УТВЕРЖДАЮ

Генеральный директор
АО «Айкумен ИБС»

_____ А. В. Буланов

____ _____ 2026 г.

Программа для ЭВМ «IQTrainer»

Инструкция по установке и настройке

86447414.62.01.206.И2

Листов 40

2026

Подп. и дата

Инв. № дубл.

Взам. инв. №

Подпись и дата

Инв. № подл.

Аннотация

Настоящий документ содержит описание технических требований, процесса установки и конфигурирования «Программа для ЭВМ «IQTrainer»» (далее – Программа).

Программа для ЭВМ «IQTrainer» предназначена для обучения нейросетевых моделей классификаторов текстов и картинок, подготовки и разметки датасетов, а также оценки качества обученных моделей.

Содержание

Аннотация 2

Перечень терминов и сокращений 5

1 Общее описание программы 6

1.1 Назначение программы 6

1.1.1 Наименование программы 6

1.1.2 Архитектура программы 6

1.1.3 Перечень функций, реализуемых программой 12

1.1.4 Модель данных программы 16

1.2 Состав программных средств 18

1.3 Комплект аппаратных и технических средств 19

1.4 Установка ПО на технических средствах 20

2 Установка и настройка программы 21

2.1 Общий порядок установки 21

2.2 Установка системного и прикладного ПО 21

2.2.1 Базовая настройка CentOS 7 minimal 21

2.2.2 Дополнительная настройка серверов комплекса 23

2.2.3 Установка и настройка СУБД PostgreSQL 25

2.3 Установка приложения **IQTrainer** 26

2.3.1 Общие положения 26

2.3.2 Настройка аутентификации пользователей 29

2.3.3 Настройка работы с картинками 29

2.4 Поддержка LDAP 30

2.5 Поддержка поиска похожих в Milvus и PGVECTOR 30

2.5.1 Настройка интеграции с Milvus 31

2.5.2 Настройка интеграции с PGVECTOR 31

2.6 Установка нейросетевого модуля для текстов 33

2.7 Установка нейросетевого модуля для картинок 34

2.8 Установка расширения для браузера (для загрузки картинок) 34

2.9 Сопряжение приложения IQTrainer с нейросетевыми модулями 35

3 Администрирование программы 37

3.1 Остановка и запуск 37
3.1.1 Общий порядок запуска 37
3.1.2 Общий порядок остановки 37
3.1.3 Запуск и остановка программных модулей 37
3.2 Диагностика работоспособности 39

Перечень терминов и сокращений

АРМ	–	автоматизированное рабочее место
ОС	–	операционная система
Браузер	–	программное обеспечение для просмотра содержимого страниц веб-сайтов, их обработки, вывода и перехода от одной страницы к другой
Группа пользователей	–	группа пользователей, предназначена для управления правами доступа пользователей к проектам и функциям Программы
Датасет	–	набор документов, который может быть использован для обучения модели либо оценки качества её обучения (тестирования)
Документ	–	это текстовый документ либо картинка (файл графического формата), которые могут быть загружены в проект и привязаны к определённому датасету
Источник	–	конкретный программно-аппаратный комплекс, построенный на базе Платформы IQPLATFORM, который может быть использован для загрузки документов при создании датасетов
Классификатор	–	набор классов, по которым производится классификация документов – текстовых документов либо картинок
Модель	–	нейросетевая модель, которая обучена классификации документов по классам определённого классификатора
Нейросетевой модуль	–	программный модуль в составе Программы, предназначенный для обучения и хранения обученных моделей, а также для реализации сервиса потоковой кластеризации документов для программно-аппаратных комплексов, построенных на базе Платформы IQPLATFORM
Пользователь	–	учётная запись пользователя Программы
Привилегия	–	право доступа к некоторой функции Программы. Привилегии назначаются на группы пользователей
Приложение IQTrainer	–	приложение IQTrainer, основной программный модуль в составе Программы, реализующий её основную бизнес-логику и предназначенный для работы пользователей
Проект	–	сущность, предназначенная для группировки и хранения совокупности данных, используемых на одном проекте для обучения классификаторов и оценки качества обучения
Программа	–	программа для ЭВМ «IQTrainer»
Тип модели	–	математическая архитектура нейросетевой модели

1 Общее описание программы

1.1 Назначение программы

1.1.1 Наименование программы

Полное наименование программного комплекса: Программа для ЭВМ «IQTrainer» (далее – Программа).

Сокращённо – «**IQTrainer**».

Программа для ЭВМ «IQTrainer» предназначена для обучения нейросетевых моделей классификаторов текстов и картинок, подготовки и разметки датасетов, а также оценки качества обученных моделей.

1.1.2 Архитектура программы

В состав программы входят три программных модуля (рис. 1):

– Приложение IQTrainer, основной программный модуль программы, реализующий её основную бизнес-логику и предназначенный для работы пользователей. Приложение написано на языке Java с использованием библиотеки Spring Boot. В качестве веб-сервера используется интегрированная в Spring версия Apache Tomcat. Приложение поставляется в виде образа Docker.

Расшифровка образа докер:

– образ докер для IQTrainer создан на базе образа *eclipse-temurin:21-jdk-alpine,* использующий Linux alpine-3.21;

– образ докер для нейромодуля картинок создан на базе образа *nvidia/cuda:11.1.1-runtime-ubuntu20.04* – использующий Linux ubuntu-20.04;

– образ докер для нейромодуля текстов создан на базе образа *nvidia/cuda:12.1.0-devel-ubuntu20.04* – использующий Linux ubuntu-20.04.

– Текстовый нейросетевой модуль, обеспечивающий работу нейросетевых моделей разного типа, предназначенных для векторизации текстов с последующей их классификацией. Модуль написан на Python 3, для корректной работы требует наличия видеокарты. Поставляется в виде образа Docker. Если инсталляция IQTrainer не предполагает обучение текстовых классификаторов, то данный модуль можно не инсталлировать.

– Нейросетевой модуль картинок, обеспечивающий работу нейросетевых моделей разного типа, предназначенных для векторизации картинок с последующей их классификацией. Модуль написан на Python 3, для корректной работы требует наличия видеокарты. Поставляется в виде образа Docker. Если инсталляция IQTrainer не предполагает обучение классификаторов картинок, то данный модуль можно не инсталлировать.

– Специальное расширение IQTrainer image marker для браузера Google Chrome, предназначенное для загрузки найденных на веб-сайтах картинок в IQTrainer. В проектах, связанных с разметкой текстовой информации, не используется и может быть не инсталлировано.

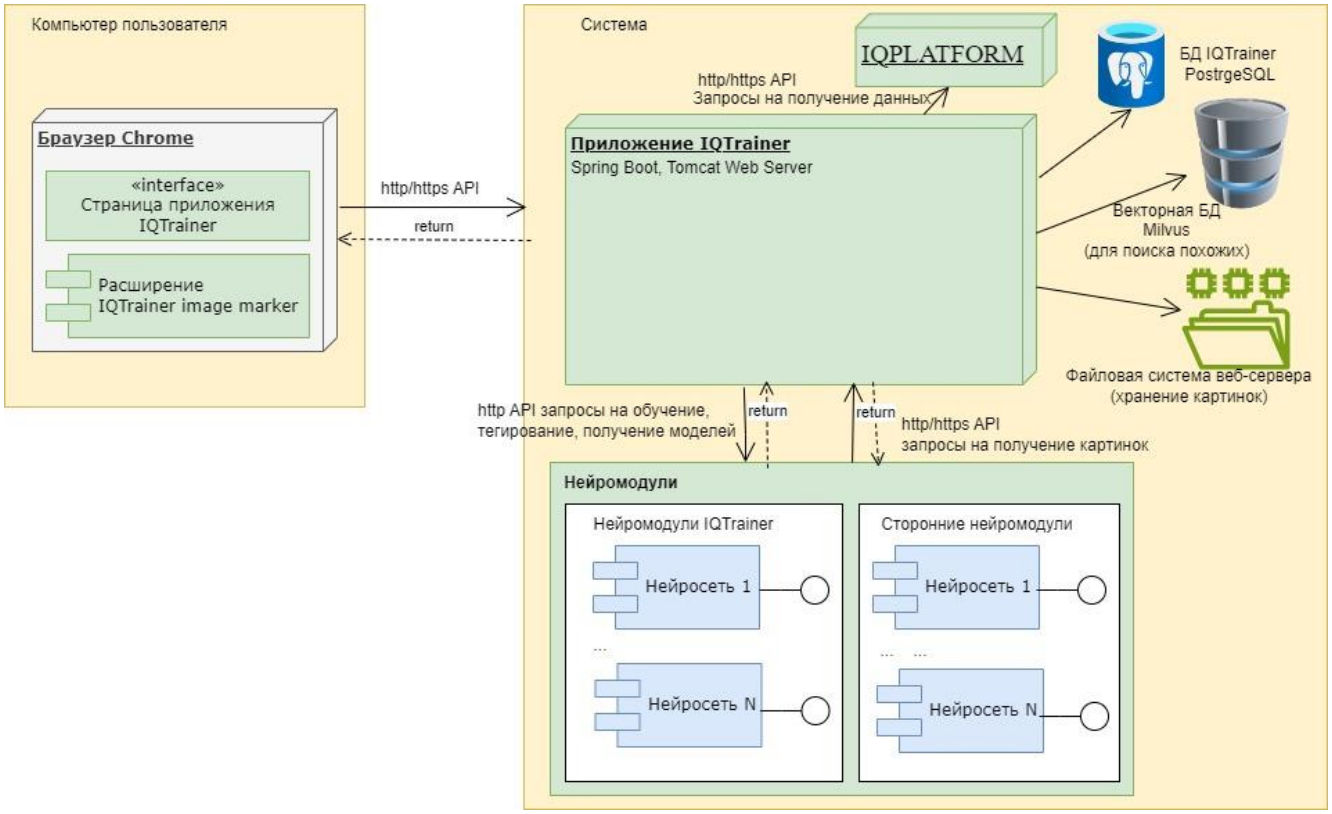

Рис. 1 – Архитектурная схема программы

Приложение IQTrainer представляет собой веб-приложение, доступ пользователей к интерфейсам которого осуществляется с помощью браузера Google Chrome актуальной версии.

Дополнительно приложение IQTrainer взаимодействует со следующим сторонним программным обеспечением.

– С программно-аппаратным комплексом на базе Платформы IQPLATFORM для возможности загрузки в программу текстовых документов либо картинок при формировании датасетов – посредством формирования HTTP-запросов на поиск

документов по заданным параметрам. Если в рамках проекта не предполагается интеграции с платформой IQPLATFORM, то данный функционал не задействуется.

– Веб-сервер IQTrainer может взаимодействовать с несколькими нейросетевыми модулями. Работа в IQTrainer ведётся в проектных пространствах. Рекомендуется под каждый проект разворачивать свою нейросеть. Взаимодействие веб-сервера IQTrainer с нейросетевыми модулями производится через API, IQTrainer может работать как собственными нейросетевыми модулями, так и сторонними нейросетевыми модулями. Сторонние нейросетевые модули в состав программы не входят.

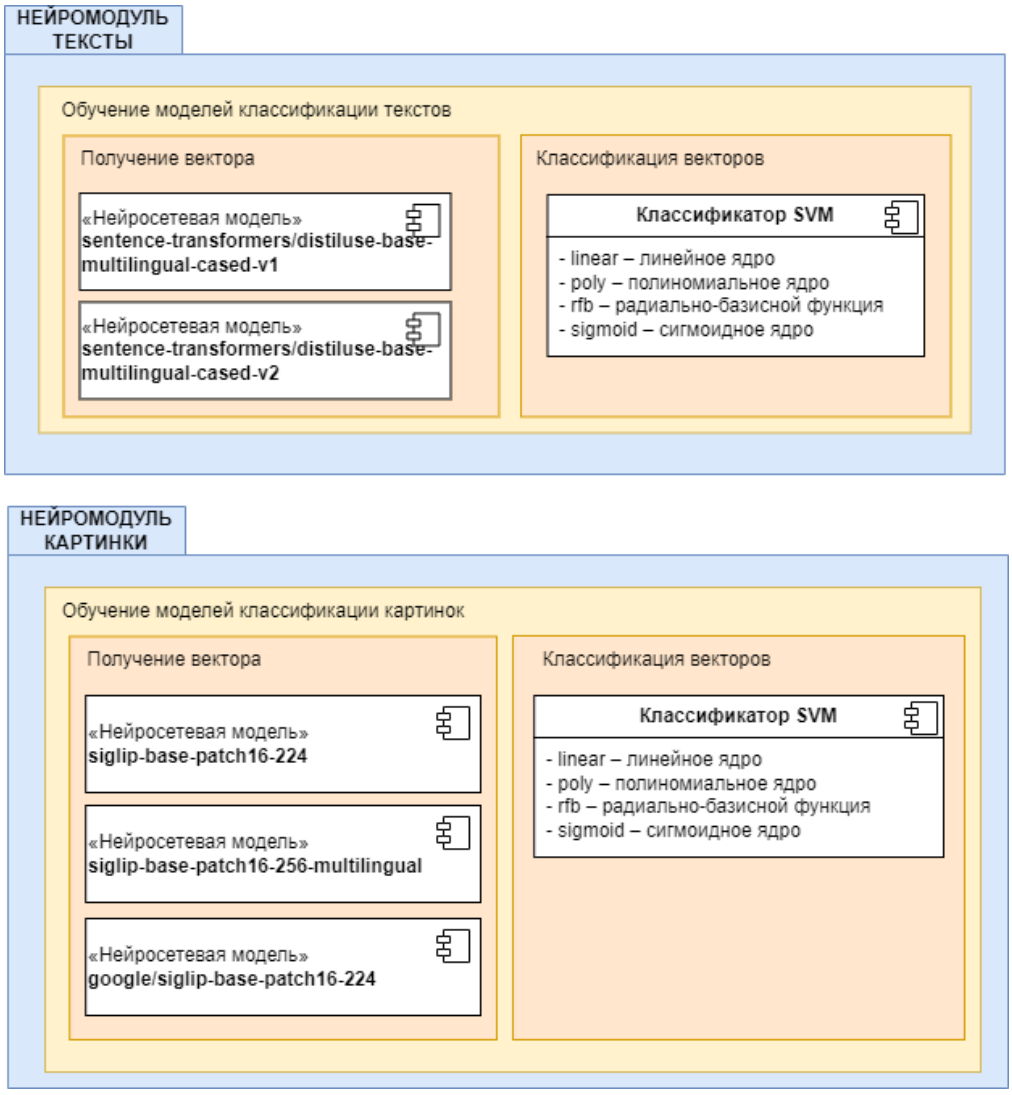

Рис. 2 – Состав нейросетевого модуля IQTrainer

В состав нейросетевого модуля IQTrainer входит следующий минимальный набор типов нейросетевых моделей:

– для текстов:

– получение вектора:

– модель *sentence-transformers/distiluse-base-multilingual-cased-v1*, модель поддерживает 50 языков, на выходе выдаёт вектор размерностью 512;

– модель *sentence-transformers/distiluse-base-multilingual-cased-v2*, модель поддерживает 30 языков, на выходе выдаёт вектор размерностью 512;

– классификация векторов – классификатор *SVM* (метод опорных векторов) со следующими ядрами:

– linear – линейное ядро;

– poly – полиномиальное ядро;

– rfb – радиально-базисной функция;

– sigmoid – сигмоидное ядро;

– для картинок:

– получение вектора

– модель *siglip-base-patch16-224*, модель строит вектор в едином пространстве как для картинок, так и для текстов, на выходе вектор размерностью 768;

– *siglip-base-patch16-256-multilingual*;

– *siglip-large-patch16-256*;

– классификация векторов – классификатор SVM (метод опорных векторов) со следующими ядрами:

– linear – линейное ядро;

– poly – полиномиальное ядро;

– rfb – радиально-базисной функция;

– sigmoid – сигмоидное ядро.

Также в нейросетевом модуле предусмотрена возможность расширения набора типов модулей.

В нейросетевом модуле может храниться несколько обученных моделей (версий) одного классификатора, но в каждый момент времени только одна из них является активной для использования. При активации одной из моделей все остальные модели этого классификатора будут деактивированы.

Расширение IQTrainer *image marker* для браузера Google Chrome предназначено для загрузки найденных на веб-сайтах картинок в IQTrainer. Для

обеспечения устойчивой работы расширения для браузера установлено максимальное количество загружаемых картинок – 1000 шт.

Схема загрузки и хранения данных в IQTrainer приведена на рисунке 2 ниже.

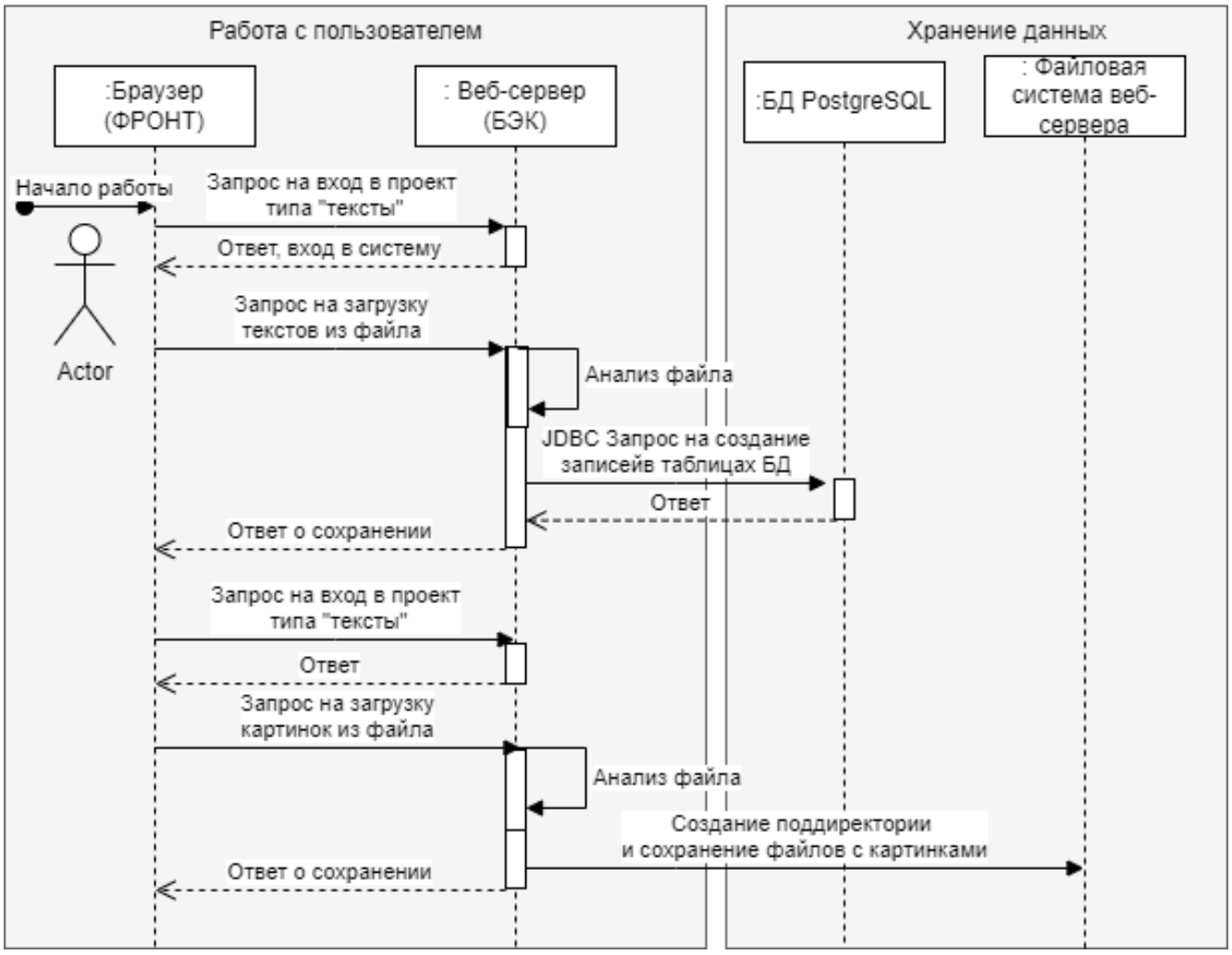

Рис. 2 – Загрузка и хранение данных в IQTrainer

Данные, необходимые для работы приложения хранятся в БД PostgreSQL.

В проектах разметки текстов данные хранятся только в базе данных, при загрузке данных из файла – сами файлы не сохраняются.

В проектах разметки картинок - сами картинки хранятся в файловой системе, данные хранятся в базе данных. При загрузке данных из ZIP-файла – сам файл не сохраняется, а извлечённые из него файлы с картинками сохраняются в файловой системе.

Вектора документов и картинок хранятся в Milvus для обеспечения поиска похожих документов, если включена интеграция с Milvus.

Milvus – это векторная база данных с открытым исходным кодом, разработанная специально для поиска сходства в массивных наборах данных многомерных векторов.

Внутри IQTrainer существует фоновый процесс, синхронизирующий содержимое базы данных и индексов в Milvus – всем документам одного проекта в IQTrainer соответствует один индекс в Milvus. Схема взаимодействия модулей программы при сохранении векторов в Milvus приведена далее на рисунке 3.

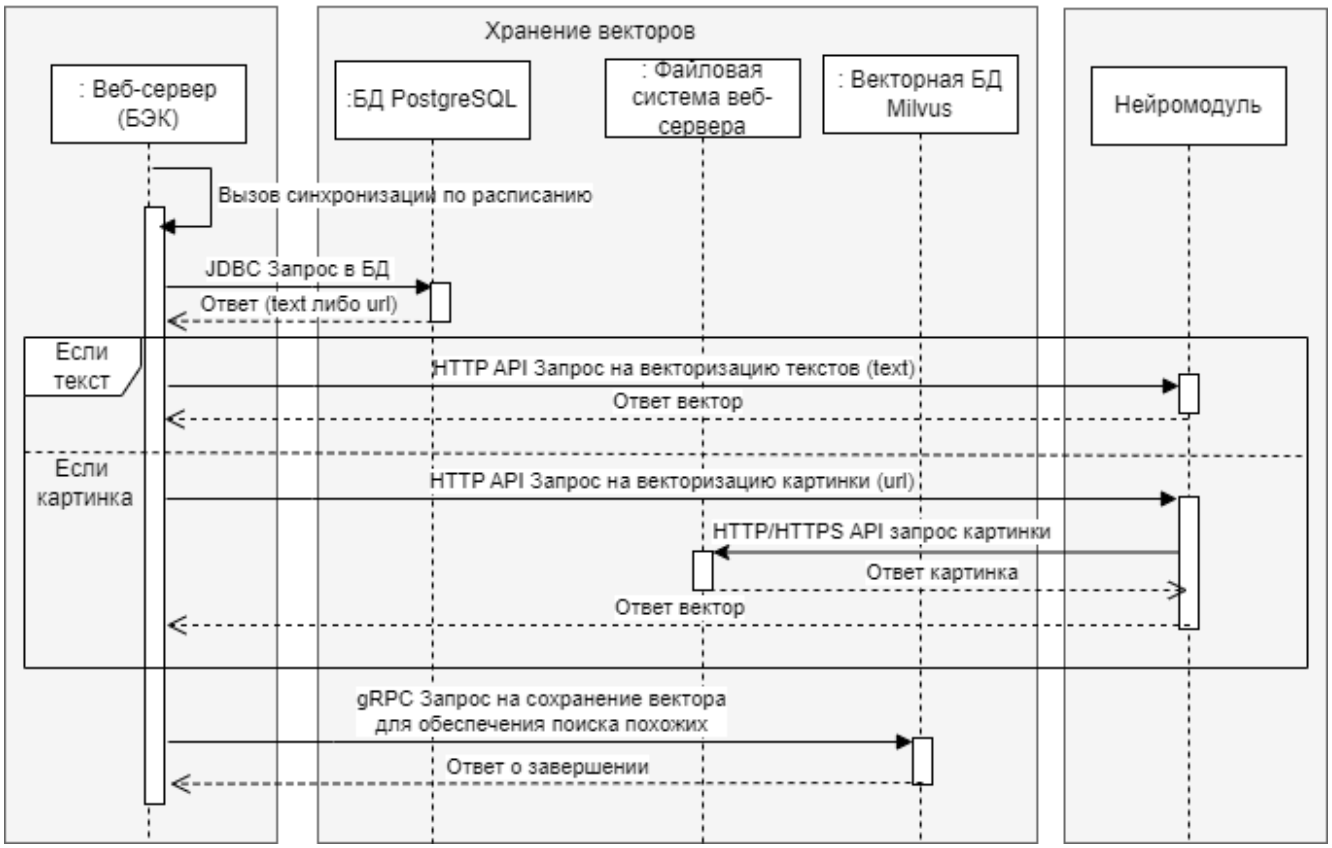

Рис. 3 – Схема взаимодействия при сохранении векторов в Milvus

Схема взаимодействия модулей Программы по API приведена далее на рисунке 4.

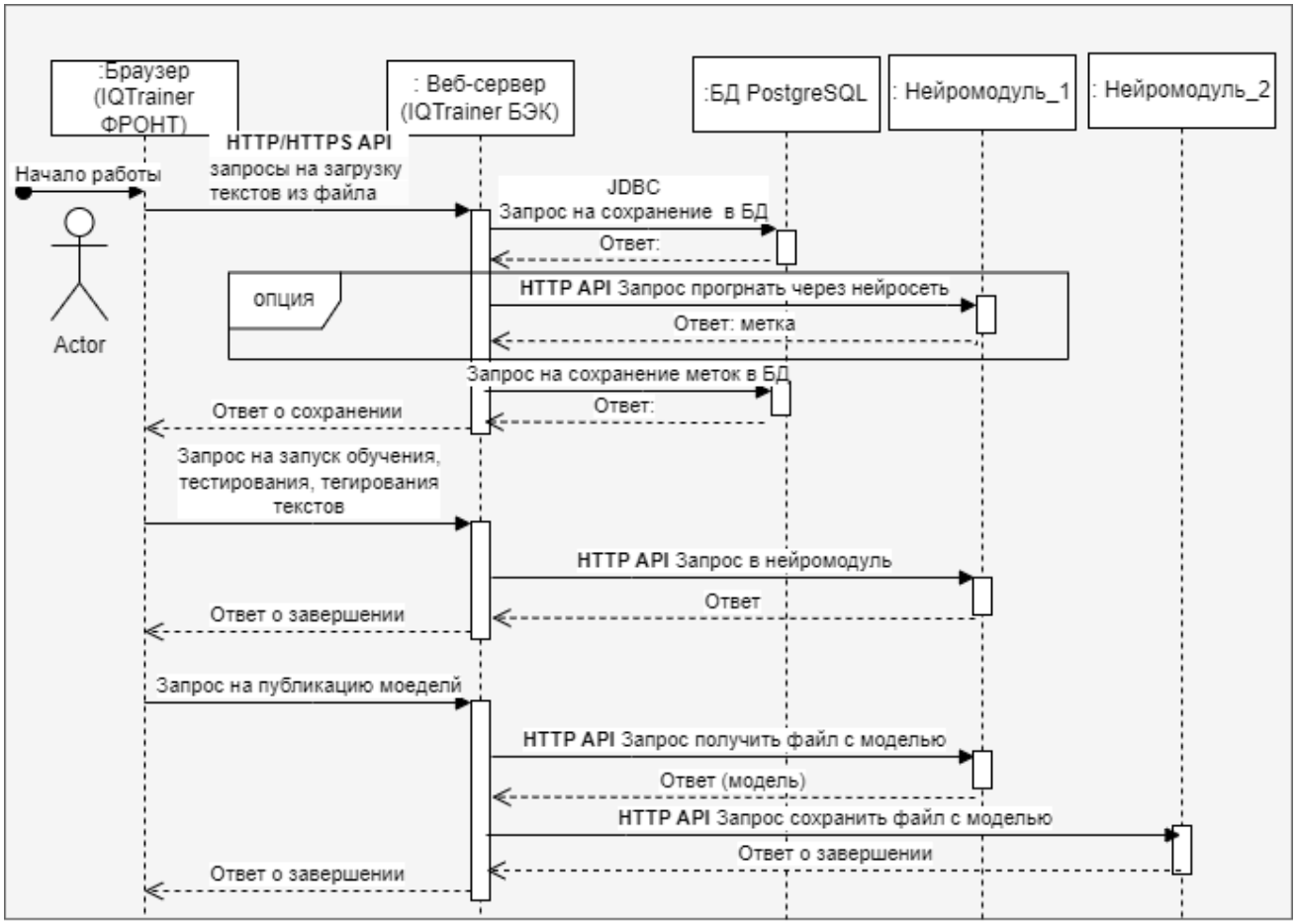

Рис. 4 – Схема взаимодействия модулей IQTrainer по API

1.1.3 Перечень функций, реализуемых программой

Программа поддерживает следующие наборы функций:

– в части управления правами доступа:

– управление учётными записями пользователей, включая CRUD-операции с учётными записями;

– управление группами пользователей, включая CRUD-операции с группами и привязку пользователей к группам;

– управление правами доступа к функциям Программы с помощью функциональных привилегий, назначаемых на группу пользователей;

– управление правами доступа пользователей к проектам;

– в части управления проектами:

– CRUD-операции с проектами;

– управление параметрами проектов, такими как:

– название и тип проекта («Текст» или «Картинки»);

– доступ пользователей к работе с проектами;

– определение вида отображения классификаторов (стандартный или иерархический);

– определение URL нейромодуля для обучения моделей;

– использование общесистемного либо кастомного кода публикации моделей;

– использование общесистемного либо кастомного алгоритма векторизации для обучения моделей в рамках проекта;

– определение параметров батча векторизации и классификации;

– управление классификаторами проекта;

– управление источниками для загрузки документов в датасеты;

– в части управления классификаторами:

– CRUD-операции с классификаторами;

– определение состава классов классификатора;

– определение набора датасетов, используемых для построения классификатора, включая операции добавления и удаления датасетов из классификатора;

– множественное использование датасетов для построения разных классификаторов;

– разметка пользователями датасетов в соответствии с классами выбранного классификатора;

– для проектов типа «Текст» и с видом отображения классификаторов «Стандартный» работа в режиме «Мультиразметка»:

– создание групп классификаторов для совместной разметки датасетов текстовых документов;

– создание датасетов, общих для группы классификаторов;

– разметка пользователями документов датасетов по классам группы классификаторов;

– поиск и удаление дублей документов из датасетов;

– сверка меток документа в разных датасетах;

– поиск и сверка меток похожих документов;

– для проектов с видом отображения классификаторов «Иерархический»:

– создание иерархических классификаторов;

– возможность создания иерархий с вложенностью до 10 уровней и количеством классов более 100;

– отображение иерархической структуры классов классификатора в интерфейсах просмотра и выбора;

– корректировка состава документов датасетов посредством массовых операций добавления-переноса-удаления всех либо выбранных документов датасета;

– в части управления датасетами:

– CRUD-операции с датасетами;

– корректировка состава документов датасетов посредством массовых операций добавления-переноса-удаления выбранных документов датасета;

– в части загрузки документов в датасет:

– создание и загрузка одиночного документа в датасет;

– загрузка найденных документов по поисковому запросу в подключённых к Программе источников (программно-аппаратный комплекс на базе Платформы IQPLATFORM);

– загрузка документов из файлов;

– просмотр истории загрузок, поиск и удаление ошибочно загруженных документов;

– автоматическое вычисление векторов похожести документов (эмбеддингов) при загрузке;

– фильтрация документов датасета по параметрам и тексту;

– поиск документов, похожих на выбранный в выбранных датасетах текущего классификатора;

– возможность удаления документов как из датасета, так и из приложения;

– импорт документов в датасет из файлов формата;

– текстов – CSV, JSON, XLS, XLSX;

– текстов – CSV, JSON;

– картинок – ZIP;

– экспорт документов датасета в файл форматов:

– текстов – CSV, JSON, XLSX;

– картинок – ZIP;

– в части работы с нейросетевыми моделями классификатора:

– выбор типа модели и обучение модели классификатора на заданном датасете;

– тестирование работы модели на тестовых датасетах, оценка качества обучения на группах документов с ошибками;

– анализ точности модели по матрице несоответствий в статистике тестирования датасета для иерархических классификаторов;

– обучение модели на новых дополнительных датасетах;

– автоматическое сохранение новых версий обучения одной модели;

– сравнение оценок качества работы разных моделей и/или разных версий одной модели для выбора оптимального варианта;

– импорт из файла формата ZIP ранее обученной модели для ее тестирования либо последующего обучения (при наличии начального обучающего датасета);

– экспорт обученной модели в файл формата ZIP для ее последующей установки на работающий программно-аппаратный комплекс на базе Платформы IQPLATFORM;

– публикация (копирование) обученной модели в рабочий проект согласно параметрам настройки обучающего проекта.

Расширение для браузера реализует следующие функции:

– настройка: ввод параметров приложения **IQTrainer** для подключения к установленной программе: URL, логин и пароль пользователя;

– отображение всех картинок с текущей страницы с возможностью увеличенного просмотра;

– выбор пользователем проекта, классификатора и датасета, в который будет произведена загрузка;

– автоматическое формирование подсказки для разметки всех или выбранных картинок по классам классификатора в случае наличия для него обученной модели;

– разметка пользователем всех или выбранных картинок по классам классификатора;

– автоматическая загрузка всех картинок с сохранением результатов пользовательской разметки в соответствии с ограничениями по количеству и размерам (ширина и/или высота), заданными пользователем.

1.1.4 Модель данных программы

В программе реализована следующая модель данных (рис. 5).

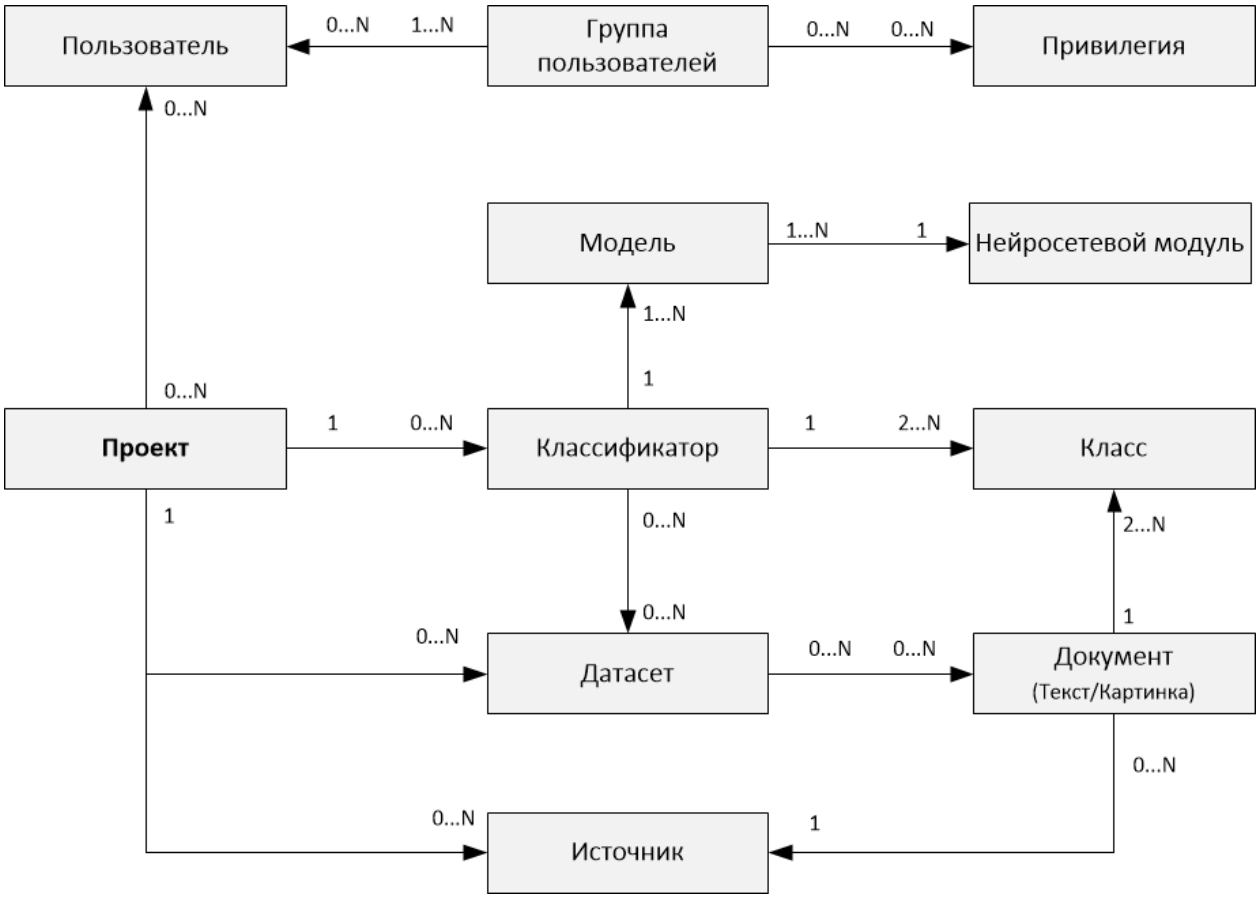

Рис. 5 – Модель данных программы

Пользователь – учётная запись пользователя Программы. Количество записей не ограничено.

Группа пользователей – группа пользователей, предназначена для управления правами доступа пользователей к функциям Программы. Групп может быть несколько. Пользователь может входить в несколько групп.

Привилегия – право доступа к некоторой функции Программы. Привилегии назначаются на группы пользователей.

Проект – сущность, предназначенная для группировки и хранения совокупности данных, используемых на одном проекте для обучения классификаторов и оценки качества обучения. Проект предназначен для работы только с одним типом документов – текстовыми документами либо картинками. Проект может содержать несколько классификаторов одного вида отображения:

стандартные либо иерархические. Доступ к проекту может быть у нескольких пользователей.

Классификатор – набор классов, по которым производится классификация документов – текстовых документов либо картинок. Минимальное количество классов – два: относится документ к классу или нет. Каждому классификатору можно сопоставить несколько датасетов под решение определённых задач (для обучения, тестирования, дообучения на специфических примерах и т.п.) с документами либо картинками, размеченными в ручном режиме пользователями. Для каждого классификатора можно обучить несколько моделей, а потом произвести сравнение качества обучения на тестовых датасетах. Для решения задач классификации текстов по большому количеству классов применяются классификаторы с иерархическим видом отображения классов, название которых содержит перечисление уровней иерархии через разделитель, задаваемый в настройках проекта (например, знак точки или наклонная черта).

Модель – нейросетевая модель, которая обучена классификации документов по классам определённого классификатора. Архитектура модели описывается ее типом. Модель может иметь версионность, после каждого цикла обучения производится сохранение новой версии модели. Обученная модель может быть экспортирована из приложения в файл. Также модель может быть импортирована в программу для тестирования либо последующего дополнительного обучения (при наличии обучающего датасета).

Тип модели – математическая архитектура нейросетевой модели, список типов используемых моделей кодируется в нейросетевом модуле при его создании.

Нейросетевой модуль – программный модуль в составе IQTrainer, предназначенный для обучения и хранения обученных моделей, а также для реализации сервиса потоковой кластеризации документов для программно-аппаратных комплексов, построенных на базе Платформы IQPLATFORM.

Датасет – набор документов, который может быть использован для обучения модели либо оценки качества её обучения (тестирования). Один и тот же датасет может быть использован в нескольких классификаторах разных проектов.

Документ – это текстовый документ либо картинка (файл графического формата), которые могут быть загружены в проект и привязаны к определённому датасету. Если датасет привязан к определённому классификатору, то пользователь может в ручном режиме задать его принадлежность к одному определённому классу этого классификатора. Данная разметка используется при обучении и оценке моделей этого классификатора. Для этого обучаемая модель также производит автоматическую разметку документов по классам классификатора.

Источник – конкретный программно-аппаратный комплекс, построенный на базе Платформы IQPLATFORM, который может быть использован для загрузки документов при создании датасетов.

Мультиразметка – режим разметки текстовых документов общего датасета для нескольких классификаторов, который применяется для обучения нескольких моделей классификаторов на одном наборе текстовых документов.

1.2 Состав программных средств

Программное обеспечение приложения IQTrainer и нейросетевого модуля работает под управлением операционных систем семейства **Linux**.

В работе программы используется следующее стороннее программное обеспечение:

– СУБД **PostgreSQL Server** 12.x (https://www.postgresql.org/), используется для хранения структурированных данных – пользователей, групп пользователей, сохранённых запросов, списков индустрий и технологических направлений, семантических ядер и т.п.;

– векторная база данных **Milvus** 2.3.x или 2.4.x (https://milvus.io/), используется для хранения и быстрого поиска похожих embedding-векторов;

– контейнеризатор приложений **Docker-**ce-24.0.x-x (https://www.docker.com/), используется для развёртывания и запуска сервисов;

– комплект разработчика JAVA-приложений **Java SE 21** (https://www.oracle.com/java/technologies/javase-downloads.html), используется для построения корпоративных приложений и сервисов на базе Java; поставляется внутри контейнеризированного приложения;

– библиотека **Hibernate** (https://hibernate.org/), используется для решения задач объектно-реляционного отображения в приложениях на языке Java; поставляется внутри контейнеризированного приложения;

– библиотека **Spring Framework** (https://spring.io/), используется для построения архитектурного каркаса корпоративных приложений на базе Java; поставляется внутри контейнеризированного приложения;

– Java фреймворк **Spring Boot** (https://spring.io/projects/spring-boot), используется при реализации веб-приложений; поставляется внутри контейнеризированного приложения;

– контейнер сервлетов **Apache Tomcat** (https://tomcat.apache.org/), используется для разработки и запуска веб-приложений на языке Java; поставляется внутри контейнеризированного приложения;

– язык программирования **JavaScript**, используется для разработки веб-приложений;

– JavaScript фреймворк **Vue.js** (https://ru.vuejs.org/), используется для создания пользовательских веб-интерфейсов;

– объектно-ориентированный язык программирования **Groovy** (https://groovy-lang.org/), разработанный как дополнение к языку Java с возможностями скриптовых языков;

– высокоуровневый объектно-ориентированный язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью **Python** (https://www.python.org/), используется для реализации нейросетевых модулей; поставляется внутри контейнеризированного приложения.

1.3 Комплект аппаратных и технических средств

Типовые требования к аппаратным и техническим средствам для установки программного обеспечения программы приведён далее в таблице 1.

Таблица 1 – Типовой комплект аппаратных и технических средств

№	Наименование аппаратных и технических средств	Кол-во, компл.	Название	Требования к оборудованию
1	Сервер приложения IQTrainer	1	`iqtrainer-appl`	8CPU, 32GB RAM, 1TB SSD

№	Наименование аппаратных и технических средств	Кол-во, компл.	Название	Требования к оборудованию
2	Сервер СУБД сервера приложений IQTrainer	1	`iqtrainer-db`	8CPU, 32GB RAM, 1TB SSD
3	Сервер нейросетевого модуля	1	`iqtrainer-neuronet`	8CPU, 32GB RAM, 1TB HDD, GPU NVidia A100 40GB
4	Сервер хранения векторов	1	`iqtrainer-milvus`	8CPU, 32GB RAM*, 2TB SSD * требования к оперативной памяти рассчитываются исходя из количества векторов: на каждые 10 млн векторов требуется 20 GB оперативной памяти для векторизатора *distiluse-base-multilingual-cased-v1*. При использовании других векторизаторов размер памяти считается по формуле: `4*<длина вектора>*<количество векторов>`

1.4 Установка ПО на технических средствах

Типовой план развёртывания программного обеспечения на технические средствах приведён далее в таблице 2.

Таблица 2 – Типовой план развёртывания ПО

№	Наименование аппаратных и технических средств и установленного программного обеспечения	Название сервера
1	Системное ПО	
1.1	Операционная система (CentOS 7.9.x)	На всех серверах
	Прикладное ПО	
2.1	СУБД PostgreSQL 12.xx	`iqtrainer-db`
2.2	База данных векторов Milvus 2.3.x или 2.4.x	`iqtrainer-milvus`
2.3	Система управления контейнерами Docker ce-24.0.x-x	На всех серверах
4	Специальное ПО	
4.1	Приложение IQTrainer	`iqtrainer-appl`
4.2	Нейросетевой модуль	`iqtrainer-neuronet`
4.3	Расширение для браузера	На сервера не устанавливается

2 Установка и настройка программы

2.1 Общий порядок установки

Установка программы производится в следующей последовательности:

– установка и настройка на серверном оборудовании Операционной Системы;

– установка и настойка прикладного программного обеспечения, требуемого для работы программы;

– установка и настройка приложения IQTrainer и нейросетевого модуля;

– запуск модулей;

– установка специального расширения для браузера на рабочих станциях пользователей.

2.2 Установка системного и прикладного ПО

2.2.1 Базовая настройка CentOS 7 minimal

2.2.1.1 Настройка сетевых интерфейсов

На каждом сервере настраиваются следующие интерфейсы:

– `eno1` (LAN1) – интерфейс управления;

– `enp94s0f0` и `enp94s0f1` – физические интерфейсы для передачи данных между серверами;

– `bond0` – объединение интерфейсов `enp94s0f0` и `enp94s0f1`;

– `bond0.10` – VLAN интерфейс для передачи данных.

Настройка интерфейса `eno1` выполняется следующей командой:

```
nmcli connection add con-name eno1 ifname eno1 type ethernet ipv6.method ignore
ipv4.method manual ipv4.addresses 172.30.77.XX/26
```

Настройка интерфейса `bond0` выполняется следующей командой:

```
nmcli connection add con-name bond0 ifname bond0 type bond bond.options
"mode=802.3ad,xmit_hash_policy=layer2+3,lacp_rate=1,miimon=100" ipv4.method
disabled ipv6.method ignore
```

Настройка интерфейсов `enp94s0f0` и `enp94s0f1` выполняется следующими

командами:

```
nmcli connection add con-name enp94s0f0 ifname enp94s0f0 type ethernet ipv6.method
ignore ipv4.method disabled slave-type bond master bond0
nmcli connection add con-name enp94s0f1 ifname enp94s0f1 type ethernet ipv6.method
ignore ipv4.method disabled slave-type bond master bond0
```

Настройка интерфейса `bond0` выполняется следующей командой:

```
nmcli connection add con-name bond0.10 ifname bond0.10 type vlan vlan.id 10 dev
bond0 ipv6.method ignore ipv4.method manual ipv4.addresses 10.10.77.XX/26
```

2.2.1.2 Подключение дополнительных репозиториев для CentOS 7

На всех машинах комплекса необходимо подключить репозиторий `EPEL` (https://fedoraproject.org/wiki/EPEL).

```
yum install –y epel-release.noarch
```

На всех машинах необходимо подключить репозиторий `PostgreSQL` (http://www.postgresql.org/).

```
yum install –y https://download.postgresql.org/pub/repos/yum/reporpms/EL-7-
x86_64/pgdg-redhat-repo-latest.noarch.rpm
```

2.2.1.3 Отключение IPv6

На всех машинах кластера необходимо отключить поддержку `IPv6`, для этого выполните следующие действия:

```
vi /etc/sysctl.conf
```

Добавьте следующие строки:

```
net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.default.disable_ipv6 = 1
```

Примените изменения и перезапустите сетевой интерфейс:

```
sysctl -p
systemctl restart network
```

2.2.1.4 Отключение selinux

Необходимо отключить `selinux` на всех машинах кроме тех, которые

смотрят напрямую в сеть Интернет, т.е. имеют внешние (белые) IP адреса на сетевых интерфейсах, если такие имеются.

```
vi /etc/selinux/config

# This file controls the state of SELinux on the system.
# SELINUX= can take one of the sethree values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=disabled
# SELINUXTYPE= can take one of three two values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected processes are
protected.
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

2.2.1.5 Отключение службы firewalld

На всех машинах комплекса необходимо отключить службу `firewalld`, кроме тех, которые смотрят напрямую в сеть Интернет, т.е. имеют внешние (белые) IP адреса на сетевых интерфейсах, если такие имеются.

Чтобы это сделать, выполните следующие команды:

```
systemctl stop firewalld
systemctl disable firewalld
systemctl mask firewalld
```

После завершения всех манипуляций сервер необходимо перезапустить командой:

```
shutdown -r now
```

2.2.2 Дополнительная настройка серверов комплекса

Дополнительная настройка производится только на серверах `neuronet1` и `neuronet2`.

1. Загружаем драйвера с официального сайта NVIDIA в соответствии с установленной видеокартой:

```
wget https://us.download.nvidia.com/XFree86/Linux-x86_64/535.104.05/NVIDIA-Linux-
x86_64-535.104.05.run
```

2. Устанавливаем необходимые зависимости:

```
yum groupinstall "Development Tools" -y
yum install kernel-devel kernel-headers dkms -y
```

3. Устанавливаем драйвер и следуем указаниям программы:

```
sh NVIDIA-Linux-x86_64-535.104.05.run
```

4. Устанавливаем `docker-ce`:

```
yum remove docker \
                  docker-client \
                  docker-client-latest \
                  docker-common \
                  docker-latest \
                  docker-latest-logrotate \
                  docker-logrotate \
                  docker-engine
yum install -y yum-utils
yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo
yum install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

5. Устанавливаем `nvidia toolkit` для `docker-ce`:

```
distribution=$(. /etc/os-release;echo $ID$VERSION_ID)
curl -s -L https://nvidia.github.io/nvidia-docker/$distribution/nvidia-docker.repo | sudo tee /etc/yum.repos.d/nvidia-docker.repo

yum install -y nvidia-container-toolkit
systemctl restart docker
```

6. Устанавливаем дополнительные пакеты:

```
yum install –y bash-completion ncdu rsync vim nano wget mc net-tools tmux screen mtr mailx zip unzip lynx bind-utils telnet tcpdump iftop htop sysstat lsof strace iotop dos2unix traceroute psmisc vim-enhanced
```

7. Добавляем пользователя:

```
useradd -u 500 iqusr
groupmod -g 500 iqusr
chgrp iqusr ~iqusr
```

8. Производим корректировку ограничений используемых ОС ресурсов:

```
echo"*       soft    nproc   65000"> /etc/security/limits.d/20-nproc.conf
echo"*       hard    nproc   1000000">> /etc/security/limits.d/20-nproc.conf
echo"*       -       nofile  1048576">> /etc/security/limits.d/20-nproc.conf
echo"root    -       memlock unlimited">> /etc/security/limits.d/20-nproc.conf
```

9. Расширяем объём журнала действий пользователей:

```
echo"export HISTSIZE=10000000">> ~/.bashrc
echo"export HISTTIMEFORMAT=\"%F %T: \"">> ~/.bashrc
```

2.2.3 Установка и настройка СУБД PostgreSQL

2.2.3.1 Установка СУБД PostgreSQL

Для установки СУБД PostgreSQL выполните следующие команды:

```
yum install -y postgresql12-server postgresql12-contrib
/usr/pgsql-12/bin/postgresql-12-setup initdb
systemctl enable postgresql-12
systemctl start postgresql-12

systemctl stop postgresql-12

mkdir /work/pg_data

mv /var/lib/pgsql/12/data /work/pg_data

ln -s /work/pg_data/data /var/lib/pgsql/12/data

chown -R postgres:postgres /work/pg_data/data

systemctl start postgresql-12
```

Затем нужно произвести настройку конфигурационного файла аутентификация клиентов:

```
vi /var/lib/pgsql/12/data/pg_hba.conf
local all   all                                   trust
host        all   all   127.0.0.1/32      trust
host        all   all   0.0.0.0/0   md5
```

Произвести настройку конфигурационного файла сервера СУБД

```
vi /var/lib/pgsql/12/data/postgresql.conf
      listen_addresses = '*'
```

Для этого следует воспользоваться генератором конфигураций https://pgconfigurator.cybertec-postgresql.com на основании спецификации сервера и назначения, а затем внести изменения в конфигурационный файл СУБД.

После чего перезапустить СУБД:

```
systemctl restart postgresql-12
```

2.2.3.2 Создание Баз данных

Подготовить для каждого модуля с БД на сервере СУБД:

```
su - postgres
psql
CREATE ROLE module_name WITH LOGIN PASSWORD 'change_me';
CREATE DATABASE module_namedb OWNER module_name;
GRANT ALL privileges ON DATABASE module_namedb TO module_name;
```

2.3 Установка приложения **IQTrainer**

2.3.1 Общие положения

Шаг 1. Необходимо создать директорию, где будут находиться рабочие и конфигурационные файлы приложения.

Далее считаем, что создана директория `/work/analytics/iqtrainer`.

Шаг 2. Распаковать `iqt-server-docker.zip` в эту директорию.

Шаг 3. Удалить старые образы приложения, выполнив скрипт `/work/analytics/iqtrainer/purge_all.sh`:

```
#!/bin/bash
# удаляет из репозитория все образы проекта
docker rmi --force $(docker images 'iqmen/iqtrainer' -a -q)
```

Шаг 4. Загрузить новый образ, выполнив скрипт `/work/analytics/iqtrainer/load_tar_image.sh`:

```
#!/bin/bash

# Пример скрипта загрузки tar образа в локальный докер репозиторий

HOME="$( cd "$( dirname "${BASH_SOURCE[0]}" )" >/dev/null 2>&1 && pwd )"

VERSION=$1
BUILD=$2

docker rmi iqmen/iqtrainer:latest
docker rmi iqmen/iqtrainer:$VERSION.latest
docker rmi iqmen/iqtrainer:$VERSION.$BUILD
docker load --input $HOME/iqt-server-v$VERSION.$BUILD.tar
```

Шаг 5. Отредактировать файл настроек `/work/analytics/iqtrainer/config/application.properties`.

Если такого файла нет, то его можно создать, скопировав файл `/work/analytics/iqtrainer/config/application.template.properties`:

```
cp /work/analytics/iqtrainer/config/application.template.properties
/work/analytics/iqtrainer/config/application.properties
```

В этом файле необходимо произвести настройку следующих параметров.

1. Параметры подключения к базе данных:

– `spring.datasource.url=jdbc:postgresql://<iqtrainer-db>:5432/iqtrainerdb?&ApplicationName=iqTrainer` – в этом параметре необходимо указать IP-адрес или доменное имя машины где будет запущен PostgreSQL;

– `spring.datasource.username=iqt_db` – в этом параметре необходимо имя пользователя для работы с базой;

– `spring.datasource.password=iqt_db` – в этом параметре необходимо пароль пользователя для работы с базой.

2. В случае, если IQTrainer будет использоваться для обучения классификаторов картинок, необходимо отредактировать параметр:

– `iqt.external.baseUrl=http://XXX.XXX.XXX.XXX:80`

Данный параметр должен задавать URL IQTrainer, видимый с машины, на которой будет развернут нейромодуль обработки картинок, так как нейромодуль получает картинки по URL.

3. Для поддержки поиска похожих документов необходимо выбрать сервис поиска похожих (`milvus`, `pgvector`, `none`) и настроить интеграцию с выбранным сервисом

– `none` – поиск похожих отключён;

– `pgvector` – поиск похожих осуществляется через расширение `postgres/pangolin pgvector`;

– `milvus` – поиск похожих осуществляется с помощью векторной базы данных Milvus.

Нужно настроить интеграцию с выбранным сервисом в файле `applicatoion.properties`. в порядке, изложенном в разделе: «2.5 Поддержка

поиска похожих в Milvus и PGVECTOR».

С целью экономия ресурсов имеется возможность включения/выключения поиска похожих для каждого проекта отдельно. Для этого необходимо зайти в IQTrainer с ролью Администратор на страницу Настройки, перейти на вкладку Проекты. Включить/выключить поиск похожих можно в карточке проекта.

4. Для запуска приложения в режиме HTTPS:

– `server.ssl.enabled=true` – включает использование SSL.

Если поменять только эту настройку для HTTPS, то по умолчанию будет использован самоподписной сертификат Iqmen.

В этом случае необходимо указать следующие параметры:

– `server.ssl.key-store-type=jks` – тип хранилища ключей JKS или PKCS12;

– `server.ssl.key-store-password=iqmen.ibs` – пароль для доступа к хранилищу ключей;

– `server.ssl.key-alias=iqmen.ru` – имя ключа внутри хранилища;

– `server.ssl.key-password=iqmen.ibs` – пароль для доступа к ключу в хранилище ключей.

Для использования своего сертификата нужно при запуске в команде указать пусть до хранилища сертификатов, например, так:

```
/work/analytics/iqtrainer/run.sh --sslkeystor=/work/analytics/iqtrainer/config/cer
tificate.jks
```

Пример импорта существующего SSL-сертификата в новое хранилище:

```
keytool -import -alias iqmen.ru -file myCertificate.crt -keystore iqmen-ssl.jks -
storepass iqmen.ibs
```

В диалоге `Trust this certificate? [no]`: надо написать `yes`.

Шаг 6. Запустить докер-контейнер с приложением. Пример запуска можно найти в файле `/work/analytics/iqtrainer/run.sh`, поставляемого с приложением.

Справку по скрипту запуска можно посмотреть, выполнив команду:

```
/work/analytics/iqtrainer/run.sh –help
```

Если необходимо запускать приложение как сервис *systemctl*, то необходимо самостоятельно создать файл конфигурации на основе содержимого `run.sh`, либо самостоятельно настроить скрипты на выполнение команды `docker run`. Ниже приведены параметры запуска команды:

```
docker run --rm -it \
  -p <http_порт_сервера>:8080 \
  [-p <jmx_порт_сервера>:7320] \
  [-p <порт_отладки_сервера>:8990] \
  -v <путь_к_файлу_настроек>:/iqtrainer/config/application.properties \
  [-v <путь_к_файлу_хранилища_ключей>:/iqtrainer/config/ssl.keystore] \
  -v <путь_к_папке_файлового_хранилища>:/iqtrainer/fileStorage \
  -v <путь_к_папке_с_логами>:/iqtrainer/logs \
  [-v <путь_к_временной_папке>:/iqtrainer/tmp] \
  -v <путь_к_рабочей_директории>:/iqtrainer/work \
  -e "JAVA_TOOL_OPTIONS=-Xmx8g" \
  iqtrainer:latest
```

В квадратных скобках заданы необязательные параметры. Если их не задать, то создадутся папки внутри контейнера, а вспомогательные порты дебаггинга и диагностики по JMX будут не доступны.

2.3.2 Настройка аутентификации пользователей

Приложение IQTrainer поддерживает LDAP (Lightweight Directory Access Protocol) для аутентификации пользователей.

Имеется возможность комбинированной аутентификации - как через LDAP, так и через систему аутентификации самого IQTrainer.

Подробнее смотри в разделе «Поддержка LDAP».

2.3.3 Настройка работы с картинками

В проектах разметки картинок данные хранятся в файловой системе в директории, которая конфигурируется при старте IQTrainer.

Директория монтируется в докер-контейнер через параметр:

`-v <путь_к_папке_файлового_хранилища>:/iqtrainer/fileStorage`.

Образ докер создан на базе образа *nvidia/cuda:11.1.1-runtime-ubuntu20.04* – используемый Linux ubuntu-20.04.

2.4 Поддержка LDAP

Для поддержки LDAP необходимо в `application.properties` добавить следующие параметры:

– `ldap.enabled=true` – данный параметр включает поддержку аутентификации по LDAP. Если его не задать, или задать равным `false`,

то аутентификация ведётся через встроенную систему аутентификации IQTrainer по логину и паролю

– `ldap.combinedAuth=true` - данный параметр включает поддержку комбинированной аутентификации - т.е. возможности аутентифицироваться

как через систему аутентификации IQTrainer по логину и паролю, так и по email и доменному паролю в LDAP. Комбинированная аутентификация доступна только если включена поддержка LDAP `ldap.enabled=true`.

Чтобы пользователь мог аутентифицировать через систему аутентификации IQTrainer, необходимо в панели администрирования IQTrainer зайти в его карточку и включить ему режим локальной аутентификации.

При миграции все существующие пользователи получают статус локальной аутентификации.

– `ldap.provider.url=ldap://10.168.212.104` - URL сервера LDAP

– `ldap.admin.login=office\\danilov` - логин технической учётной записи, под которой к LDAP будет обращаться IQTrainer

– `ldap.admin.pass=XXX` -пароль технической учётной записи, под которой к LDAP будет обращаться IQTrainer

– `ldap.context.factory=com.sun.jndi.ldap.LdapCtxFactory` - необходимо добавить этот параметр как есть

– `ldap.searchBase=OU=IQMen IBS,DC=office,DC=iqmen,DC=ru` - строка конфигурации, которая задаёт базовый путь (search base) для поиска в LDAP. Задайте её согласно вашим настройкам Active Directory.

2.5 Поддержка поиска похожих в Milvus и PGVECTOR

Вектора документов (и текстов, и картинок) хранятся в Milvus либо PGVECTOR для обеспечения поиска похожих документов, если включена

интеграция с выбранным сервисом.

Внутри IQTrainer существует фоновый процесс, синхронизирующий содержимое базы данных и индексов в выбранном сервисе – всем документам одного проекта в IQTrainer соответствует один индекс в выбранном сервисе.

Интеграция с выбранным сервисом настраивается в файле `applicatoion.properties`.

Для начала требуется выбрать сервис поиска похожих:

– `pgvector` - поиск похожих осуществляется через расширение `postgres/pangolin pgvector`;

– `milvus` - поиск похожих осуществляется с помощью векторной базы данных Milvus;

– `none` - поиск похожих отключён.

2.5.1 Настройка интеграции с Milvus

Для настройки подключения к Milvus `iqt.similarity.milvus.client` необходимо в файл application.properties добавить следующие настройки: `iqt.similarity.service=milvus`.

Следующие сроки можно скопировать отсюда и проставить только хост и порт. Остальные оставить как есть:

```
iqt.similarity.milvus.client.host=localhost
iqt.similarity.milvus.client.port=19530
iqt.similarity.milvus.client.connectTimeout=10
iqt.similarity.milvus.client.keepAliveTime=-1
iqt.similarity.milvus.client.keepAliveTimeout=20
iqt.similarity.milvus.client.keepAliveWithoutCalls=false
iqt.similarity.milvus.client.idleTimeout=86400
```

Название индекса векторов в milvus – `iqt.similarity.milvus.embeddingIndexName=idx_emb`

2.5.2 Настройка интеграции с PGVECTOR

Для настройки подключения к PostgreSQL `iqt.similarity.pgvector.client.*` необходимо в файл `application.properties` добавить следующие настройки.

– `iqt.similarity.service=pgvector`, если они не заданы, то предполагается что данные по поиску похожих будут храниться в той же базе данных, что и все остальные данные IQTrainer.

– Т.к. pgvector это плагин к PostgreSQL то драйвер всегда от PostgreSQL:

– `iqt.similarity.pgvector.client.url=${spring.datasource.url:}`;

– `iqt.similarity.pgvector.client.username=${spring.datasource.username:}`;

– `iqt.similarity.pgvector.client.password=${spring.datasource.password:}`;

– `iqt.similarity.pgvector.client.driver-class-name=org.postgresql.Driver`.

– Размер батча векторов для вставки в векторную базу данных: `iqt.similarity.insert.batchSize=1000`.

– Интервал синхронизации документов базы данных и сервиса поиска похожих, в миллисекундах: `iqt.similarity.update.interval=60000`.

– Типы проектов, в которых надо синхронизировать документы (доступные значения: `text`, `image`):

– `iqt.similarity.update.doctypes=text`,

– `iqt.similarity.update.doctypes=image`.

– Настройки коллекции для проекта типа текст.

– Префикс названия коллекции проекта: `iqt.similarity.text.prefix=iqt_txt_`. Каждый проект хранит вектора своих документов в своей коллекции.

– Размерность характеристического вектора, данный параметр нужен только для векторизатора по умолчанию, который используется, если в настройках проекта не задан свой векторизатор. Если же в настройках проекта указан свой векторизатор, то размерность берётся по API из информации о векторизаторе: `iqt.similarity.text.dimension=512`

– Максимальный размер файла сегмента индекса, в мегабайтах нужен для создания коллекций в Milvus: `iqt.similarity.text.indexFileSize=1024`.

– Метрика расчёта похожести векторов:

– `L2 (Euclidean distance), IP (Inner product)`;

– `iqt.similarity.text.metricType=IP`.

– Тип векторизатора текстов. Данный параметр нужен только для векторизатора по умолчанию, который используется, если в настройках проекта не задан свой векторизатор: `iqt.similarity.text.vectorizerType=sentence-transformers/distiluse-base-multilingual-cased-v1`.

– Аналогичные параметры в проектах с картинками для поиска похожих картинок:

– Префикс названия коллекции проекта типа картинка: `iqt.similarity.image.prefix=iqt_img_`.

– Размерность характеристического вектора по умолчанию: `iqt.similarity.image.dimension=768`.

– Максимальный размер файла сегмента индекса, в мегабайтах: `iqt.similarity.image.indexFileSize=1024`.

– Метрика расчёта похожести векторов:

– `L2 (Euclidean distance), IP (Inner product)`;

– `iqt.similarity.image.metricType=IP`.

– Тип векторизатора картинок по умолчанию: `iqt.similarity.image.vectorizerType=google/siglip-base-patch16-224`.

2.6 Установка нейросетевого модуля для текстов

Импортировать переданный докеровский образ нейромодуля в докер.

Образ докер создан на базе образа *nvidia/cuda:12.1.0-devel-ubuntu20.04* – используемый Linux ubuntu-20.04.

Запуск производится командой:

```
docker run -v /work/models:/classifier/data/models/trained -p 8080:8080 --gpus all -it trainer5 --port 8080 –name trainer5
```

где:

– `-v /work/models` – папка на диске, там будут храниться обученные классификаторы;

– `--port 8080` – порт, на который будет запрос;

– `-p 8080:8080` – пробрасывается порт в докер;

– `--gpus all` – ГПУ, на котором будет работать сервис;

– `-name` – имя контейнера.

2.7 Установка нейросетевого модуля для картинок

Импортировать переданный докеровский образ нейромодуля в докер.

Образ докер создан на базе образа *nvidia/cuda:12.1.0-devel-ubuntu20.04* – используемый Linux ubuntu-20.04.

Запуск производится командой:

```
docker run -v /home/user/image_serv:/classifier/data -p 8580:8580 --gpus device=0
-it image_service11 --port 8580
```

где:

– `-v  /home/user/image_serv` – папка на диске, там будут храниться обученные классификаторы;

– `-port 8580` – порт, на который будет запрос;

– `-port 8580` – порт, на который будет запрос;

– `-p 8580:8580` – пробрасывается порт в докер;

– `-gpus device=0`– ГПУ, на котором будет работать сервис.

После запуска будут выведены названия всех классификаторов, если они есть в папке `/home/user/image_serv`.

Запуск считается, успешным если в выводе команды просмотра лог файла docker контейнера присутствует запись «INFO - http://YOUR_IP:9580/»

2.8 Установка расширения для браузера (для загрузки картинок)

Открыть приложение IQTrainer. перейти на страницу настроек во вкладку «Прочее». Скачать расширение, распаковать.

Как установить расширение Google Chrome описано здесь: https://support.google.com/chrome_webstore/answer/2664769?hl=ru

Перейти в параметры расширения, указать адрес IQTrainer, логин и пароль пользователя.

2.9 Сопряжение приложения IQTrainer с нейросетевыми модулями

Настройка сопряжения с нейросетевыми модулями осуществляется в настройках в веб-интерфейсе приложения для конкретного проекта.

Для это войдите в Настройки системы нажатием иконки в правом верхнем углу (рис. 6), затем откройте вкладку Проекты. Проект для публикации моделей и публикующий проект должны быть привязаны к разным нейросетям (нейросетевым модулям).

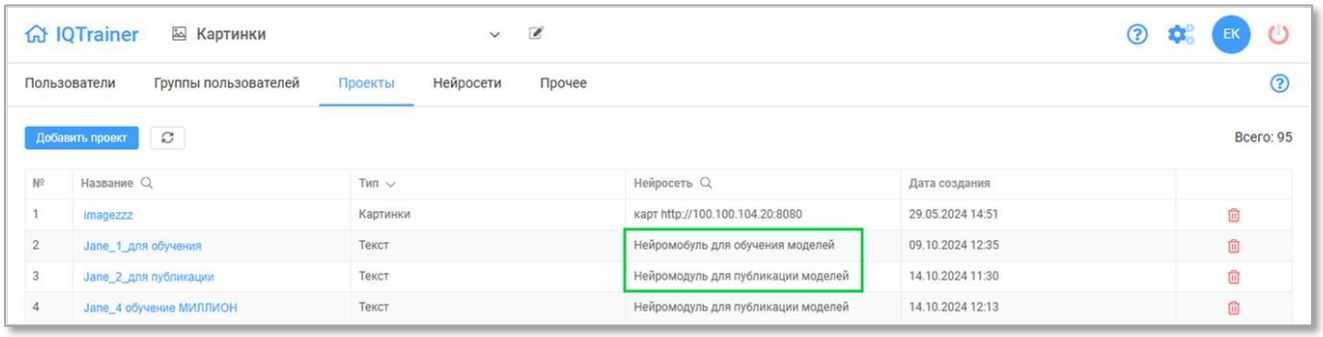

Рис. 6 – Вызов диалога настройки

Для настройки сопряжения со модулем для публикации модели необходимо выбрать один или несколько проектов из выпадающего списка проектов, привязанным к нейросетям, отличным от нейросети редактируемого проекта (рис. 7) и сохранить проект:

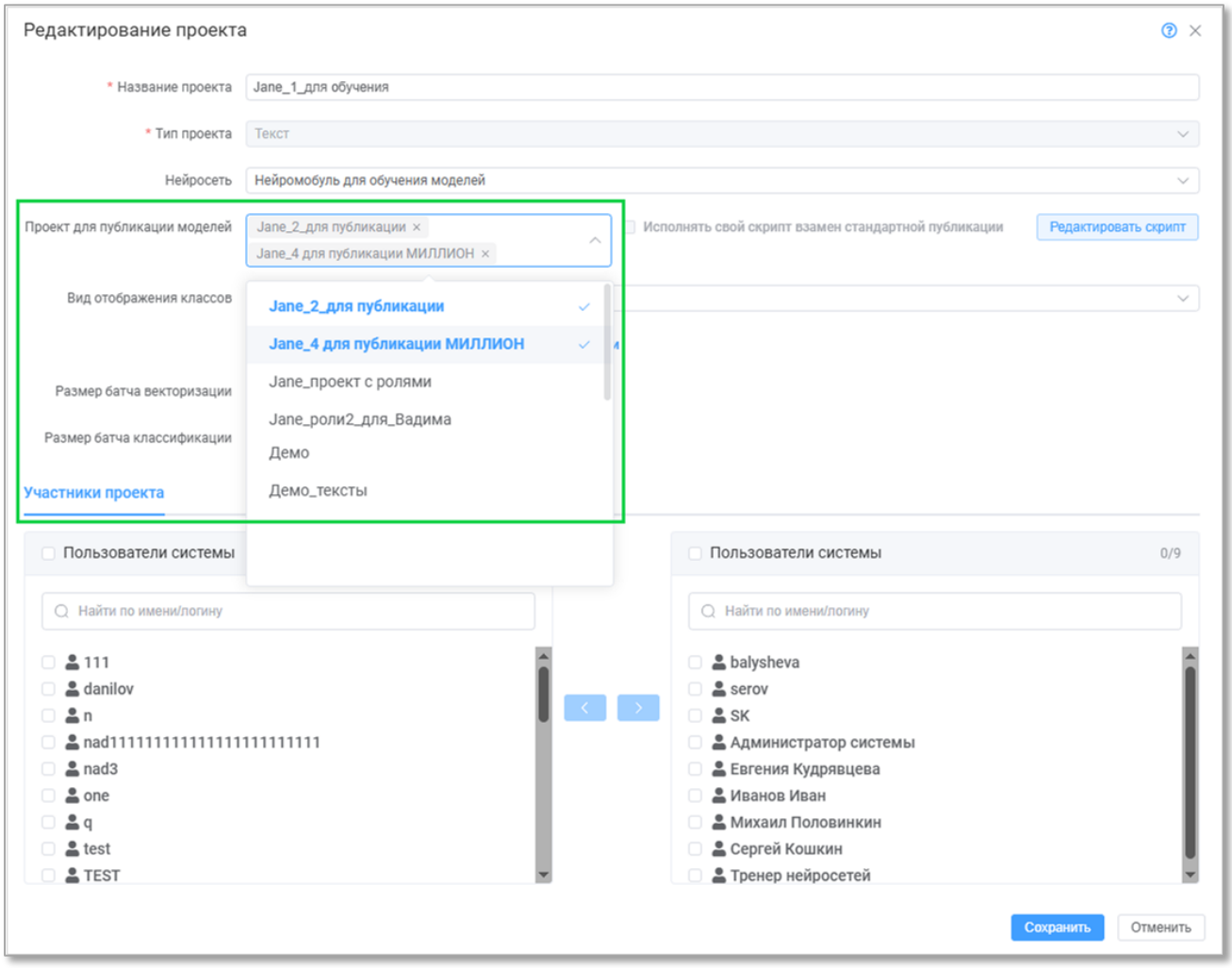

Рис. 7 – Настройка проекта для публикации

3 Администрирование программы

3.1 Остановка и запуск

3.1.1 Общий порядок запуска

Запуск программных модулей осуществляется в следующем порядке:

– запуск СУБД PostgreSQL;
– запуск приложения IQTrainer;
– запуск веб-сервера.

Время запуска модулей обычно не превышает от нескольких минут до часа и зависит от множества факторов, например, от кол-ва проектов, классификаторов и датасетов.

Критерием успешного запуска для программы является запись об успешном старте в логе конкретного модуля, а также вход в веб-интерфейс IQTrainer.

3.1.2 Общий порядок остановки

Остановка программных модулей осуществляется в обратном порядке:

– остановка веб-сервера;
– остановка приложения IQTrainer;
– остановка СУБД PostgreSQL.

3.1.3 Запуск и остановка программных модулей

3.1.3.1 Запуск и остановка нейросетевого модуля для текстов

Запуск:

```
docker run -v /work/models:/classifier/data/models/trained -p 8080:8080 --gpus all
-it trainer5 --port 8080 —name trainer5
```

Запуск считается, успешным если в выводе команды просмотра лог файла docker контейнера присутствует запись «INFO - http://YOUR_IP:8682/», например:

```
$docker logs trainer5
INFO:root:Summary model loading...
2025-01-30 10:05:07,942 - INFO - Summary model loading...
```

```
...
2025-01-30 10:05:28,673 - INFO - Loading completed
INFO:root:http://YOUR_IP:8682/
2025-01-30 10:05:28,679 - INFO - http://YOUR_IP:8682/
```

Остановка:

```
docker stop trainer5
```

3.1.3.2 Запуск и остановка нейросетевого модуля для картинок

Запуск:

```
docker run -v /home/user/image_serv:/classifier/data -p 8580:8580 --gpus device=0
-it image_service11 --port 8580
```

Где,

– `-v /home/user/image_serv` – папка на диске, там будут храниться обученные классификаторы;

– `-port 8580` – порт, на который будет запрос;

– `-port 8580` – порт, на который будет запрос;

– `-p 8580:8580` – пробрасывается порт в докер;

– `-gpus device=0`– ГПУ, на котором будет работать сервис.

После запуска будут выведены названия всех классификаторов, если они есть в папке `/home/user/image_serv`.

Запуск считается, успешным если в выводе команды просмотра лог файла docker контейнера присутствует запись «INFO - http://YOUR_IP:9580/», например:

```
$docker logs image_service11
INFO:root:Summary model loading...
…
RabbitMQ is not available
2025-06-04 07:42:31,664 - INFO - http://YOUR_IP:8580/
```

Остановка:

```
docker stop image_service11
```

3.1.3.3 Запуск и остановка СУБД PostgreSQL

Запуск модуля:

```
systemctl start postgresql-12
```

Запуск считается успешным, если в лог файле `/var/lib/pgsql/12/data/log/postgresql-XXX.log`, где `XXX` ¬ Fri, Mon, Sat, Sun, Thu, Tue, Wed, присутствует запись:

```
система БД готова принимать подключения
```

Остановка модуля:

```
systemctl stop postgresql-12
```

3.1.3.4 Запуск и остановка сервера приложений IQTrainer

Запуск:

```
/work/analytics/iqtrainer/run.sh
```

Запуск считается, успешным если в выводе команды просмотра лог файла docker контейнера присутствует запись «Started IQTrainerApplication in x seconds (process running for x)», например:

```
$docker logs iqtrainer
 INFO 2025-01-29 16:44:55.737 --- [main] c.i.iqTrainer.app.IQTrainerApplication
: Starting IQTrainerApplication v0.1.829 using Java 21.0.5 with PID 1
(/iqtrainer/classpath/iqt-server-0.1.829-plain.jar started by root in
/iqtrainer/work)
...
 INFO 2025-01-29 16:45:11.023 --- [main] c.i.iqTrainer.app.IQTrainerApplication
: Started IQTrainerApplication in 16.025 seconds (process running for 16.875)
```

Остановка:

```
docker stop iqtrainer
```

3.2 Диагностика работоспособности

Наиболее простой способ проверки работоспособности модулей – в окне браузера осуществить вход в веб-интерфейс проверяемого модуля:

– `http://[ip_appl_server]:port`;

где `ip_appl_server ip` – IP-адрес сервера приложений IQTrainer, `port` – номер порта сервера, например, `8080`.

В случае если вход в веб-интерфейс модуля был отклонён не по ошибке авторизации, следует провести полную диагностику соответствующего модуля.

Если по результатам диагностики видно, что модуль работает некорректно, то следует выполнить перезапуск модуля. С этой целью необходимо:

– выполнить остановку программных модулей в порядке, изложенном в разделе «3.1.2 Общий порядок остановки»;

– выполнить запуск программных модулей программы в порядке, изложенном в разделе «3.1.1 Общий порядок запуска».

Диагностика работоспособности СУБД PostgreSQL и Milvus выполняются средствами этих инструментов.

Диагностика работоспособности нейросетевых модулей осуществляется HTTP-запросом `http://:<IP_iqtrainer-neuronet>:<port>/health`, где `IP_iqtrainer-neuronet` и `port` – IP-адрес и порт сервера `iqtrainer-neuronet` установки нейромодулей.

Если один или несколько программных модулей программы не запускаются, или происходит остановка в процессе запуска, то необходимо:

– проверить записи логов модуля;

– при необходимости выполнить восстановление сбойного модуля из резервной копии, дистрибутива модуля или дистрибутива модуля крайнего обновления, предварительно сняв копию текущей сбойной версии модуля и его БД (при её наличии);

– обратиться в службу технической поддержки Исполнителя.