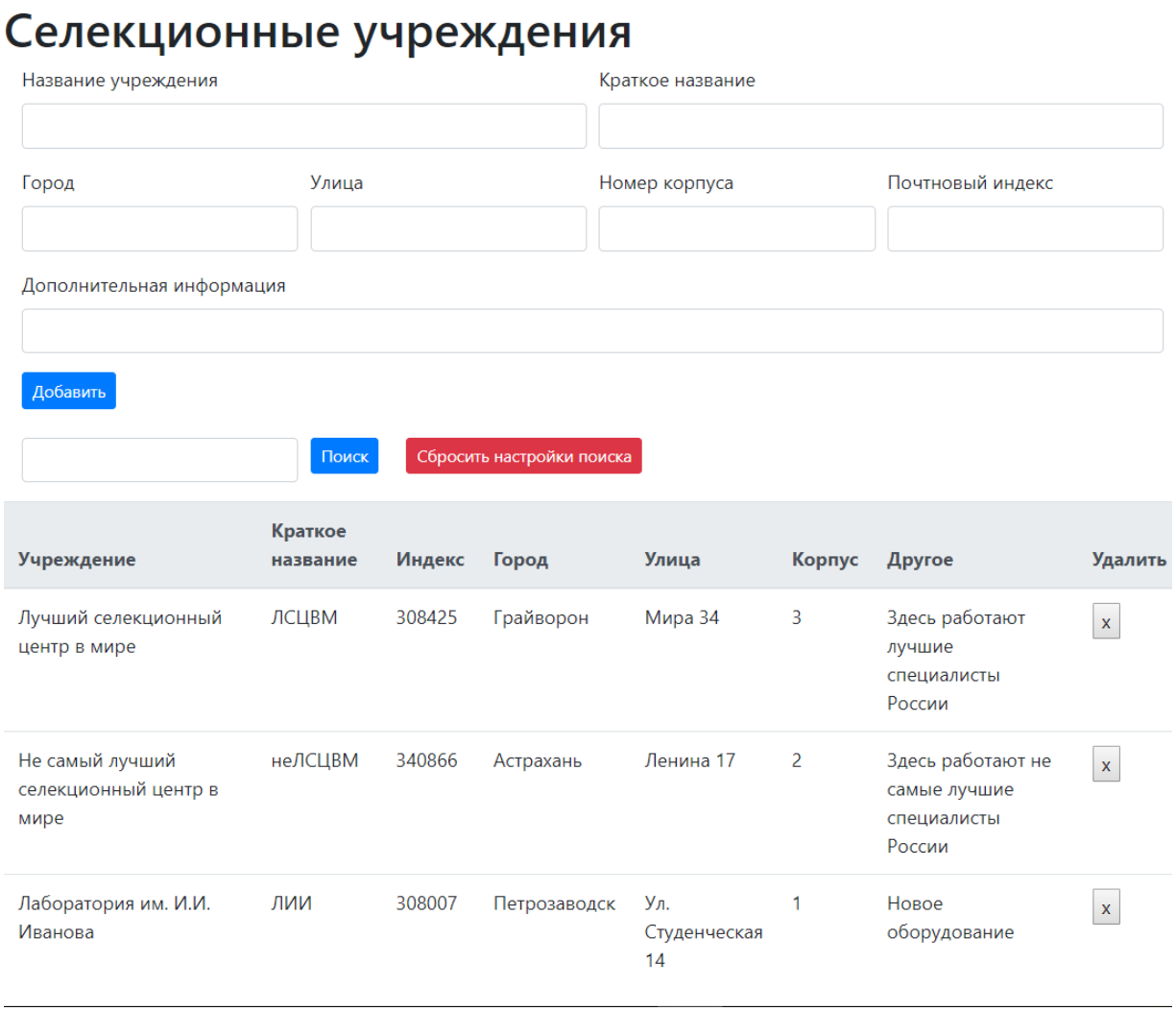


Рисунок 3.26 – Страница «Селекционные учреждения» с данными

Принцип работы с другими 38 таблицами аналогичен, за исключением отличий в названии полей. В следствии чего нецелесообразно приводить пример работы с другими таблицами.

3.4 Оценка эффективности проекта

После тестирования была проведена оценка эффективности проекта. Основной задачей является расчет временной эффективности. Также проведена стоимостная оценка затрат при использовании программного продукта.

Расчет показателя повышения производительности труда произведен по формуле 3.1:

$$P = (\frac{\Delta T}{F - \Delta T}) * 100, \quad (3.1)$$

где F – время, которое планировалось пользователем для выполнения работы до внедрения программ;

ΔT – экономия времени после внедрения подсистемы.

Таблица 3.1 – Оценка времени работы пользователей

№ п/п	Вид работ	Среднее время на операцию в неделю на одного сотрудника, минут		Экономия времени в неделю, минут ΔT	Повышение производительности труда, % P
		До автоматизации	После автоматизации		
1.	Ввод данных	150	100	50	50
2.	Выбор таблиц	50	20	30	150
3.	Просмотр данных	180	120	60	50
4.	Редактирование данных	120	60	60	100
5.	Поиск данных	160	100	60	60
ИТОГО		660	400	260	

При расчете сделаны следующие допущения:

– на пять типов операций, приведенных в таблице 3.1, каждый пользователь тратит 30% рабочего времени;

– фонд рабочего времени в неделю составляет 2 400 минут;

– все сотрудники проводят одинаковое время при работе с операциями.

Была рассчитана экономия времени работы пользователя, для этого значение ΔT разделили на показатель времени до автоматизации. Таким образом, экономия времени составила 39, 4%.

Трудовые затраты на разработку составили 168 часов или 21 рабочий дня при восьмичасовом рабочем дне. Среднемесячный фонд рабочего времени инженера 166,25 часов, среднемесячная заработная плата 20000 рублей.

Расчет основной заработной платы ($З_{осн}$) производился по формуле 3.2.

$$З_{осн} = \frac{З_{ср}}{Ф_{ср}} * Ч \, , \qquad (3.2)$$

где $З_{ср}$ – среднемесячная заработная плата;

$Ф_{ср}$ – среднемесячный фонд рабочего времени;

$Ч$ – это количество отработанных часов.

В соответствии с формулой 3.2 основная заработная плата инженера составила 20210 рублей.

В соответствии с Федеральный законом от 24 июля 2009 года N 212-ФЗ "О страховых взносах в Пенсионный фонд РФ, Фонд социального страхования РФ, Федеральный фонд обязательного медицинского страхования и территориальные фонды обязательного медицинского страхования" (в редакции Федерального закона от 03.12.2011 № 379-ФЗ) страховой взнос в составляет 30% от дохода, который вычисляется по формуле 3.3:

$$СВ = З_{осн} * Р_{св} \, , \qquad (3.3)$$

где $З_{осн}$ – основная заработная плата;

$Р_{св}$ – размер страхового взноса на социальные нужды.

Итоговые отчисления на социальные нужды составили 6063 рублей, а основная заработная плата разработчика с учетом отчислений – 14147 руб.

Стоимостная оценка затрат при использовании программного продукта рассчитали по формуле 3.4.

$$Р_{г} = (З_{да} – З_{авт}), \qquad (3.4)$$

где $З_{да}$ – затраты на обработку информации до автоматизации, руб/год;

$З_{авт}$ – затраты на автоматизированную обработку информации, руб/год.

Затраты на обработку информации до автоматизации вычислялись по формуле 3.5:

$$З_{да} = V_р * Ц_ч \quad (3.5)$$

где $V_р$ – время, затрачиваемое на обработку информации вручную, ч/год;

$Ц_ч$ – цена 1 ч работы, руб/год.

Годовые затраты на 5 рабочих мест за год (12 месяцев) при ручной обработке информации составляют 44 ч в месяц:

$З_{да} = 44* 12* 84,2*5= 222\ 310$ *руб*.

Затраты на автоматизированную обработку информации вычислялись по формуле 3.6:

$$З_{авт} = V_а * Ц_ч \quad (3.6)$$

где $V_а$ – затраты времени на автоматизированную обработку информации, руб/год.

При автоматизации затраты времени – 26,7 ч в месяц:

$З_{авт} = 26,7*12*84,2*5= 134\ 888$ *руб.*

Годовой результат от внедрения программного продукта рассчитали по формуле 3.5:

$Р_г = З_{да} – З_{авт} = 222\ 310 – 134\ 888 = 87\ 422$ *руб*.

Таким образом, в третьем разделе была проведена разработка серверной и клиентской части приложения, проведено тестирование разработанного приложения и проведена оценка эффективности проекта.

ЗАКЛЮЧЕНИЕ

Были выявлены и проанализированы требования, предъявляемые к информационному сопровождению. Всесторонне изучив деятельность ГК «ЭФКО» и, в частности, деятельность Инновационного центра «Бирюч – НТ» и его селекционного отдела, был выполнен структурно-функциональный анализ деятельности селекционного отдела. Был выбран ряд программных продуктов, для разработки информационного сопровождения.

Исходя из выявленных требований, была смоделирована и разработана база данных, после чего было разработано веб-приложение для ведения базы генетических и фенотипических данных сои для селекционного отдела инновационного центра «Бирюч – НТ». Приведен пример работы с разработанным приложением.

Данный программный продукт разработан непосредственно под особенности предметной области. Интерфейс приложения достаточно прост и интуитивно понятен, поэтому в процессе работы сотрудника селекционного отдела с данной системой не требуется специальных знаний в области проектирования информационных систем.

Приложение разработано с учетом того, что в дальнейшем может потребоваться его доработка или добавление новых модулей. Для этого серверная и клиентская часть являются отдельными приложениями, а доступ к данным разработан по архитектурному стилю REST, что обеспечивает возможность получать данные любым приложениям.

Все поставленные задачи были выполнены

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Хорстман К., Корнелл Г. Java. Библиотека профессионала. Том 1. Основы [Текст] / К. Хорстман, Г. Корнелл – Вильямс, 2017, – 864 с.

2. Хорстман К., Корнелл Г. Java. Библиотека профессионала. Том 2 Расширенные средства программирования [Текст] / К. Хорстман, Г. Корнелл – Вильямс, 2017 – 972 с.

3. Официальная документация фреймворка Spring [Электронный ресурс] – Режим доступа: http://docs.spring.io/spring/docs/current/spring-framework-reference, свободный.

4. Уоллс К. Spring в действии [Текст] / К. Уоллс – ДМК-Пресс, 2015 – 752 с.

5. Уорбэртон Р. Лямбда-выражения в Java 8 [Текст] / Р. Уорбэртон – ДМК Пресс, 2014 – 194 с.

6. Сайт формата данных JSON [Электронный ресурс] – Режим доступа: https://www.json.org, свободный.

7. Официальная документация СУБД PostgreSQL [Электронный ресурс] – Режим доступа: https://postgresql.ru/docs/postgresql/9.6, свободный.

8. Ричардсон Л. RESTful Web APIs: Services for a Changing World [Текст] / Л. Ричардсон – O'Reilly Media, 2013 – 406 с.

9. Лутова Л.А., Биотехнология высших растений [Текст] / Л.А. Лутова, 2010 – 240 с.

10. Финансовый словарь [Электронный ресурс] – Режим доступа: dic.academic.ru/contents.nsf/fin_enc, свободный.

11. Сайт Инновационного центра «Бирюч» [Электронный ресурс] – Режим доступа: http://biruch.ru, свободный.

12. Официальный сайт ГК «ЭФКО» [Электронный ресурс] – Режим доступа: www.efko.ru/o-kompanii, свободный.

13. Официальный сайт фреймворка Angular [Электронный ресурс] – Режим доступа: http://angular.io, свободный.

14. Зыков С. Основы проектирования информационных систем [Текст] / С. Зыков – Издательство дом Высшей школы экономики, 2012 – 237с.

15. Официальный сайт языка TypeScript [Электронный ресурс] – Режим доступа: http://www.typescriptlang.org, свободный.

16. Официальный сайт Hibernate [Электронный ресурс] – Режим доступа: http://hibernate.org/orm/, свободный.

17. Официальный сайт среды разработки IntelliJ IDEA [Электронный ресурс] – Режим доступа: https://www.jetbrains.com/idea/, свободный.

16. Бегг, К.Т. Базы данных. Проектирование, реализация и сопровождение. Теория и практика. [Текст] / К.Т. Бегг. – Екатеринбург: Издательство АРМ, 2016. – 345с.

17. Корнеев, В.В. Базы данных. Интеллектуальная обработка информации. [Текст] / В.В. Корнеев, А.Ф. Гареев, С.В. Васютин. – Москва: Нолидж, 2015. – 258с.

18. Голицына, О.Л. Базы данных [Текст] / О.Л. Голицына, Н.В. Максимов, И.И. Попов. - Москва: Форум, 2014. – 289с.

19. Финансовый словарь [Электронный ресурс] – Режим доступа: https://dic.academic.ru/contents.nsf/fin_enc/, свободный.

20. Советов, Б.Я. Базы данных: теория и практика. Общество с ограниченной ответственностью [Текст] / Б.Я. Советов, В.В. Цехановский, В.Д. Чертовской. – Москва: Издательство ЮРАЙТ, 2016. – 439с.

21. Уорсли Дж., Дрейк Дж. PostgreSQL. Для профессионалов. [Текст] / Дж. Уорсли, Дж. Дрейк. – Питер, 2013. – 496с.

22. Obe R., Hsu L. PostgreSQL: Up and Running. [Текст] / R. Obe, L. Hsu. – O'Reilly Media, 2014. – 234с.

23. Миронов, В.В. Ситуационно-ориентированные базы данных: концепция, архитектура, XML-реализация. [Текст] / В.В. Миронов, Н.И. Юсупова, Г.Р. Шакирова. – Уфа: Вестник, 2015. – 376с.

24. Маклаков, С.В. BPwin и ERwin. CASE-средства разработки информационных систем. [Текст] / С.В. Маклаков. – Москва: Диалог-МИФИ, 2017. – 412с.

25. Грекул, В.И. Проектирование информационных систем. [Текст] / В.И. Грекул, Г.Н. Денищенко, Н.Л. Коровкина. – Москва: Интернет-университет информационных технологий-ИНТУИТ, 2015.

26. Якимов, И.М. Комплексный подход к моделированию сложных систем в системе BPwin-Arena. [Текст] / И.М. Якимов. – Казань: Вестник Казанского технологического университета, 2014. – № 6.

27. Маклаков, С.В. Моделирование бизнес-процессов. [Текст] / С.В. Маклаков. – Москва: Диалог, 2016. – 258с.

28. Репин, В.В. Сравнительный анализ нотаций ARIS/IDEF и продуктов, их поддерживающих. [Электронный ресурс] / В.В. Репин. – Электрон. текстовые дан. – Москва, 2015. – Режим доступа: Web: http://www. iteam. ru/publications/it/section_51/article_2518, свободный.

29. Атисков, А.Ю. Автоматизированная система трансформации диаграмм бизнес-процессов в диаграммы классов. [Текст] / А.Ю. Атисков. – Самара: Издательства №3, 2016. – 155с.

30. Григорьев, А.В. Анализ существующих способов создания интерфейса «языки формальных спецификаций проблемно-ориентированные языки». [Текст] / А.В. Григорьев. – Москва: Паблик, 2017. – 325с.

31. Программирования Java. [Электронный ресурс] / Официальный сайт языка программирования Java. Электрон. журн., 2015. – Режим доступа: htt p://java. com, свободный.

32. Арнольд, К.Т. Язык программирования JAVA. [Текст] /К.Т. Арнольд, Д.П. Гослинг. – СПб.: Питер-Пресс, 2014. – 269с.

33. Васильев, А.Н. Java: объектно-ориентированное программирование: для магистров и бакалавров: базовый курс по объектно-ориентированному программированию. [Текст] / А.Н. Васильев. – СПб.: Издательский дом «Питер», 2015. – 513с.

34. Романов, В.Ю. Моделирование и верификация архитектуры программного обеспечения, разработанного на языке Java. Современные информационные технологии и ИТ-образование. [Текст] / В.Ю. Романов. – Москва: Издательский дом «Москва», 2014. – 178с.

35. Алексанян, Г.К. Анализ возможности применения языка программирования Java в задачах электроимпедансной томографии. [Текст] / Г.К. Алексанян, А.Д. Тарасов, К.В. Клевец. – Москва: Паблик, 2015. – 321с.

36. Allen, L. et al. Orbital angular momentum of light and the transformation of Laguerre-Gaussian laser modes. [Текст] / Allen L. – New-York: Publishing house, 2014. – 357с.

37. Edmonds, A.R. Angular momentum in quantum mechanics. [Текст] / A.R. Edmonds. – New Jersey: Princeton University Press, 2016. – 356с.

38. Williams, M.L. Stress singularities resulting from various boundary conditions in angular corners of plates in extension. [Текст] / M.L. Williams. – New York: Journal of applied mechanics, 2014. – 528с.

39. Mair, A. et al. Entanglement of the orbital angular momentum states of photons. [Текст] / A. Mair. – New Jersey: Nature, 2015. – 313с.

40. Wilczek, F. Magnetic flux, angular momentum, and statistics. [Текст] / F. Wilczek. – Canada: Physical Review Letters, 2016. –1144с.

41. Zubin, J. Vulnerability: a new view of schizophrenia. [Текст] / J. Zubin, B. Spring. – New York: Journal of abnormal psychology, 2016. – 103с.

ПРИЛОЖЕНИЕ А

Должностная инструкция ведущего биоинформатика

УТВЕРЖДАЮ:
Генеральный директор
ООО «ИЦ «Бирюч-НТ»
_____ Санина Т.В.
«____»_____ 2017 г.

ДОЛЖНОСТНАЯ ИНСТРУКЦИЯ

Ведущего биоинформатика Управления биотехнологических проектов обособленного подразделения с.Малобыково
ди _____

1. Общие положения

1.1. Настоящая должностная инструкция определяет функциональные обязанности, права и ответственность ведущего биоинформатика Управления биотехнологических проектов о.п.с. Малобыково (далее ведущий биоинформатик) общества с ограниченной ответственностью «Инновационный центр «Бирюч-новые технологии» (далее ООО «ИЦ «Бирюч-НТ», компания, предприятие, организация).

1.2. Ведущий биоинформатик относится к категории специалистов, назначается и освобождается от должности в установленном действующим трудовым законодательством порядке приказом Генерального директора ИЦ «Бирюч-НТ» на основании протоколов внутреннего Комитета по аттестации сотрудников ООО «ИЦ «Бирюч-НТ», внутреннего Комитета по оценке эффективности персонала ООО «ИЦ «Бирюч-НТ».

1.3. Ведущий биоинформатик непосредственно подчиняется заместителю генерального директора по биотехнологическим проектам.

1.4. На период отсутствия ведущего биоинформатика, его обязанности исполняет лицо, назначенное в установленном порядке, которое приобретает соответствующие права и несёт ответственность за ненадлежащее исполнение возложенных на него обязанностей.

1.5. На должность ведущего биоинформатика назначается лицо, имеющее высшее профессиональное образование, прошедшее профессиональную подготовку/переподготовку по направлению биоинженерия/биоинформатика и подтвердившее профессиональные компетенции в соответствии с требованиями, указанными в личной карточке ведущего биоинформатика.

1.6. Ведущий биоинформатик в своей деятельности руководствуется действующим законодательством, настоящей инструкцией, приказами и распоряжениями руководства, внутренними нормативными документами ООО «ИЦ «Бирюч-НТ».

1.7. Ведущий биоинформатик должен знать:

1.7.1 Генетику, биотехнологию, молекулярную биологию, генную инженерию, биохимию, микробиологию, селекцию и семеноводство, биоинформатику, основы программирования.

1.7.2 Основные принципы селекционно-генетической работы

1.7.3 Основные биологические базы данных и алгоритмы работы с ними NCBI (PubMed), UniPROT, Protein data bank (PDB).

1.7.4 Основные языки программирования: «R», Python, Perl, C++ и другие.

1.7.5 Нормы действующего законодательства, регламентирующего хозяйственную деятельность организации.

1.7.6 Технологию производства продукции Компании ЭФКО.

1.7.7 Документы, регламентирующие деятельность Компании ЭФКО.

1.7.8 Правила внутреннего трудового распорядка.

1.7.9 Стандарт документооборота Компании ЭФКО.

1.7.10 Основы деловой переписки и документооборота.

1.7.11 Этику делового общения.

1.7.12 Основы трудового законодательства.

1.7.13 Настоящую должностную инструкцию.

1.7.14 Инструкции по охране труда, норм и правил производственной санитарии и противопожарной защиты.

2. Должностные обязанности

Ведущий биоинформатик Управления биотехнологических проектов:

2.1 Осуществляет работу с базами данных биологической информации.

2.2 Осуществляет обработку больших объемов информации (последовательностей геномов, генов, фрагментов ДНК).

2.3 Занимается расчетом структуры и функций биологических молекул (аннотацией генов, регуляторных элементов и других структур).

2.4 Занимается множественным выравниванием последовательностей и поиском гомологичных генов с помощью прикладных компьютерных программ.

2.5 Занимается систематизацией биологической информации и разработкой алгоритмов работы с целью создания собственных баз данных.

2.6 Занимается составлением программы скрещивания растений с известными признаками и генами совместно с селекционером для получения новых сортов сои с заранее заданными признаками.

2.7 Осуществляет разработку программ для анализа данных о структуре геномов, а также для автоматического составления генетических карт скрещивания.

2.8 Занимается поиском, анализом, идентификацией генетических маркеров на основании данных о строении геномов.

2.9 Занимается разработкой и реализацией сложных алгоритмов проектирования геномов, а также созданием удобных инструментов (программных оболочек) для их использования.

2.10 Систематически изучает отечественные и зарубежные научно-технические достижения и передовой опыт в области генной инженерии и биоинформатики.

2.11 Участвует в совещаниях и комитетах в рамках своей компетенции.

2.12 Осуществляет написание статей, подготовку докладов по темам, актуальным для компании.

2.13 Своевременно предоставляет информацию в системе электронного документооборота компании (СЭДО), по соответствующим формам.

2.14 Формирует КПГ работ на месяц и долгосрочный период.

2.15 Подготавливает отчетную документацию о выполненных работах для предоставления на Комитеты, рассмотрение руководителю проекта, другим сотрудникам ООО «ИЦ «Бирюч-НТ» и других подразделений компании по согласованию с руководителем проекта.

2.16 Проходит периодический медицинский осмотр не реже 1 раз в год.

2.17 Соблюдает требования инструкций по охране труда, норм и правил промышленной и пожарной безопасности.

2.18 Соблюдает правила носки и хранения спецодежды, спецобуви, средств индивидуальной защиты.

2.19 Проходит обучение по охране труда и проверки знаний в комиссии предприятия не реже 1 раза в 3 года.

2.20 Обеспечивает конфиденциальность информации, сохранность коммерческой тайны. Последнее включает в себя весь комплекс обязанностей, связанных с информацией ограниченного доступа, а именно:

2.20.1 соблюдение требований по получению, обработке информации ограниченного доступа (передаче, хранению, получению и т.д.);

2.20.2 принятие мер по установлению и сохранению режима коммерческой тайны;

2.20.3 ограничение использования информации ограниченного доступа в целях, не связанных с осуществлением трудовой функции, без разрешения обладателя;

2.20.4 не разглашение информации ограниченного доступа, а также не совершение иных деяний, влекущих уничтожение или утрату информации ограниченного доступа или потерю ее коммерческой или иной ценности для обладателя.

2.21 Обязан знать структуру предприятия, направления деятельности ООО «ИЦ «Бирюч-НТ», в том числе финансовое и экономическое, содержание и методику заполнения личной карточки, а также существующие на предприятии механизмы оплаты труда и дополнительной мотивации работников. Для подтверждения квалификации обязан проходить аттестацию по вышеуказанным предметам, а также по дисциплинам, отраженным в личной карточке в качестве необходимых для занимаемой должности.

3. Права

Ведущий биоинформатик имеет право:

3.1 Вносить на рассмотрение руководства предложения по совершенствованию работы, связанной с предусмотренными настоящей инструкцией обязанностями.

3.2 Участвовать в рассмотрении вопросов, связанных с выполнением должностных обязанностей.

3.3 В пределах своей компетенции сообщать обо всех выявленных в процессе своей деятельности недостатках и вносить предложения по их устранению.

3.4 Запрашивать от подразделений информацию и документы, необходимые для выполнения должностных обязанностей.

3.5 Вносить предложения по плановому повышению квалификации управления и участвовать в программах компании по развитию персонала.

3.6 Представлять интересы проекта в сторонних учреждениях и организациях по вопросам, относящимся к его функциональным обязанностям.

3.7 Представлять материалы по вопросам, требующим вмешательства руководства подразделения или предприятия, включая предложения о поощрении или взыскании.

3.8 Принимать решения по конкретным вопросам в рамках своей компетенции.

3.9 Получать необходимую информацию, оборудование, материалы.

3.10 Утверждать, подписывать или визировать документы в рамках своей компетенции.

3.11 Докладывать непосредственному руководителю о положении дел и выполнении поручений.

3.12 Обжаловать неправомочные действия непосредственного руководителя в отношении себя у вышестоящего руководства и представителя профсоюза.

3.13 Рабочее место, соответствующее требованиям охраны труда.

3.14 Обязательное социальное страхование от несчастных случаев на производстве и профессиональных заболеваний в соответствии с законодательством.

3.15 Получение достоверной информации об условиях охраны труда на рабочем месте, о существующем риске повреждения здоровья, а также о мерах по защите от воздействия вредных и (или) опасных производственных факторов.

3.16 Обеспечение средствами индивидуальной и коллективной защиты в соответствии с требованиями охраны труда.

3.17 Обучение безопасным методам и приемам труда.

3.18 Личное или через своих представителей участие в рассмотрении вопросов, связанных с обеспечением безопасных условий труда на рабочем месте.

3.19 Периодический медицинский осмотр (обследование) в соответствии с медицинскими рекомендациями.

4. Ответственность

Ведущий биоинформатик несёт ответственность за:

4.1. Ненадлежащее исполнение и неисполнение своих должностных обязанностей, предусмотренных настоящей должностной инструкцией, в пределах, определенных действующим трудовым законодательством Российской Федерации.

4.2 Причинение материального ущерба в пределах, определенных действующим трудовым и уголовным законодательством Российской Федерации.

4.3 Правонарушения, совершенные в процессе своей деятельности в пределах, установленных действующим административным, уголовным и гражданским законодательством Российской Федерации.

4.4 Разглашение умышленное или по неосторожности конфиденциальной информации, нарушение обязанностей по обеспечению сохранности коммерческой тайны, секретов производства.

4.5 Несоблюдение системы нормативного регулирования и внутреннего трудового распорядка ООО «ИЦ «Бирюч-НТ».

4.6 Несоблюдение требований трудовой и производственной дисциплины, правил техники безопасности и охраны труда, пожарной безопасности; промышленной санитарии.

4.7 Несоблюдение правил обращения и сохранности документов с грифом «для служебного пользования» и «конфиденциально».

4.8 Невыполнение приказов, распоряжений и указаний руководства предприятия.

5. Условия работы

5.1 Режим работы ведущего биоинформатика определяется в соответствии с Правилами внутреннего трудового распорядка ООО «ИЦ «Бирюч-НТ».

5.2 В связи с производственной необходимостью ведущий биоинформатик может направляться в служебные командировки, в т.ч. местного значения.

5.3 В целях предотвращения нарушения работоспособности программного обеспечения персональных компьютеров и рационального использования рабочего времени ведущему биоинформатику запрещается хранение и использование в персональных компьютерах программ и файлов, не относящихся к работе (музыкальных и видео файлов, игр, развлекательных программ, почты личного характера и фотоизображений личного характера).

ПРИЛОЖЕНИЕ Б

Код программы

```
import lombok.Data;
import lombok.NoArgsConstructor;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
@Entity
@Data
@NoArgsConstructor
public class DicMutEffect {
    @Id
    @GeneratedValue (strategy = GenerationType.AUTO)
    private long id;
    private String term; // Название мутации на английском
    private String fullName; // Полное название мутации на английском языке
    private String shortName;
    private String nameRus; // Название мутации на русском
    private String fullNameRus;
    private String shortNameRus;
    private String comment; // Внутренний  комментарий
    private String description; // Описание
}
import com.db.Tables.Dictionary.DicMutEffect;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.repository.query.Param;
import org.springframework.data.rest.core.annotation.RepositoryRestResource;
import org.springframework.web.bind.annotation.CrossOrigin;
import java.util.List;
@CrossOrigin
@RepositoryRestResource (path = "dicMutEffect")
public interface DicMutEffectRepository extends JpaRepository<DicMutEffect,
Long>{
    List<DicMutEffect> findByTerm(@Param("term")String term);
    List<DicMutEffect> findByFullName(@Param("fullName")String fullName);
    List<DicMutEffect> findByShortName(@Param("shortName")String shortName);
    List<DicMutEffect> findByNameRus(@Param("nameRus") String nameRus);
    List<DicMutEffect> findByFullNameRus(@Param("fullNameRus") String
fullNameRus);
    List<DicMutEffect> findByShortNameRus(@Param("shortNameRus") String
shortNameRus);

import com.db.Tables.Dictionary.DicChromosome;
import com.db.Tables.Dictionary.DicMutEffect;
import com.db.Tables.Dictionary.DicMutType;
import com.db.Tables.Dictionary.DicZygocity;
import lombok.Data;
import lombok.NoArgsConstructor;
import javax.persistence.*;
@Entity
@Data
@NoArgsConstructor
public class Mutation {
    @Id
    @GeneratedValue (strategy = GenerationType.AUTO)
    private long id;
    @ManyToOne
```

```
    private DicMutEffect mutEffect;
    @ManyToOne
    private DicMutType mutType;
    @ManyToOne
    private DicZygocity zygocity;
    @ManyToOne
    private DicChromosome chrom;
    private String sbSyntax; // Кодировка мутации в соответствии с номенклатурой
SoyBase
    private int alleleNumber; // Колличество аллелей
    private long sbCount; // Id мутации из базы SoyBase
    private String comment; // Внутренний комментарий
}
import com.db.Tables.Mutation;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.repository.query.Param;
import org.springframework.data.rest.core.annotation.RepositoryRestResource;
import java.util.List;
@RepositoryRestResource(path = "mutation")
public interface MutationRepository extends JpaRepository <Mutation, Long>{
    List<Mutation> findBySbSyntax(@Param("sbSyntax") String sbSyntax);
    List<Mutation> findBySbCount(@Param("sbCount")Long sbCount);
}


import {Injectable} from '@angular/core';
import {HttpClient} from '@angular/common/http';
import {Observable} from 'rxjs/index';
import {MutEffect} from '../models/mut-effect';
import {map} from 'rxjs/internal/operators';
@Injectable()
export class MutEffectService {
  constructor(private http: HttpClient) {}
  url = 'http://localhost:8080/dicMutEffect';
  searchUrl = 'http://localhost:8080/dicMutEffect/search/';
  public getMutEffect(): Observable<MutEffect[]> {
    return this.http.get<MutEffect[]>(this.url)
      .pipe(map(response => {
        return <MutEffect[]>response['_embedded']['dicMutEffects'].map(mutEffect
=> {
          return new MutEffect({
            term: mutEffect.term,
            fullName: mutEffect.fullName,
            shortName: mutEffect.shortName,
            nameRus: mutEffect.nameRus,
            fullNameRus: mutEffect.fullNameRus,
            shortNameRus: mutEffect.shortNameRus,
            comment: mutEffect.comment,
            description: mutEffect.description,
            href: mutEffect['_links']['self'].href
          });
        });
      }));
  }
  public addMutEffect(me: MutEffect): Observable<MutEffect> {
    return this.http.post<MutEffect>(this.url, me)
      .pipe(map(mutEffect => {
        return new MutEffect({
          term: mutEffect.term,
          fullName: mutEffect.fullName,
          shortName: mutEffect.shortName,
          nameRus: mutEffect.nameRus,
          fullNameRus: mutEffect.fullNameRus,
```

```
          shortNameRus: mutEffect.shortNameRus,
          comment: mutEffect.comment,
          description: mutEffect.description,
          href: mutEffect['_links']['self'].href
        });
      }));
  }
  public deleteMutEffect(href: string): Observable<any> {
    return this.http.delete(href);
  }
  public searchByTerm(term: string): Observable<MutEffect[]> {
    return this.http.get<MutEffect[]>(this.searchUrl + 'findByTerm?term=' +
term)
      .pipe(map(response => {
        return <MutEffect[]>
response['_embedded']['dicMutEffects'].map(mutEffect => {
        return new MutEffect({
          term: mutEffect.term,
          fullName: mutEffect.fullName,
          shortName: mutEffect.shortName,
          nameRus: mutEffect.nameRus,
          fullNameRus: mutEffect.fullNameRus,
          shortNameRus: mutEffect.shortNameRus,
          comment: mutEffect.comment,
          description: mutEffect.description,
          href: mutEffect['_links']['self'].href
        });
        });
      }));
  }
}
import { Component, OnInit } from '@angular/core';
import {MutEffectService} from '../service/mut-effect.service';
import {MutEffect} from '../models/mut-effect';

@Component({
  selector: 'app-mut-effect',
  templateUrl: './mut-effect.component.html',
  styleUrls: ['./mut-effect.component.css']
})
export class MutEffectComponent implements OnInit {
  mutEffect: MutEffect[];

  constructor(private mutEffectService: MutEffectService) { }

  ngOnInit() {
    this.getAll();
  }

  getAll(): void {
    this.mutEffectService.getMutEffect()
      .subscribe(response => {
        this.mutEffect = response;
      });
  }

  getByTerm(term: string): void {
    this.mutEffectService.searchByTerm(term)
      .subscribe(response => {
        this.mutEffect = response;
      });
  }
```

```
  del(href: string): void {
    this.mutEffectService.deleteMutEffect(href).subscribe(() => {
      this.getAll();
    });
  }

  add(term: string, fullName: string, shortName: string, nameRus: string,
      fullNameRus: string, shortNameRus: string, comment: string, description:
string): void {
    this.mutEffectService.addMutEffect(new MutEffect({
      term: term,
      fullName: fullName,
      shortName: shortName,
      nameRus: nameRus,
      fullNameRus: fullNameRus,
      shortNameRus: shortNameRus,
      comment: comment,
      description: description
    })).subscribe(response => {
        this.getAll();
      }
    );
  }
}
```