СОДЕРЖАНИЕ

# ВВЕДЕНИЕ

Актуальность разработки настоящей информационной системы для предметной области «Техникум» заключается в необходимости сокращения времени обработки информации и увеличении скорости обработки данных.

Объектом данной исследовательской работы является техникум.

Предметом настоящей работы является нагрузка учителей.

Разработка ИС включает в себя выполнение следующих задач:

* Написание технической документации для автоматизированного подсчёта нагрузки учителей;
* изучение новых библиотек для написания приложений на языке программировании Java;
* расширение знаний использования библиотеки Hibernate, для работы с базами данных;

Целью учебной практики является создание автоматизированного подсчёта нагрузки учителей.

# 1 ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ

## 1.1 Анализ предметной области

Техникум - учебное заведение, главным образом в системах образования Великобритании, стран Британского Содружества и США.

По уровню подготовки учащихся различают три типа колледжей:

- соответствующие уровню высшей школы: входят в состав университетов или автономные вузы;

- занимающие промежуточное положение между средними и высшими учебными заведениями;

- соответствующие старшим классам средней школы.

Техникум предоставляет услуги обучения студентов, интересующим их специальностями.

Каждый год, у сотни групп, у сотен преподавателей, в средних специальных учебных учреждениях происходит перераспределение часов нагрузки.

Учителя должны читать дисциплины разным группам, но возможности читать всем, у них нет. Из-за чего расчет нагрузки – одно из важнейших и довольно муторных действий.

Нагрузка учителей и групп содержит в себе количество часов по определённой дисциплине, в определённой группе, у определённого учителя.

Виды запросов в информационной система «Нагрузка учителей» :

1. получить полный список всех учителей, преподающих в техникуме;
2. получить полный список всех дисциплин, читаемых в техникуме;
3. получить полный список групп, обучающихся в техникуме;
4. получить список всех дисциплин, читаемых определённым преподавателем в техникуме;
5. убрать определённого учителя из списка, вместе со всеми нагрузочными часами;
6. убрать определённую группу из списка, вместе со всеми нагрузочными часами;
7. убрать определённый предмет из списка, вместе со всеми нагрузочными часами;
8. убрать определённую нагрузку из списка;
9. добавить учителя в список преподавателей, обучающих в техникуме;
10. добавить группу в список тех, что обучаются в техникуме
11. добавить предмет в список тех, что читаются в техникуме
12. добавить часы нагрузки для определённого учителя группе по читаемой дисциплине;
13. внести изменения в инициалы определённого учителя, преподающего в техникуме;
14. внести изменения в название определённой группы, обучающейся в техникуме;
15. внести изменения в название определённой дисциплины, читаемой в техникуме;
16. внести изменения в нагрузку учителя группе по определённой дисциплине;
17. найти нагрузку определённого учителя, преподающего в техникуме.

## 1.2 Техническое задание

### 1.2.1 Общие сведения

### 1.2.1.1 Полное наименование системы и ее условное обозначение

Полное наименование системы: Автоматизированная информационная система подсчёта нагрузки учителей в техникуме

Краткое наименование системы: АИС "НАУТ".

#### 1.2.1.2 Шифр темы или шифр (номер) договора

Шифр темы: АИС-НАУТ -10

#### 1.2.1.3 Наименование предприятий разработчика и заказчика системы, их реквизиты

Полное наименование заказчика: Частное учреждение профессиональная образовательная организация техникум «Бизнес и право»

Сокращённое название: ЧУПОО «БиП».

Юридический и почтовый адрес: 352630 Краснодарский край г.Белореченск ул.Чапаева д.48 каб.204

Контактный телефон: 8-(86155)-3-39-12.

ИНН: 2310029652 КПП: 236801001.

БИК: 040349722 ОГРН: 1022301620791

Расчетный счет: 40703810600170000042.

Полное наименование разработчика: Индивидуальный предприниматель «Корниенко М.А.»

Краткое наименование: ИП"Корниенко М.А.".

Юридический и почтовый адрес: 352630, Краснодарский край, г. Белореченск, ул.Конармейская, 42а.

Контактный телефон: 8-906-401-08-30

ИНН: 3435653093.

КПП: 342341002.

Расчетный счет: 40702653800000000001

#### 1.2.1.4 Порядок оформления и предъявления заказчику результатов работ по созданию системы, ее частей и отдельных средств

Система передается в виде полностью функционирующего комплекса на базе средств вычислительной техники заказчика и исполнителя в сроки, установленные договором. Приемка системы осуществляется комиссией в составе уполномоченных представителей заказчика и исполнителя.

Весь комплект документации на АИС "НАУТ", разработанный исполнителем, передается заказчику на бумажных (в двух экземплярах) и на машинных носителях (DVD), один экземпляр после подписания передается исполнителю. Текстовые документы, передаваемые на машинных носителях, должны быть представлены в форматах PDF.

### 1.2.2 Назначение и цели создания (развития) системы

#### 1.2.2.1 Назначение системы

Разрабатываемая система предназначена для ведения картотеки в различных филиалах, выполненных по заказу ЧУПОО «БиП».

Программа будет отражать в приложении недвижимость, которая доступна для сделки. С ее помощью учителя могут с лёгкостью посчитать нагрузку учителя, группы или предмета.

#### 1.2.2.2 Цели создания системы

Основными целями создания АИС "НАУТ" являются:

* быстрая работа с таблицами и записями в них;
* доступность для любого сотрудника компании.

Реализация вышеизложенных пунктов преследуют ключевую цель - повышения прибыли предприятия.

Для достижения поставленных целей система должна решать

следующие задачи:

- наличие сформированной базы данных учителей, групп и предметов;

- добавление новых записей с нагрузкой учителей группе по читаемой дисциплине;

- поиск информации по определенным пользователями критериям.

### 1.2.3 Характеристика объектов автоматизации

В ходе проведения работ по разработке АИС "НАУТ" автоматизируются процессы компании заказчика по подсчёту нагрузки учителей в локальной сети посредством установленных клиентских приложений

Автоматизируемым процессом будет процесс подсчёта нагрузки. Система будет все записи одной категории и сортировать их по характеристикам. При выборе определенной недвижимости система будет отправлять пользователя на страницу этой недвижимости.

При внедрении системы будет достигнуто:

- простой поиск необходимых недвижимостей;

- удобное сравнение недвижимостей по характеристикам;

- повышении прибыли;

Объекты автоматизации характеризуются:

- потребностью к постоянному доступу к локальному серверу.

### 1.2.4 Требования к системе

#### 1.2.4.1 Требования к системе в целом

##### 1.2.4.1.1 Требования к структуре и функционированию системы

Рисунок 1 – Архитектура АИС

Клиент состоит из:

1. Controller – основная логика запросов и работы с сервером.

* RealtorController
* ClientController
* SuccessfulTransactionController
* SeizenController
* PropertyController
* MainController

1. Entity – класс-пустышка для обработки данных с сервера

* RealtorEntity
* ClientEntity
* SuccessfulTransactionEntity
* SeizenEntity
* PropertyEntity

1. Utils – класс для посылания запросов на сервер

* HTTPUtils

Сервер состоит из:

1. Controller –обработка запросов с клиента.

* GroupController
* LoadController
* LogicController
* TeacherController
* SubjectController

1. Entity – класс-скелет для работы с БД

* GroupEntity
* LoadEntity
* TeacherEntity
* SubjectEntity

1. Repo – класс для разграничения доступа к таблицам

* GroupRepo
* LoadRepo
* TeacherRepo
* SubjectRepo

1. Service – Для доступа к бд

* GroupService
* LoadService
* TeacherService
* SubjectService

1. Response – класс для предоставления ответа на запросы

* GroupResponse
* LoadResponse
* TeacherResponse
* SubjectResponse
* GroupListResponse
* LoadListResponse
* TeacherListResponse
* SubjectListResponse
* BaseResponse

1. Utils – класс для обработки данных с клиента

* GroupValidation
* LoadValidation
* TeacherValidation
* SubjectValidation

1. Model – класс для отбрасываний лишней информации для простоты работы с БД

* LoadModel

Для того, чтобы сделать приложение высчитывания нагрузки учителя, мы будем использовать клиент-серверную архитектуру. Чтобы убедиться, что она нам точно подходит рассмотрим ее подробнее. В клиент-серверной архитектуре имеется три звена:

- представление данных — на стороне клиента;

- прикладной компонент — на выделенном сервере приложений, здесь происходит вся бизнес-логика;

- управление ресурсами — сервер БД, который и представляет запрашиваемые данные.

Мы выбрали эту архитектуру, так как у нее есть ряд преимуществ

перед другими архитектурами:

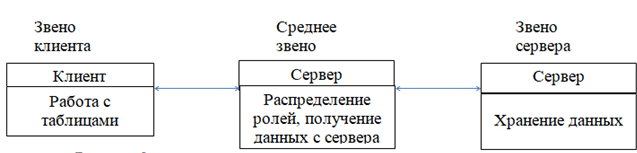
- высокую степень гибкости и масштабируемости;

- высокую безопасность;

- высокую производительность (т.к. задачи распределены междусерверами).

На рисунке 2 представлена структура клиент-серверной

архитектуры.

Рисунок 2- структура приложения

Требования к системе разработаны в соответствии со стандартом качества программного обеспечения ISO 9126:2001, описывающиммногоуровневую модель характеристик качества и соответствующийим набор атрибутов. Требования к АИС "НАУТ" представлены в таблице 1.

Таблица 1 – Требования к система АИС "НАУТ"

|  |  |  |
| --- | --- | --- |
| Атрибуты | Метрики | Требования |
| 1 | 2 | 3 |
| Удобство | | |

Продолжение таблицы 1

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| Привлекательность | Соответствие интерфейса требованиям заказчика | 97% элементов интерфейса должны быть адаптированы для пользователя и 99% для заказчика; |
| Обучаемость | Показатель, затрачиваемый пользователями на обучение работе с АИС | Пользователь должен изучить работу ПОза первые 3-5 минут использования |
| Производительность | | |
| Временная эффективность | Время выполнения компонента программы | - Приложение должно обслуживать любую страницу не дольше 300 мсек (не включая задержки в сети);  - Главная страница сайта в среднем случае должна загружаться быстрее 3-x секунд. |
| Эффективность использования ресурсов | Объемы ресурсов требуемых для выполнения задач; | - Постоянная память, используемая программой – не более 10МБ; - Приложение должно потреблять не более 4 Кбайт памяти на каждый неактивный сеанс с пользователем;  - Нагрузка на CPU и используемый объем жесткого диска на сервере баз данных не должны превышать 70%, а время обработки запросов не должно превышать 2 секунд;  - Нагрузка на CPU в режиме простоя приложения не должна превышать 1%;  - Время от возникновения ошибки до вывода предупреждения не должно превышать 25 мсек. |
| Переносимость | | |
| Удобство установки | Легкость установки | Необходима установка: .Java, SQL Server, |
| Адаптируемость | Способность ПО приспосабливаться к различным окружениям | - Программа должна корректно отображаться на всех доступных разрешениях экрана мониторов; |

Продолжение таблицы 1

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| Удобство замены другого ПО данным | Совместимость структур данных | - Система должна быть разработана по общим принципам и не требовать навыков от пользователя; - Должна поддерживать внедрение новых версий;- Система должна быть применима вместо других программных систем для решения тех же задач в определенном окружении. |

##### 1.2.4.1.2 Требования к численности и квалификации персонала системы и режиму его работы

В системе существуют 1 группа пользователей: специалисты по закупкам. Для данной группы пользователей определен набор функций. Каждая учетная запись может принадлежать только одному человеку из компании. Группа специалистов по закупке товаров имеет доступ ко всем функциям системы.

Пользователи системы должны иметь опыт работы с персональным компьютером на базе операционных систем MicrosoftWindows на уровне квалифицированного пользователя.

##### 1.2.4.1.3 Требования к надежности

В таблице 2 представлены требования к надежности.

Таблица 2 – Требования к надёжности

|  |  |  |
| --- | --- | --- |
| Атрибуты | Метрики | Требования |
| 1 | 2 | 3 |
| Належность | | |

Продолжение таблицы 2

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| Устойчивость к отказу | Коэффициент аварийных отказов | - Не менее 90% ошибок в программе должны обрабатываться без экстренного авершения; - Обеспечение бесперебойного питание активного сетевого оборудования; |
| Способность к восстановлению | Отношение количества числа успешного восстановление работы системы к ее аварийному завершению за определённый промежуток времени. | Восстановление работоспособности системы не должно превышать 5 минут |
| Уровень зрелости | Среднее время работы без сбоев | - Система должна работать 24 часа в сутки; - Система должна оповещать пользователя при некорректных действиях в системе; - Данные, которые вводит пользователь должны сохраняться. |

Средняя доступность АИС должна составлять не менее 99%

Среднее время между сбоями — это среднее время, за которое компонент или модуль может выполнять свои функции без перерыва.

Измеряется от начала работы до момента следующего сбоя.

Среднее временя работы без сбоев должно составлять не менее 1500 часов.

В таблице 3 представлен перечень и критерии отказов для каждой функции, по которой задаются требования по надежности.

Таблица 3 - Перечень и критерии отказов для каждой функции

|  |  |
| --- | --- |
| Функция | Критерии отказа |
| Просмотр всех элементов | Отсутствие соединения с базой данных |
| Сортировка списков | Отсутствие соединения с сервером |
| Редактирование таблиц | Отсутствие соединения с сервером |

#### 1.2.4.2 Требования к функциям (ВИ), выполняемым программным обеспечением

Сценарий:

1.программа выводит весь список данных

2. пользователь нажимает кнопку «создать»

3. пользователь заполняет форму создания

4. пользователь нажимает кнопку «Добавить»

5.система выводит сообщение об успешности создания

Расширения: в случае некорректного формата введенных данных, при нажатии на кнопку «Добавить», выведется сообщение с описанием ошибки.

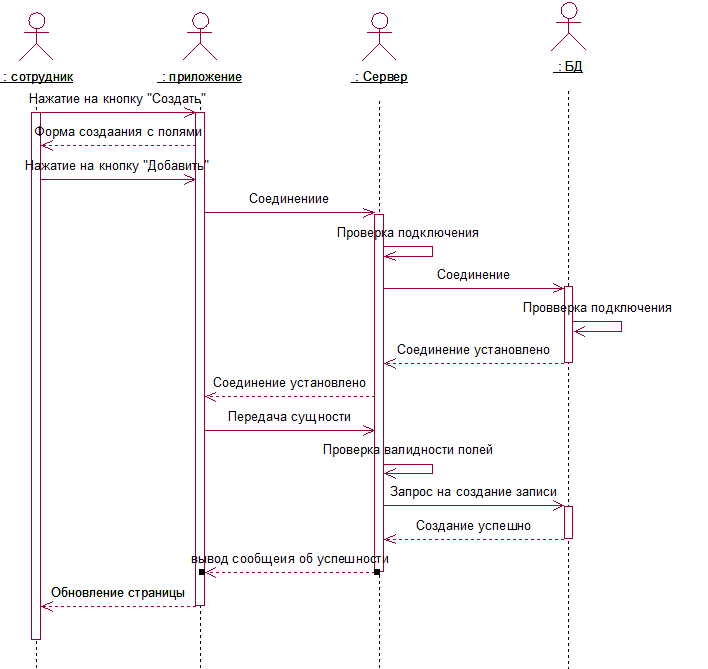


Рисунок 3 – Диаграмма последовательности сценария «Создание новой записи»

Имя: Удаление

Описание: Один из сотрудников компании заходит в клиентское приложение, чтобы удалить запись недвижимости

Действующий субъект: сотрудник

Предусловие: сотрудник зашел в приложение картотеки агентства недвижимости уже авторизированный

Сценарий:

1Пользователь выбирается запись для удаления

2 Пользователь нажимает кнопку «Удалить»

3Система выводи окно подтверждение

3.1 пользователь нажимает «да»

3.1.1 Система удаляет запись

3.1.2 Система уведомляет пользователя об успешности

3.2 Пользователь нажимает «нет»

3.2.1 Система отменяет удаление

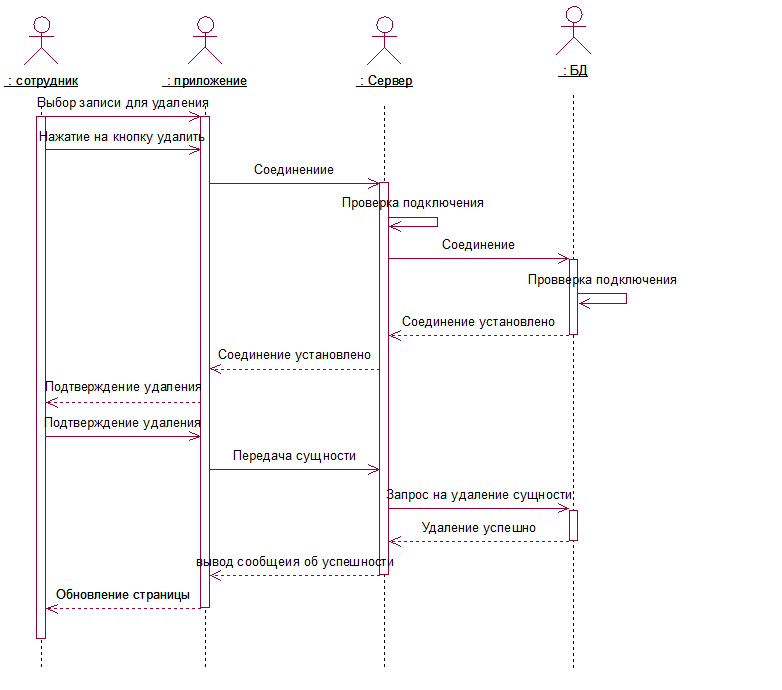
Расширения: в случае некорректного формата введенных данных, при нажатии на кнопку «Удалить», выведется сообщение с описанием ошибки.

Рисунок 4 – Диаграмма последовательности сценария «Удаление записи»

Имя: Редактирование

Описание: Один из сотрудников компании заходит в клиентское приложение чтобы изменить запись

Действующий субъект: сотрудник

Предусловие: сотрудник зашел в приложение картотеки агентства недвижимости уже авторизированный

Сценарий:

1.Пользователь выбирает нужную запись для редактирования

2. Пользователь нажимает кнопку «Редактировать»

3. Системы вызывает форму редактирования с заполненными полями

4. Пользователь нажимает «Отредактировать»

5. Система уведомляет пользователя об успешности выполнения операции

Расширения: в случае некорректного формата введенных данных, при нажатии на кнопку «Отредактировать», выведется сообщение с описанием ошибки.

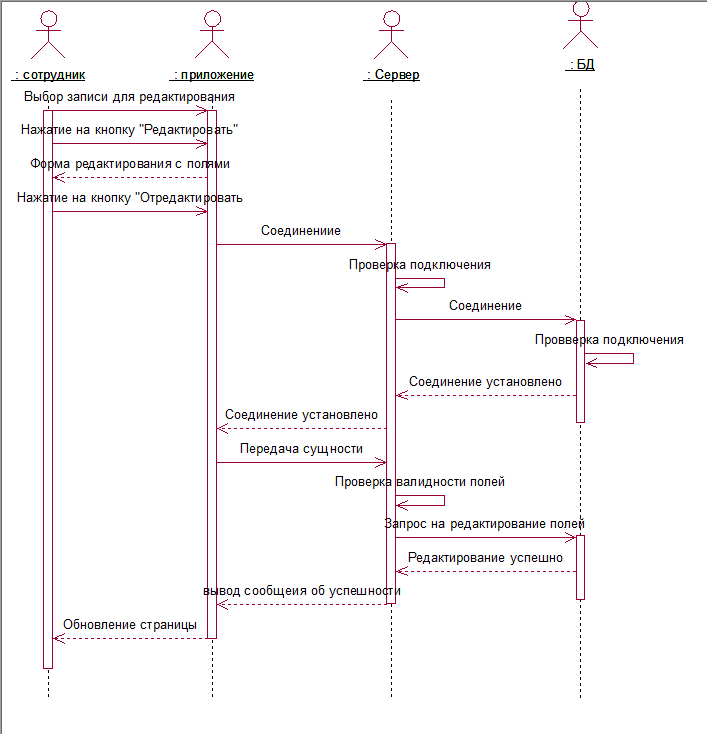


Рисунок 5 – Диаграмма последовательности сценария «Редактирование записи»

Имя: Считать, занести в базу и записать в файл

Описание: Один из сотрудников компании заходит в клиентское приложение чтобы

Действующий субъект: сотрудник

Предусловие: сотрудник зашел в приложение подсчёта нагрузки уже авторизированный

Сценарий:

1.Пользователь нажимает кнопку «Считать, занести в базу и записать в файл»

2. Система вызывает диалоговую форму с расположением файла Excel

3. Пользователь вводит расположение к файлу и нажимает “Ok”

4. Система считывает файл и сохраняет в базу

5. Система спрашивает, использовать параметры подсчёта нагрузки

5.1 При выборе «Да» считаются нужные параметры и в файл запишутся данные по заданным параметрам

5.2 При выборе «Нет» используются лишь стандартные параметры и в файл запишутся данные по стандартным параметрам

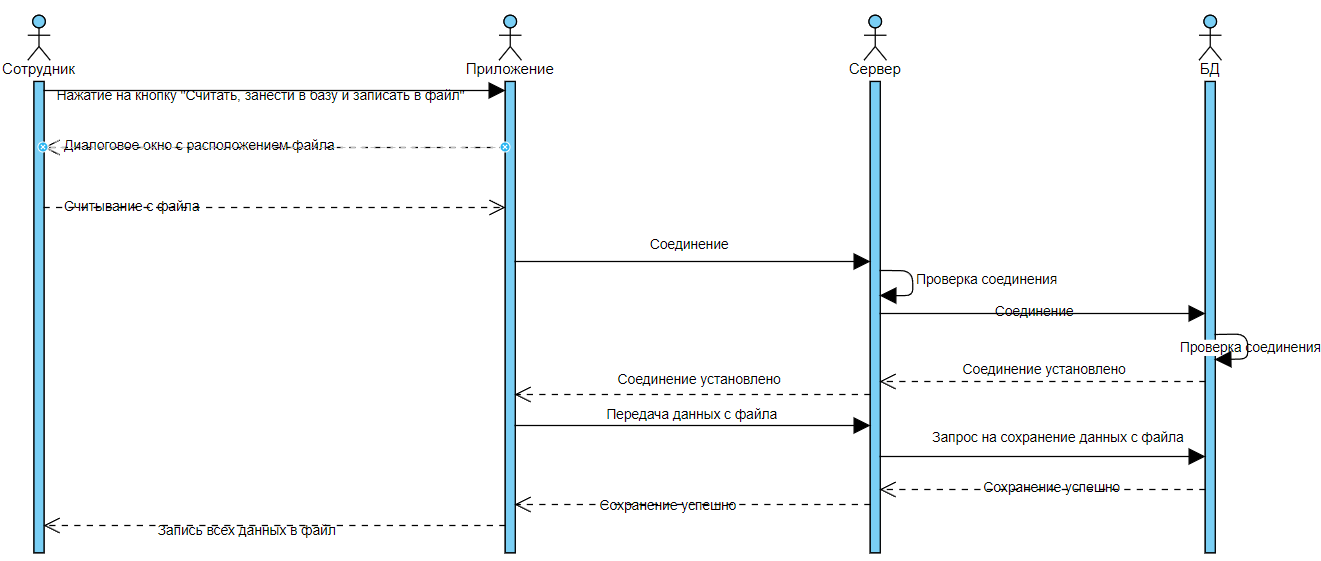
Расширения: в случае некорректного расположения файла, при нажатии на кнопку «Ok», выведется сообщение с описанием ошибки.

Рисунок 6 – Диаграмма последовательности сценария «Считывание, сохранение и запись в файл»

#### 1.2.4.3 Требования к видам обеспечения

Для функционирования системы требуется наличие сервера и клиентских компьютеров, на которых будет производится работа с системой. Перед вводом системы в действие необходимо:

- Создать в компании локальный сервер;

- Обеспечить каждого специалиста персональным компьютером;

- Установить на каждый ПК клиент последней версии;

- Создание сервера БД;

- Эксплуатация системы.

При анализе предметной области разрабатываемой системы были выделены следующие сущности, на основе которых будет разрабатываться база данных системы:

-Группы;

-Учителя;

-Предметы;

-Нагрузка учителей;

Разрабатываемая система требует в своей архитектуре наличие БД-сервера со свободной реляционной системой управления – MySQL.

База данных информационной системы должна состоять из следующих связанных таблиц:

1.Teachers – таблица содержащая информацию о клиентах.

2.Groups – таблица содержащая информацию о номерах телефонов клиентов.

3.Subjects – таблица содержащая информацию о владении недвижимостью

4.LoadOfTeachers – таблица содержащая информацию о недвижимости.

Информация об учителях должна размещаться в таблице Teachers

Таблица 4 – Таблица Teachers в базе данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя Поля | Тип | Размер | Комментарий | Ограничение |
| ID | BIGINT | 20 | ID учителя | Первичный ключ |
| initials | CHAR | 255 | Инициалы учителя | NOT NULL |

Информация о группах должна размещаться в таблице Groups

Таблица 5 – Таблица Groups в базе данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя Поля | Тип | Размер | Комментарий | Ограничение |
| ID | BIGINT | 20 | ID группы | Первичный ключ |
| groupName | CHAR | 255 | Название группы | NOT NULL |

Информация о дисциплинах должна размещаться в таблице Subjects

Таблица 6 – Таблица Subjects в базе данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя Поля | Тип | Размер | Комментарий | Ограничение |
| ID | BIGINT | 20 | ID предмета | Первичный ключи |
| subjectName | CHAR | 255 | Название предмета | NOT NULL |

Информация о риэлторах должна размещаться в таблице Realtors

Таблица 7 – Таблица Realtors в базе данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя Поля | Тип | Размер | Комментарий | Ограничение |
| 1 | 2 | 3 | 4 | 5 |
| ID | BIGINT | 20 | ID риэлтора | Первичный ключ |
| semesterFirstHours | INT | 10 | Часы первого семестра | NOT NULL |
| semesterSecondHours | INT | 10 | Часы второго семестра | NOT NULL |

Продолжение таблицы 7

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| examHours | INT | 10 | Часы за экзамен | NOT NULL |
| courseworkHours | INT | 10 | Часы за курсовую | NOT NULL |
| diplomaHours | INT | 10 | Часы за диплом | NOT NULL |
| consultationHours | INT | 10 | Часы за консультации | NOT NULL |
| teacherInLoad | BIGINT | 20 | Внешний ключ к таблице Учителя | NOT NULL |
| subjectInLoad | BIGINT | 20 | Внешний ключ к таблице Предметы | NOT NULL |
| groupInLoad | BIGINT | 20 | Внешний ключ к таблице Группы | NOT NULL |

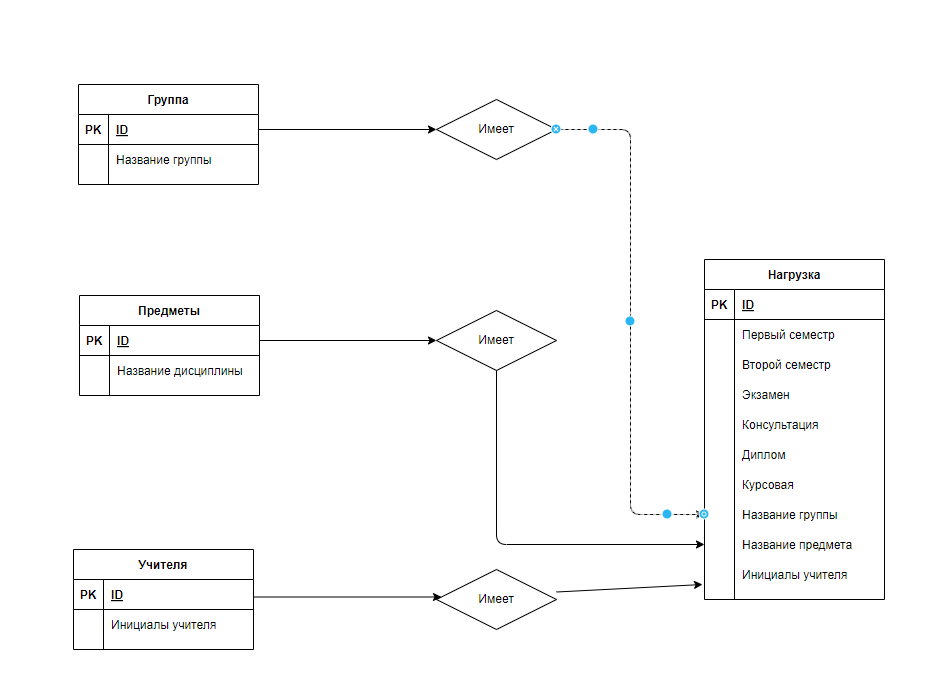
Модель базы данных в нотации Питера Чена представлена на рисунке 7:

Рисунок 7 – Модель базы данных в нотации Питера Чена

1.2.4.4 Требования к эргономике и технической эстетике

##### 1.2.4.4.1 Требования к размещению элементов управления на экранных формах

Общие требования:

- Фоновый цвет всех страниц –светло серый.

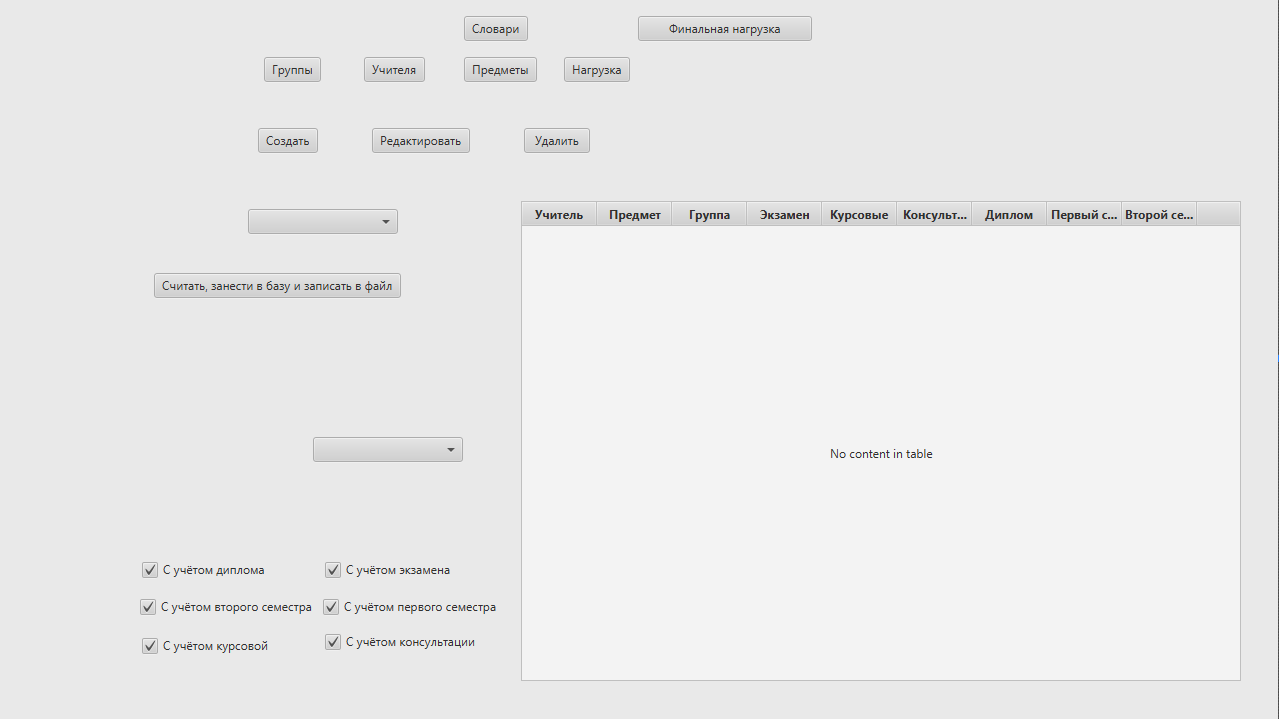
На рисунках 7-9 изображены основные окна приложения

Рисунок 7 – Главное окно приложения

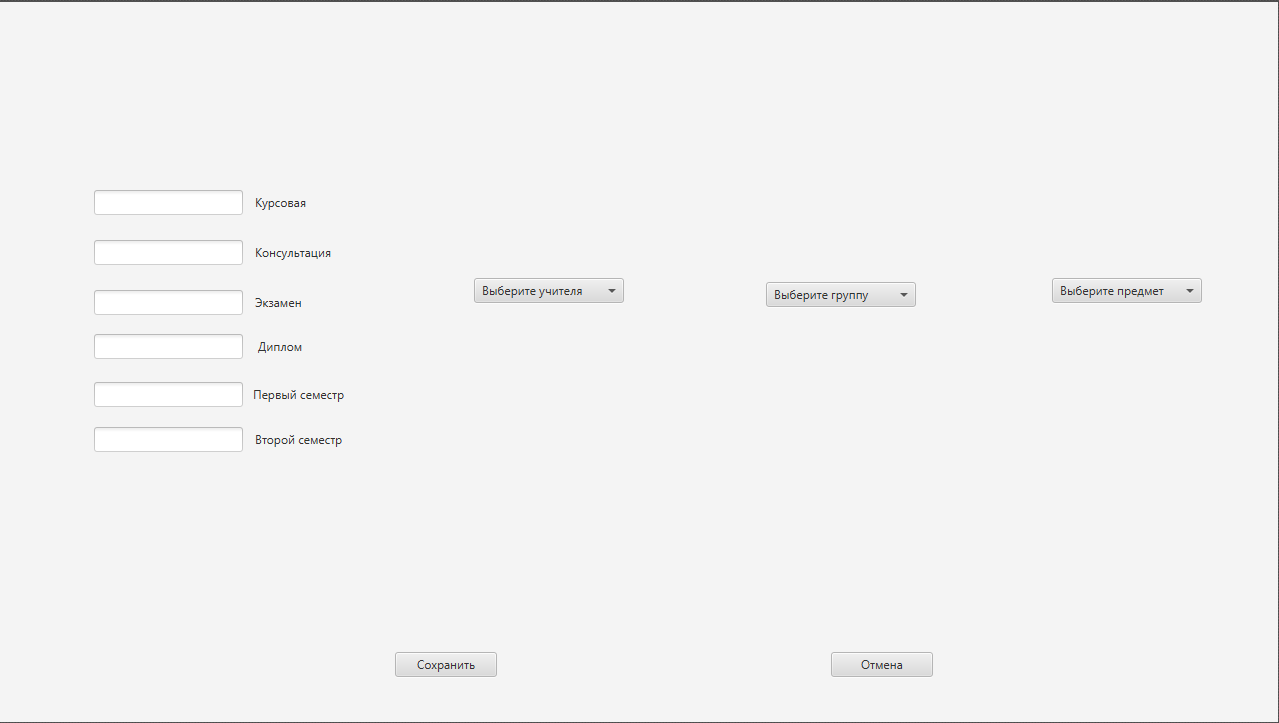
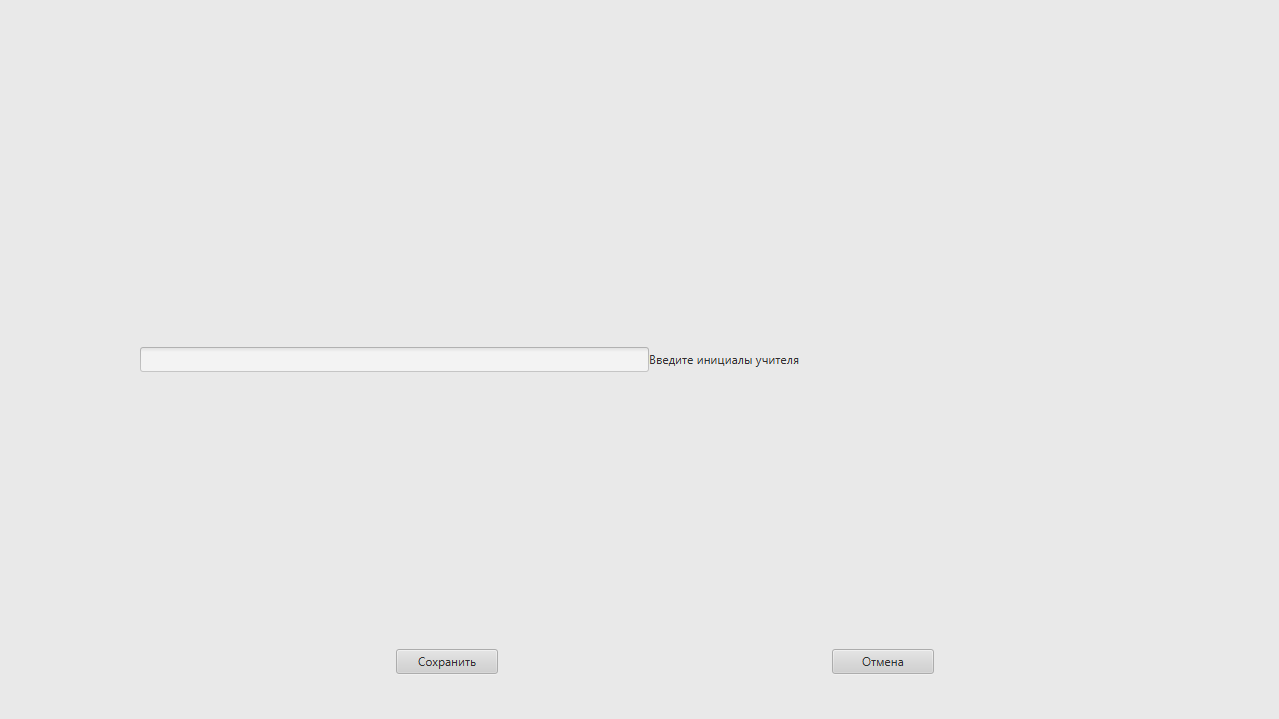
  
 Рисунок 8 – Окно добавление учителя, группы или предмета

Рисунок 9 – Окно добавление нагрузки

##### 1.2.4.3.2 Требования к видам обеспечения

Для функционирования системы требуется сервер и клиентских компьютеры, на которых будет производится запуск системы.

Требования для сервера БД:

- процессор 2 ядра (8 логических потоков), частота – 2—2,53;

- оперативная память 8 Гб и выше;

- свободное дисковое пространство не менее 3 Гб;

- HDD (жесткий диск), тип - SATA2, скорость вращения 7200 об/мин.

Требования к клиентским устройствам:

Персональные компьютеры должны быть под управлением Windows 7 или выше. Аппаратное обеспечение должно соответствовать рекомендованными требованиями операционной системы, поддерживатьApache.

Монитор на клиентских компьютерах должен быть с разрешением не ниже 1280х720 пикселей.

В качестве ОС клиентских устройств может использоваться:

- MacOS;

- Linux;

- Microsoft Windows 7 ивыше.

Для доступа к функционалу программы необходимо клиентское приложение последней версии и драйвера на компьютере:  
Javaне ниже 17JDK

Таблица 8 – Этапы Работ по созданию АИС "НАУТ"

|  |  |  |  |
| --- | --- | --- | --- |
| Стадия работ | Выполняемые работы | Сроки | Итоги выполнения работ |
| 1 | 2 | 3 | 4 |
| Формирование требований | Обследование объектов автоматизации |  | Отчет о результатах обследования; |
| Разработка технического задания на создание системы | Утверждение заказчиком ТЗ на создание системы; |

Продолжение таблицы 8

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Проектирование | Разработка технического проекта на систему  Разработка прототипа системы |  | Технический проект на систему;  Спецификации программно- аппаратных средств системы; |
| Разработка проектов организационно- распорядительной, программной и эксплуатационной документации на систему |
| Поставка программно- технических средств для эксплуатации | Поставка программно- технических средств для опытной эксплуатации на объектах автоматизации, входящих в состав опытной зоны |  | Акты; |
| Разработка программных средств | Разработка, отладка и тестирование программных средств Подсистемы |  | Программные средства на цифровых носителях;  Комплект проектов организационно- распорядительной , программной и эксплуатационной документации на систему; |
| Приемка работ | Проведение предварительных испытаний на стенде исполнителя |  | Протоколы испытаний;  Акт готовности подсистемы к развертыванию в опытной зоне. |

### 1.2.5 Порядок контроля и приемки системы

Система передается в виде полностью функционирующего комплекса на базе средств вычислительной техники заказчика и исполнителя в сроки, установленные договором. Приемка системы осуществляется комиссией в составе уполномоченных представителей заказчика и исполнителя.

Весь комплект документации на АИС "НАУТ", разработанный исполнителем, передается заказчику на бумажных (в двух экземплярах) и на машинных носителях (DVD), один экземпляр после подписания передается исполнителю. Текстовые документы, передаваемые на машинных носителях, должны быть представлены в форматах PDF.

### 1.2.6 Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие

Заказчик обязан предоставить сервер и персональные компьютеры, требования к которым указаны в разделе 1.2.4.3 «Требования к видам обеспечения».

Необходимое программное обеспечение для сервера:

- операционная система: Windows 10/Linux/macOS;

- сервер:Apache;

- СУБД: MySQL;

- java не ниже 17JDK.

Необходимое программное обеспечение для персонального компьютера:

- операционная система: Windows 10/Linux/macOS;

- Apache;

- java не ниже 17JDK.

### 1.2.7 Требования к документированию

Перечень подлежащих разработке комплектов и видов документов:

- описание бизнес-процессов предметной области;

- техническое задание;

- эскизный проект;

- программа и методика испытаний;

- графический материал.

### 1.2.8 Источники разработки

Источниками разработки настоящего Технического Задания являются:

- перечень документации, представленный в разделе 1.2.1.2 «Основания для проведения работ»;

- ГОСТ 34.602-89 Техническое задание на создание автоматизированной системы;

- ГОСТ 24.601-86 Автоматизированные системы. Стадии создания;

- ГОСТ Р ИСО/МЭК 9126-93. Информационная технология. Оценка программной продукции. Характеристики качества и руководства по их применению;

- ГОСТ 24.601 Исследование и обоснование создания АС;

- ГОСТ РД 50-34.698-90 Руководство пользователя;

- ГОСТ 19.504-79 Руководство программиста;

- ГОСТ 2.119-2013 Эскизный проект.

## 1.3 Эскизный проект

### 1.3.1 Общие сведения

#### 1.3.1.1 Полное наименование системы и ее условное обозначение

Полное наименование системы: Автоматизированная информационная система подсчёта нагрузки учителей в техникуме

Краткое наименование системы: АИС "НАУТ".

#### 1.3.1.2 Шифр темы или шифр (номер) договора

Шифр темы: АИС-НАУТ -10

#### 1.3.1.3 Наименование предприятий разработчика и заказчика системы, их реквизиты

Полное наименование заказчика: Частное учреждение профессиональная образовательная организация техникум «Бизнес и право»

Сокращённое название: ЧУПОО «БиП».

Юридический и почтовый адрес: 352630 Краснодарский край г.Белореченск ул.Чапаева д.48 каб.204

Контактный телефон: 8-(86155)-3-39-12.

ИНН: 2310029652 КПП: 236801001.

БИК: 040349722 ОГРН: 1022301620791

Расчетный счет: 40703810600170000042.

Полное наименование разработчика: Индивидуальный предприниматель «Корниенко М.А.»

Краткое наименование: ИП"Корниенко М.А.".

Юридический и почтовый адрес: 352630, Краснодарский край, г. Белореченск, ул.Конармейская, 42а.

Контактный телефон: 8-906-401-08-30

ИНН: 3435653093.

КПП: 342341002.

Расчетный счет: 40702653800000000001

#### 1.3.1.4 Порядок оформления и предъявления заказчику результатов работ по созданию системы, ее частей и отдельных средств

Система передается в виде полностью функционирующего комплекса на базе средств вычислительной техники заказчика и исполнителя в сроки, установленные договором. Приемка системы осуществляется комиссией в составе уполномоченных представителей заказчика и исполнителя.

Весь комплект документации на АИС "НАУТ", разработанный исполнителем, передается заказчику на бумажных (в двух экземплярах) и на машинных носителях (DVD), один экземпляр после подписания передается исполнителю. Текстовые документы, передаваемые на машинных носителях, должны быть представлены в форматах PDF.

### 1.3.2 Назначение и цели создания (развития) системы

#### 1.3.2.1 Назначение системы

Разрабатываемая система предназначена для ведения картотеки в различных филиалах, выполненных по заказу ЧУПОО «БиП».

Программа будет отражать в приложении недвижимость, которая доступна для сделки. С ее помощью учителя могут с лёгкостью посчитать нагрузку учителя, группы или предмета.

#### 1.3.2.2 Цели создания системы

Основными целями создания АИС "НАУТ" являются:

* быстрая работа с таблицами и записями в них;
* доступность для любого сотрудника компании.

Реализация вышеизложенных пунктов преследуют ключевую цель - повышения прибыли предприятия.

Для достижения поставленных целей система должна решать

### 1.3.3 Описание процесса деятельности

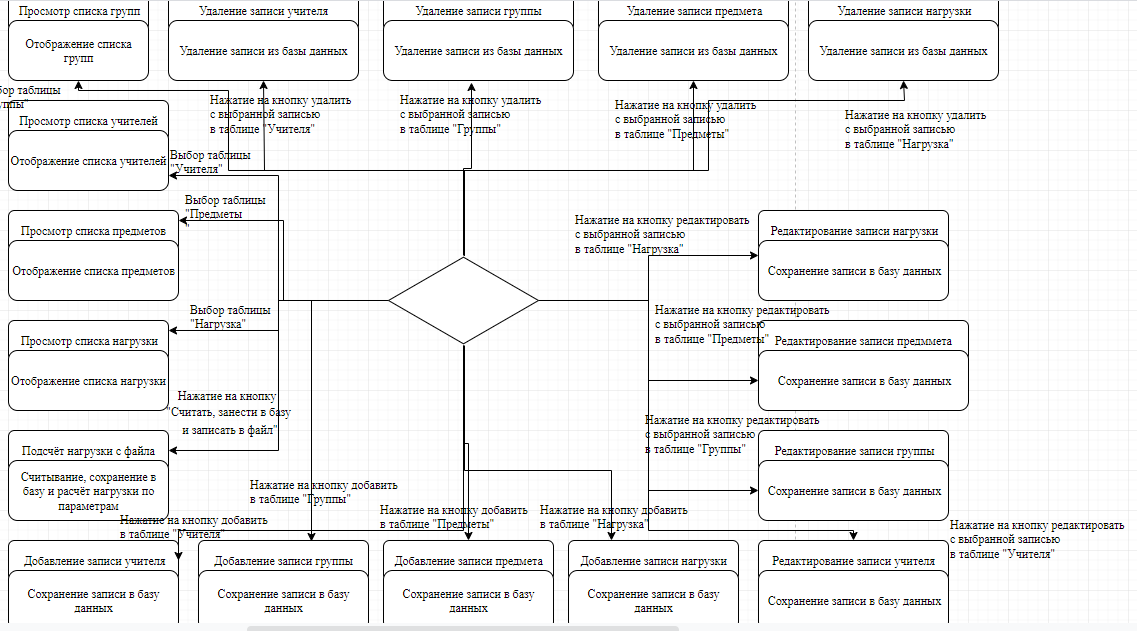
Для достижения поставленных целей система должна решать следующие задачи:

- наличие сформированной базы данных нагрузки учителей техникума;

- добавление новых записей в любую из таблиц;

На данном этапе разберем, то как будем вести себя система при выполнение выделенных модулей. Для этого мы будем использовать диаграммы состояний в нотации UML.

На рисунке 10 представлена общая диаграмма состояний. При заходе пользователем в программу АИС "НАУТ", пользователь будет видеть главное окно, с которого он уже может осуществлять деятельность, согласно выделенным вариантам

Рисунок 10 – Диаграмма состояний работы пользователя с системой

Клиент - серверная архитектура

В клиент-серверной архитектуре имеется три звена:

- Представление данных — на стороне клиента.

- Прикладной компонент — на выделенном сервере приложений,

здесь происходит вся бизнес-логика (как вариант, выполняющем

функции промежуточного ПО).

- Управление ресурсами —сервер БД, который и представляет

запрашиваемые данные.

Преимущества данной архитектуры:

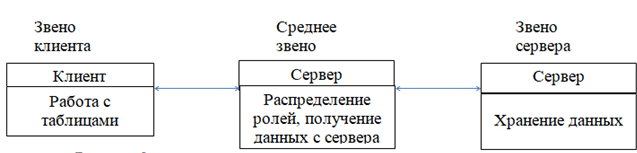
- Высокую степень гибкости и масштабируемости.

- Высокую безопасность (т.к. защиту можно определить для

каждого сервиса или уровня).

- Высокую производительность (т.к. задачи распределены между

серверами).

Схема клиент - серверной архитектуры представлена на

рисунке 11

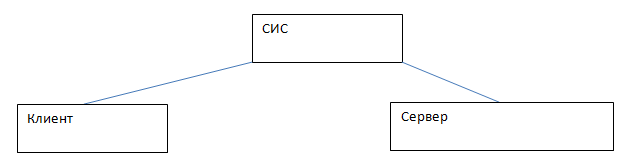
Рисунок 11 – Клиент серверная архитектура

Рисунок 12 – Архитектура АИС

Клиент состоит из:

1. Controller – основная логика запросов и работы с сервером.

* RealtorController
* ClientController
* SuccessfulTransactionController
* SeizenController
* PropertyController
* MainController

1. Entity – класс-пустышка для обработки данных с сервера

* RealtorEntity
* ClientEntity
* SuccessfulTransactionEntity
* SeizenEntity
* PropertyEntity

1. Utils – класс для посылания запросов на сервер

* HTTPUtils

Сервер состоит из:

1. Controller –обработка запросов с клиента.

* GroupController
* LoadController
* LogicController
* TeacherController
* SubjectController

1. Entity – класс-скелет для работы с БД

* GroupEntity
* LoadEntity
* TeacherEntity
* SubjectEntity

1. Repo – класс для разграничения доступа к таблицам

* GroupRepo
* LoadRepo
* TeacherRepo
* SubjectRepo

1. Service – Для доступа к бд

* GroupService
* LoadService
* TeacherService
* SubjectService

1. Response – класс для предоставления ответа на запросы

* GroupResponse
* LoadResponse
* TeacherResponse
* SubjectResponse
* GroupListResponse
* LoadListResponse
* TeacherListResponse
* SubjectListResponse
* BaseResponse

1. Utils – класс для обработки данных с клиента

* GroupValidation
* LoadValidation
* TeacherValidation
* SubjectValidation

1. Model – класс для отбрасываний лишней информации для простоты работы с БД

* LoadModel

#### 1.3.3.1 Решения по режимам функционирования,диагностированию работы системы

Средняя доступность АИС должна составлять не менее 99%

Среднее время между сбоями — это среднее время, за которое компонент или модуль может выполнять свои функции без перерыва.

Измеряется от начала работы до момента следующего сбоя.

Среднее временя работы без сбоев должно составлять не менее 1500 часов.

#### 1.3.3.2 Решения по численности, квалификации и функциям персонала АС, режимам его работы, порядку взаимодействия

В состав персонала, необходимого для обеспечения эксплуатации АИС "НАУТ" в рамках соответствующих подразделений Заказчика, необходимо выделение следующих ответственных лиц, представленных в таблице 8:

Таблица 9 - Состав персонала, необходимого для обеспечения эксплуатации АИС "НАУТ"

|  |  |  |
| --- | --- | --- |
| Должность | Количество | Подсистема |
| Адбминистратор | 1 | Администрирование БД |

Данные лица должны выполнять следующие функциональныеобязанности:

* Администратор
  + внесение правок в базу данных.

#### 1.3.3.3 Сведения об обеспечении заданных в техническом задании

(ТЗ) потребительских характеристик системы (подсистем), определяющих ее качество

Обеспечение заявленных требований будет обеспеченоследующим образом:

* Требования к надежности обеспечиваются использованием отказоустойчивых решений, реализованных в составе Apache;
* Требования к безопасности обеспечиваются использованием(возможностью использования) защищенных протоколов и средствами обеспечения безопасности, реализованными в Apache.
* Требования к эргономике и технической эстетике обеспечиваются использованием JavaFx для создания графического интерфейса.

#### 1.3.3.4 Состав функций, комплексов задач, реализуемых системой(подсистемой)

Выделенные варианты сценариев использования системы и

соответствующие им диаграммы последовательностей приведены в

разделе 1.2.4.2 «Требования к функциям, выполняемым ПО».

#### 1.3.3.5 Решения по комплексу технических средств, егоразмещению на объекте

Перед вводом системы в действие необходимо:

* Провести в компании интернет-соединение.
* Обеспечить каждого специалиста персональным компьютером.
* Установить на каждый ПК любой браузер последней версии
* - Аренда хостинга и размещение на нем web-сервера и сервера БД.
* - Изучение персоналом документа “Руководства пользователя”.
* - Эксплуатация системы.

Информация об учителях должна размещаться в таблице Teachers

Таблица 10 – Таблица Teachers в базе данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя Поля | Тип | Размер | Комментарий | Ограничение |
| ID | BIGINT | 20 | ID учителя | Первичный ключ |
| initials | CHAR | 255 | Инициалы учителя | NOT NULL |

Информация о группах должна размещаться в таблице Groups

Таблица 11 – Таблица Groups в базе данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя Поля | Тип | Размер | Комментарий | Ограничение |
| ID | BIGINT | 20 | ID группы | Первичный ключ |
| groupName | CHAR | 255 | Название группы | NOT NULL |

Информация о дисциплинах должна размещаться в таблице Subjects

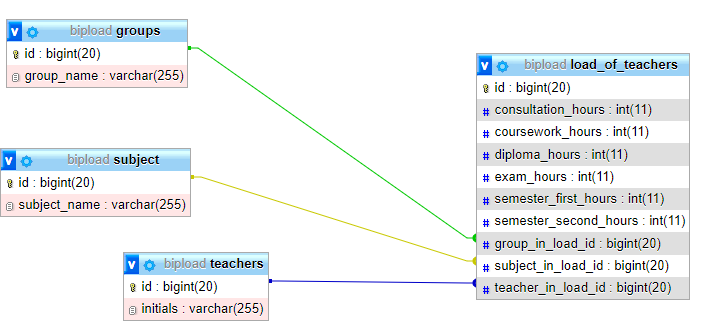
Таблица 12 – Таблица Subjects в базе данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя Поля | Тип | Размер | Комментарий | Ограничение |
| ID | BIGINT | 20 | ID предмета | Первичный ключи |
| subjectName | CHAR | 255 | Название предмета | NOT NULL |

Информация о риэлторах должна размещаться в таблице Realtors

Таблица 13 – Таблица Realtors в базе данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя Поля | Тип | Размер | Комментарий | Ограничение |
| ID | BIGINT | 20 | ID риэлтора | Первичный ключ |
| semesterFirstHours | INT | 10 | Часы первого семестра | NOT NULL |
| semesterSecondHours | INT | 10 | Часы второго семестра | NOT NULL |
| examHours | INT | 10 | Часы за экзамен | NOT NULL |
| courseworkHours | INT | 10 | Часы за курсовую | NOT NULL |
| diplomaHours | INT | 10 | Часы за диплом | NOT NULL |
| consultationHours | INT | 10 | Часы за консультации | NOT NULL |
| teacherInLoad | BIGINT | 20 | Внешний ключ к таблице Учителя | NOT NULL |
| subjectInLoad | BIGINT | 20 | Внешний ключ к таблице Предметы | NOT NULL |
| groupInLoad | BIGINT | 20 | Внешний ключ к таблице Группы | NOT NULL |

Рисунок 13 – физическая модель базы данных

#### 1.3.3.6 Решения по составу программных средств, языкам деятельности, алгоритмам процедур и операций и методам их реализации

При разработке ИС использованы следующие программные

компоненты:

-SceneBuilder– интерфейс

- Хранилище данных создано на базе MySQL

- IntelliJIDE

В качестве ОС клиентских устройств может использоваться:

- MicrosoftWindows 10

- Linux

- macOS

Заказчик обязан предоставить сервер и персональные компьютеры, требования к которым указаны в разделе 1.2.4.3 «Требования к видам обеспечения».

Необходимое программное обеспечение для сервера:

- Операционная система: Windows 10/Linux/macOS

- Сервер: Apache

- СУБД: MySQL

- Java не ниже 17JDK

Необходимое программное обеспечение для персонального компьютера:

- Операционнаясистема: Windows 10/Linux/macOS

-Javaне ниже 17JDK

#### 1.3.3.7 Эскиз пользовательского интерфейса АИС

Эскизы пользовательского интерфейса представлены в разделе 1.2.4.4.1 «Эскизы основных окон системы».

# 2 РАЗРАБОТКА И ДОКУМЕНТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ

## 2.1 Документирование API

Таблица 14 – POST запросы

|  |  |
| --- | --- |
| Тело И запрос | Ответ |
| 1 | 2 |
| Перевод данных из файла в базу данных | |
| http://localhost:28242/bipapi/fromexcel/conversion  [{ "semesterFirstHours" : 52,  "semesterSecondHours" : 25,  "examHours" : 15,  "courseworkHours" : 35,  "diplomaHours" : 26,  "consultationHours" : 54,  "teacherInLoad" : {"id" : 1,  "initials" : "Корниенко М.А." },  "subjectInLoad" : {"id" : 1,  "subjectName" : "МДК 08.01"},  "groupInLoad" : {"id" : 1,  "groupName" : "ИС2"}},{  "semesterFirstHours" : 40,  "semesterSecondHours" : 20,  "examHours" : 10,  "courseworkHours" : 40,  "diplomaHours" : 26,  "consultationHours" : 54,  "teacherInLoad" : {"id" : 1,  "initials" : "Корниенко М.А."},  "subjectInLoad" : {"id" : 1,  "subjectName" : "МДК 08.01"},  "groupInLoad" : {"id" : 1,  "groupName" : "ИС-2"}}] | {  "message": "Из файла были добавлены все записи",  "success": true  } |
| Добавление записи в таблицу Groups | |
| http://localhost:28242/bipapi/groups/add  {  "groupName" : "Д-4"  } | {  "message": "Группа добавлена",  "success": true  } |
| Добавление записи в таблицу Subjects |  |

Продолжение таблицы 14

|  |  |
| --- | --- |
| 1 | 2 |
| http://localhost:28242/bipapi/subjects/add  {  "subjectName" : "МДК 05.01"  } | {  "message": "Предмет добавлена",  "success": true  } |
| Добавление записи в таблицу Teachers | |
| http://localhost:28242/bipapi/teachers/add  {  "initials" : "Мартыненко В.Ч."  } | {  "message": "Учитель добавлен",  "success": true  } |
| Добавление записи в таблицу LoadOfTeachers | |
| <http://localhost:28242/bipapi/loads/add>  {  "semesterFirstHours" : 40,  "semesterSecondHours" : 20,  "examHours" : 10,  "courseworkHours" : 40,  "diplomaHours" : 26,  "consultationHours" : 54,  "teacherInLoad" : "Корниенко М.А.",  "subjectInLoad" : "МДК 08.01",  "groupInLoad" : "ИС-2"  } | {  "message": "Нагрузка добавлена",  "success": true  } |

Таблица 15 – GET запросы

|  |  |
| --- | --- |
| Тело И запрос | Ответ |
| 1 | 2 |
| Получить все группы | |

Продолжение таблицы 15

|  |  |
| --- | --- |
| 1 | 2 |
| http://localhost:28242/bipapi/groups/all | {  "message": "Список всех групп:",  "success": true,  "data": [  {  "id": 1,  "groupName": "ИС-3"  },  {  "id": 2,  "groupName": "Д-4"  }  ]  } |
| Получить группу по ID | |
| http://localhost:28242/bipapi/groups/byId/?id=1 | {  "message": "Группа по id",  "success": true,  "data": {  "id": 1,  "groupName": "ИС-3"  }  } |
| Получить всю нагрузку | |

Продолжение таблицы 15

|  |  |
| --- | --- |
| 1 | 2 |
| http://localhost:28242/bipapi/loads/all | { "message": "Список всей нагрузки:", "success": true,"data": [{ "id": 1, "semesterFirstHours": 52, "semesterSecondHours": 25,"examHours": 15,"courseworkHours": 35,"diplomaHours": 26,"consultationHours": 54,"teacherInLoad":"Корниенко М.А.","subjectInLoad": "МДК 08.01","groupInLoad": "ИС-2"},{ "id": 2,"semesterFirstHours": 40,"semesterSecondHours": 20,"examHours": 10,"courseworkHours": 40,"diplomaHours": 26"consultationHours": 54,"teacherInLoad": "Корниенко М.А.","subjectInLoad": "МДК 08.01","groupInLoad": "ИС-2" },  {"id": 3,"semesterFirstHours": 40,"semesterSecondHours": 20,  "examHours": 10,"courseworkHours": 40,"diplomaHours": 26,"consultationHours": 54,"teacherInLoad": "Корниенко М.А.","subjectInLoad": "МДК 08.01","groupInLoad": "ИС-2" }]} |
| Получить нагрузку по ID | |

Продолжение таблицы 15

|  |  |
| --- | --- |
| 1 | 2 |
| http://localhost:28242/bipapi/loads/byId/?id=22 | {  "message": "Группа по id",  "success": true,  "data": {  "id": 22,  "semesterFirstHours": 40,  "semesterSecondHours": 20,  "examHours": 10,  "courseworkHours": 40,  "diplomaHours": 26,  "consultationHours": 54,  "teacherInLoad": {  "id": 10,  "initials": "Смольняков А.В."  },  "subjectInLoad": {  "id": 3,  "subjectName": "МДК 08.03"  },  "groupInLoad": {  "id": 1,  "groupName": "ИС-3"  }  }  } |
| Получить все предметы | |
| http://localhost:28242/bipapi/subjects/all | {"message": "Список всех предметов:","success": true,  "data": [{"id": 2045,"subjectName": "МДК 08.01"}]} |
| Получить предмет по ID | |

Продолжение таблицы 15

|  |  |
| --- | --- |
| http://localhost:28242/bipapi/subjects/byId/?id=3 | {  "message": "Предмет по id",  "success": true,  "data": {  "id": 3,  "subjectName": "МДК 08.03"  }  } |
| Получение всех учителей | |
| http://localhost:28242/bipapi/teachers/all | {  "message": "Список всех учителей:",  "success": true,  "data": [  {  "id": 470,  "initials": "Корниенко М.А."  }  ]  } |
| Получение учителя по ID | |
| http://localhost:28242/bipapi/teachers/byId/?id=9 | {  "message": "Учитель по id",  "success": true,  "data": {  "id": 9,  "initials": "Корниенко М.А."  }  } |

Таблица 16 – DELETE запросы

|  |  |
| --- | --- |
| Тело и запрос | Ответ |
| 1 | 2 |
| Удаление записи в таблице Groups | |

Продолжение таблицы 16

|  |  |
| --- | --- |
| 1 | 2 |
| http://localhost:28242/bipapi/groups/delete617 | {  "message": "Группа удалена",  "success": true  } |
| Удаление записи в таблице LoadOfTeachers | |
| http://localhost:28242/bipapi/loads/delete6 | {  "message": "Нагрузка удалёна",  "success": true  } |
| Удаление записи в таблице Subjects | |
| http://localhost:28242/bipapi/subjects/delete2045 | {  "message": "Предмет удалён",  "success": true  } |
| Удаление записи в таблице Teachers | |
| http://localhost:28242/bipapi/teachers/delete470 | {  "message": "Учитель удалён",  "success": true  } |

## 2.2 Тестирование клиентской части

Таблица 17 – Тестирование клиента без сервера

|  |  |  |
| --- | --- | --- |
| Действие | Ожидание | Реальность |
| 1 | 2 | 3 |
| Инициализация таблицы Groups | Сообщение об ошибке | Сообщение об ошибке |
| Инициализация таблицы Teachers | Сообщение об ошибке | Сообщение об ошибке |
| Инициализация таблицы Subjects | Сообщение об ошибке | Сообщение об ошибке |
| Инициализация таблицы LoadOfTeachers | Сообщение об ошибке | Сообщение об ошибке |
| Добавление записи в таблицу Groups | Сообщение об ошибке | Сообщение об ошибке |

Продолжение таблицы 17

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| Добавление записи в таблицу LoadOfTeachers | Сообщение об ошибке | Сообщение об ошибке |
| Добавление записи в таблицу Subjects | Сообщение об ошибке | Сообщение об ошибке |
| Добавление записи в таблицу Teachers | Сообщение об ошибке | Сообщение об ошибке |
| Редактирование записи в таблице LoadOfTeachers без выбора записи | Сообщение об ошибке | Сообщение об ошибке |
| Редактирование записи в таблице Groups | Сообщение об ошибке | Сообщение об ошибке |
| Редактирование записи в таблице Teachers без выбора записи | Сообщение об ошибке | Сообщение об ошибке |
| Редактирование записи в таблице Teachers | Сообщение об ошибке | Сообщение об ошибке |
| Редактирование записи в таблице Subjects без выбора записи | Сообщение об ошибке | Сообщение об ошибке |
| Редактирование записи в таблице Subjects | Сообщение об ошибке | Сообщение об ошибке |
| Редактирование записи в таблице LoadOfTeachers без выбора записи | Сообщение об ошибке | Сообщение об ошибке |
| Редактирование записи в таблице LoadOfTeachers | Сообщение об ошибке | Сообщение об ошибке |
| Удаление записи в таблице Groups без выбора записи | Сообщение об ошибке | Сообщение об ошибке |
| Удаление записи в таблице Groups | Сообщение об ошибке | Сообщение об ошибке |
| Удаление записи в таблице Teachers без выбора записи | Сообщение об ошибке | Сообщение об ошибке |
| Удаление записи в таблице Teachers | Сообщение об ошибке | Сообщение об ошибке |
| Удаление записи в таблице Subjects без выбора записи | Сообщение об ошибке | Сообщение об ошибке |

Продолжение таблицы 17

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| Удаление записи в таблице Subjects | Сообщение об ошибке | Сообщение об ошибке |
| Удаление записи в таблице LoadOfTeachers без выбора записи | Сообщение об ошибке | Сообщение об ошибке |
| Удаление записи в таблице LoadOfTeachers | Сообщение об ошибке | Сообщение об ошибке |
| Перенос записей в базу данных без выбора файла | Сообщение об ошибке | Сообщение об ошибке |
| Перенос записей в базу данных | Сообщение об ошибке | Сообщение об ошибке |

Таблица 18 – Тестирование клиента

|  |  |  |
| --- | --- | --- |
| Действие | Ожидание | Реальность |
| 1 | 2 | 3 |
| Инициализация таблицы Groups | Выведет все записи | Вывело все записи |
| Инициализация таблицы Teachers | Выведет все записи | Вывело все записи |
| Инициализация таблицы Subjects | Выведет все записи | Вывело все записи |
| Инициализация таблицы Loads | Выведет все записи | Вывело все записи |
| Добавление записи в таблицу Groups | Запись будет добавлена | Запись была добавлена |
| Добавление записи в таблицу Groups | Запись будет добавлена | Запись была добавлена |
| Добавление записи в таблицу Groups | Запись будет добавлена | Запись была добавлена |
| Добавление записи в таблицу Groups | Запись будет добавлена | Запись была добавлена |
| Редактирование записи в таблице Groups без выбора записи | Сообщение об ошибке | Сообщение об ошибке |
| Редактирование записи в таблице Groups | Запись будет отредактирована | Запись была отредактирована |

Продолжение таблицы 18

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| Редактирование записи в таблице Teachers без выбора записи | Сообщение об ошибке | Сообщение об ошибке |
| Редактирование записи в таблице Teachers | Запись будет отредактирована | Запись была отредактирована |
| Редактирование записи в таблице Subjects без выбора записи | Сообщение об ошибке | Сообщение об ошибке |
| Редактирование записи в таблице Subjects | Запись будет отредактирована | Запись была отредактирована |
| Редактирование записи в таблице LoadOfTeachers без выбора записи | Сообщение об ошибке | Сообщение об ошибке |
| Редактирование записи в таблице LoadOfTeachers | Запись будет отредактирована | Запись была отредактирована |
| Удаление записи в таблице Groups без выбора записи | Сообщение об ошибке | Сообщение об ошибке |
| Удаление записи в таблице Groups | Запись будет удалена | Запись была удалена |
| Удаление записи в таблице Teachers без выбора записи | Сообщение об ошибке | Сообщение об ошибке |
| Удаление записи в таблице Teachers | Запись будет удалена | Запись была удалена |
| Удаление записи в таблице Subjects без выбора записи | Сообщение об ошибке | Сообщение об ошибке |
| Удаление записи в таблице Subjects | Запись будет удалена | Запись была удалена |
| Удаление записи в таблице LoadOfTeachers без выбора записи | Сообщение об ошибке | Сообщение об ошибке |
| Удаление записи в таблице LoadOfTeachers | Запись будет удалена | Запись была удалена |
| Перенос записей в базу данных без выбора файла | Сообщение об ошибке | Сообщение об ошибке |
| Перенос записей в БД | Файл успешно занесётся | Файл успешно занесён |

# ЗАКЛЮЧЕНИЕ

По итогам исследовательской деятельности было достигнуто:

* Создан сервер, обрабатывающий запросы и работающий с базой данных
* Создан клиент для удобной работы с сервером
* Обработаны большинство ошибок и предупреждений

По итогам настоящей работы не было достигнуто:

* Проверка целостности данных(правильный ввод данных)
* Оптимизация приложения для быстрой и лёгкой работы без зависаний
* Красивый и лаконичный дизайн

В процессе исследований были закреплены и систематизированы знания, полученные на дисциплинах "Основы проектирования баз данных", "Разработка и эксплуатация информационных систем", создание ИС, "Компьютерная графика", распределение кнопок по формам и создание интуитивно понятного интерфейса.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Кристиан Б., Гэвин К., Гэри Грегори.. Java Persistence API with Hibernate2019. № 6 (164). С. 16-25.
2. Машинин Т. JavaFX 2.0. Разработка RIA-приложений, 2019. С. 63-68.
3. Шилдт Г. Java. Руководство для начинающих. Современные методы создания, компиляции и выполнения программ на Java. 2019. С. 84-87.
4. Джошуа Б. Java. Эффективное программирование. 2019. С. 244-250.
5. Сьерра К., Бэйтс Б. Изучаем Java. 2017. С. 58-65.
6. Прохоренок Н.А. JavaFX, 2020.
7. Готц Б., Ли Д.JavaConcurrency на практике, 2020
8. Валерий Красько. Блог «Как измерять ROI и зачем это делать». – Режим доступа: https://netpeak.net/ru/blog/kak-izmeryat-roi-i-zachem-eto-delat/ (дата обращения 28.05.2017)
9. «Жилье станет доступнее к 2020 году». – Режим доступа: http://expert.ru/2012/12/4/zhile-stanet-dostupnee-k-2020-godu/ (дата обращения 28.05.2017)
10. Официальный сайт «Минстрой России», «Объем финансирования ФЦП Жилище». – Режим доступа: http://www.minstroyrf.ru/press/obem-finansirovaniyaftsp-zhilishche-v-2016-godu/ (дата обращения 30.05.2017)
11. Официальный сайт «Минстрой России», «Развитие рынка ипотечного кредитования». – Режим доступа: http://www.minstroyrf.ru/press/razvitie-rynkaipotechnogo-kreditovaniya-itogi-2016-goda/ (дата обращения 02.06.2017)

ПРИЛОЖЕНИЕ

GroupController.java

package practice.server.practiceServer.controller;import lombok.AllArgsConstructor;import org.springframework.http.ResponseEntity;import org.springframework.web.bind.annotation.\*;import practice.server.practiceServer.entity.GroupEntity;import practice.server.practiceServer.exception.GroupExceptionValidation;import practice.server.practiceServer.response.BaseResponse;import practice.server.practiceServer.response.GroupListResponse;import practice.server.practiceServer.response.GroupResponse;import practice.server.practiceServer.service.GroupService;import practice.server.practiceServer.utils.GroupValidation;import java.util.ArrayList;import java.util.List;@RestController @RequestMapping("bipapi/groups") @AllArgsConstructor public class GroupController { public GroupService service;@GetMapping("/all") public ResponseEntity<BaseResponse> getAll() {return ResponseEntity.*ok*(new GroupListResponse("Список всех групп:", true, service.getAll()));}@GetMapping("/byId") public ResponseEntity<BaseResponse> getById(@RequestParam Long id) {return ResponseEntity.*ok*(new GroupResponse("Группа по id", true, service.getById(id)));}@PostMapping("/add") public ResponseEntity<BaseResponse> add(@RequestBody GroupEntity group) {try {GroupValidation.*validation*(group);if (service.getAll().contains(group)) throw new GroupExceptionValidation("Такая группа уже существует");service.save(group);return ResponseEntity.*ok*(new BaseResponse("Группа добавлена", true));} catch (GroupExceptionValidation exc) {return ResponseEntity.*badRequest*().body(new BaseResponse(exc.getMessage(), false));} catch (Exception e) {return ResponseEntity.*badRequest*().body(new BaseResponse(e.getMessage(), false));}}@PutMapping("/edit") public ResponseEntity<BaseResponse> edit(@RequestBody GroupEntity group) {try {GroupValidation.*validation*(group);service.save(group);return ResponseEntity.*ok*(new BaseResponse("Группа отредактирована", true));} catch (GroupExceptionValidation e) {return ResponseEntity.*badRequest*().body(new BaseResponse(e.getMessage(), false));} catch (Exception e) {return ResponseEntity.*badRequest*().body(new BaseResponse(e.getMessage(), false));}}@DeleteMapping("/delete{id}") public ResponseEntity<BaseResponse> delete(@PathVariable long id) {try {service.delete(id);return ResponseEntity.*ok*(new BaseResponse("Группа удалена", true));} catch (Exception e) {return ResponseEntity.*badRequest*().body(new BaseResponse(e.getMessage(), false));}}}

LoadController.java

package practice.server.practiceServer.controller;import lombok.AllArgsConstructor;import org.springframework.http.ResponseEntity;import org.springframework.web.bind.annotation.\*;import practice.server.practiceServer.entity.GroupEntity;import practice.server.practiceServer.entity.LoadEntity;import practice.server.practiceServer.entity.SubjectEntity;import practice.server.practiceServer.entity.TeacherEntity;import practice.server.practiceServer.exception.LoadExceptionValidation;import practice.server.practiceServer.model.LoadModel;import practice.server.practiceServer.response.BaseResponse;import practice.server.practiceServer.response.FinalLoadResponse;import practice.server.practiceServer.response.LoadListResponse;import practice.server.practiceServer.response.LoadResponse;import practice.server.practiceServer.service.GroupService;import practice.server.practiceServer.service.LoadService;import practice.server.practiceServer.service.SubjectService;import practice.server.practiceServer.service.TeacherService;import practice.server.practiceServer.utils.LoadValidation;import java.util.ArrayList;import java.util.List;@RestController @RequestMapping("bipapi/loads") @AllArgsConstructor public class LoadController { public LoadService service;public GroupService groupService;public TeacherService teacherService;public SubjectService subjectService;public List<LoadModel> toModel (List<LoadEntity> toModelList) {List<LoadModel> model = new ArrayList<>();for (int i = 0; i < toModelList.size(); i++) {LoadEntity toModel = toModelList.get(i);model.add(new LoadModel(toModel.getId(), toModel.getSemesterFirstHours(), toModel.getSemesterSecondHours(), toModel.getExamHours(), toModel.getCourseworkHours(), toModel.getDiplomaHours(), toModel.getConsultationHours(), toModel.getTeacherInLoad().getInitials(), toModel.getSubjectInLoad().getSubjectName(), toModel.getGroupInLoad().getGroupName()));}return model;}public LoadModel toModel (LoadEntity toModel) {return new LoadModel(toModel.getId(), toModel.getSemesterFirstHours(), toModel.getSemesterSecondHours(), toModel.getExamHours(), toModel.getCourseworkHours(), toModel.getDiplomaHours(), toModel.getConsultationHours(), toModel.getTeacherInLoad().getInitials(), toModel.getSubjectInLoad().getSubjectName(), toModel.getGroupInLoad().getGroupName());}public List<LoadEntity> fromModel (List<LoadModel> modelList) {List<LoadEntity> entity = new ArrayList<>();for (int i = 0; i < modelList.size(); i++) {LoadModel model = modelList.get(i);List<SubjectEntity> subjectsFuckYourself = subjectService.findByName(model.getSubjectInLoad());SubjectEntity subject = new SubjectEntity();for (int j = 0; j < subjectsFuckYourself.size(); j++) subject = subjectsFuckYourself.get(j).getSubjectName().equals(model.getSubjectInLoad())? subjectsFuckYourself.get(j) : null;entity.add(new LoadEntity(model.getSemesterFirstHours(), model.getSemesterSecondHours(), model.getExamHours(),model.getCourseworkHours(),model.getDiplomaHours(),model.getConsultationHours(), teacherService.findByName(model.getTeacherInLoad()), subject, groupService.findByName(model.getGroupInLoad())));}return entity;}public LoadEntity fromModel (LoadModel model) {List<SubjectEntity> subjectsFuckYourself = subjectService.findByName(model.getSubjectInLoad());SubjectEntity subject = new SubjectEntity();for (int j = 0; j < subjectsFuckYourself.size(); j++) subject = subjectsFuckYourself.get(j).getSubjectName().equals(model.getSubjectInLoad())? subjectsFuckYourself.get(j) : null;return new LoadEntity(model.getSemesterFirstHours(), model.getSemesterSecondHours(), model.getExamHours(), model.getCourseworkHours(), model.getDiplomaHours(), model.getConsultationHours(), teacherService.findByName(model.getTeacherInLoad()), subject, groupService.findByName(model.getGroupInLoad()));}@GetMapping("/all") public ResponseEntity<BaseResponse> getAll() {return ResponseEntity.*ok*(new LoadListResponse("Список всей нагрузки:", true, toModel(service.getAll())));}@GetMapping("/byId") public ResponseEntity<BaseResponse> getById(@RequestParam Long id) {return ResponseEntity.*ok*(new LoadResponse("Нагрузка по id", true, toModel(service.getById(id))));}@GetMapping("/finalLoadBy") public ResponseEntity<BaseResponse> getFinalLoadByTeacher(@RequestParam String groupname, @RequestParam String initials, @RequestParam String subjectname) {return ResponseEntity.*ok*(new FinalLoadResponse("Группа по id", true, service.finalLoadOfTeacher(service.getLoadBy(groupname, initials, subjectname))));}@PostMapping("/add") public ResponseEntity<BaseResponse> add(@RequestBody LoadModel loadModel) {try {LoadEntity load = fromModel(loadModel);LoadValidation.*validation*(load);if (service.getAll().contains(load)) throw new LoadExceptionValidation("Такая нагрузка уже существует");service.save(load);return ResponseEntity.*ok*(new BaseResponse("Нагрузка добавлена", true));} catch (LoadExceptionValidation exc) {return ResponseEntity.*badRequest*().body(new BaseResponse(exc.getMessage(), false));} catch (Exception e) {return ResponseEntity.*badRequest*().body(new BaseResponse(e.getMessage(), false));}}@PutMapping("/edit") public ResponseEntity<BaseResponse> edit(@RequestBody LoadModel loadModel) {try {LoadEntity load = fromModel(loadModel);LoadValidation.*validation*(load);service.save(load);return ResponseEntity.*ok*(new BaseResponse("Нагрузка отредактирована", true));} catch (LoadExceptionValidation e) {return ResponseEntity.*badRequest*().body(new BaseResponse(e.getMessage(), false));} catch (Exception e) {return ResponseEntity.*badRequest*().body(new BaseResponse(e.getMessage(), false));}}@DeleteMapping("/delete{id}") public ResponseEntity<BaseResponse> delete(@PathVariable long id) {try {service.delete(id);return ResponseEntity.*ok*(new BaseResponse("Нагрузка удалена", true));} catch (Exception e) {return ResponseEntity.*badRequest*().body(new BaseResponse(e.getMessage(), false));}}}

LogicController.java

package practice.server.practiceServer.controller;import lombok.AllArgsConstructor;import org.springframework.http.ResponseEntity;import org.springframework.web.bind.annotation.PostMapping;import org.springframework.web.bind.annotation.RequestBody;import org.springframework.web.bind.annotation.RequestMapping;import org.springframework.web.bind.annotation.RestController;import practice.server.practiceServer.entity.\*;import practice.server.practiceServer.response.BaseResponse;import practice.server.practiceServer.service.\*;import java.util.ArrayList;import java.util.HashSet;import java.util.List;import java.util.regex.Matcher;import java.util.regex.Pattern;@RestController @RequestMapping("bipapi/fromexcel") @AllArgsConstructor public class LogicController { public GroupService group;public LoadService load;public SubjectService subject;public TeacherService teacher;@PostMapping("/conversion") public ResponseEntity<BaseResponse> convertToDB(@RequestBody List<LoadEntity> sheets) {load.truncateMyTable();subject.deleteAll();group.deleteAll();teacher.deleteAll();int size = sheets.size();iterateHashToList(sheets);List<LoadEntity> loads = load.getAll();List<LoadEntity> finalLoads = new ArrayList<>();for (int i = 0; i < sheets.size(); i++) {LoadEntity loades = sheets.get(i);loades.setGroupInLoad(group.findByName(loades.getGroupInLoad().getGroupName()));loades.setTeacherInLoad(teacher.findByName(loades.getTeacherInLoad().getInitials()));List<SubjectEntity> subjectsFuckYourself = subject.findByName(loades.getSubjectInLoad().getSubjectName());SubjectEntity subject = new SubjectEntity();for (int j = 0; j < subjectsFuckYourself.size(); j++) subject = subjectsFuckYourself.get(j).getSubjectName().equals(loades.getSubjectInLoad().getSubjectName())? subjectsFuckYourself.get(j) : null;loades.setSubjectInLoad(subject);if (!loads.contains(loades)) {loads.add(loades);finalLoads.add(loades);}}load.saveAll(finalLoads);return ResponseEntity.*ok*(new BaseResponse(size == sheets.size()? "Из файла были добавлены все записи" : "Количество записей, не добавленных в бд:" + (sheets.size() - size), true));}public void iterateHashToList(List<LoadEntity> loads) {Pattern pattern = Pattern.*compile*("[0-9]\*[А-Я]+-[0-9]+[А-Я]\*");Pattern patternChars = Pattern.*compile*("[А-Я]+");Matcher matcherMain;Matcher matcherChars;HashSet<GroupEntity> groupsHash = new HashSet<>();HashSet<TeacherEntity> teachersHash = new HashSet<>();HashSet<SubjectEntity> subjectsHash = new HashSet<>();for (int i = 0; i < loads.size(); i++) {LoadEntity loadMaybe = loads.get(i);GroupEntity groupMaybe = loadMaybe.getGroupInLoad();matcherMain = pattern.matcher(groupMaybe.getGroupName());matcherChars = patternChars.matcher(groupMaybe.getGroupName());if (!matcherMain.find() && matcherChars.find()) groupMaybe.setGroupName(matcherChars.replaceFirst(loadMaybe.getGroupInLoad().getGroupName().substring(matcherChars.start(), matcherChars.end()).concat("-").trim()).trim());groupMaybe.setGroupName(groupMaybe.getGroupName().replace(" ", ""));groupsHash.add(groupMaybe);TeacherEntity teacherMaybe = loadMaybe.getTeacherInLoad();teacherMaybe.setInitials(loadMaybe.getTeacherInLoad().getInitials().trim());teachersHash.add(teacherMaybe);SubjectEntity subjectMaybe = loadMaybe.getSubjectInLoad();subjectMaybe.setSubjectName(subjectMaybe.getSubjectName().trim());subjectsHash.add(subjectMaybe);}List<GroupEntity> groups = new ArrayList<>(groupsHash.stream().toList());List<TeacherEntity> teachers = new ArrayList<>(teachersHash.stream().toList());List<SubjectEntity> subjects = new ArrayList<>(subjectsHash.stream().toList());group.saveAll(groups);teacher.saveAll(teachers);subject.saveAll(subjects);}}

SubjectController.java

package practice.server.practiceServer.controller;import lombok.AllArgsConstructor;import org.springframework.http.ResponseEntity;import org.springframework.web.bind.annotation.\*;import practice.server.practiceServer.entity.SubjectEntity;import practice.server.practiceServer.exception.SubjectExceptionValidation;import practice.server.practiceServer.response.BaseResponse;import practice.server.practiceServer.response.SubjectListResponse;import practice.server.practiceServer.response.SubjectResponse;import practice.server.practiceServer.service.SubjectService;import practice.server.practiceServer.utils.SubjectValidation;import java.util.List;@RestController @RequestMapping("bipapi/subjects") @AllArgsConstructor public class SubjectController { public SubjectService service;@GetMapping("/all") public ResponseEntity<BaseResponse> getAll() {return ResponseEntity.*ok*(new SubjectListResponse("Список всех предметов:", true, service.getAll()));}@GetMapping("/byId") public ResponseEntity<BaseResponse> getById(@RequestParam Long id) {return ResponseEntity.*ok*(new SubjectResponse("Предмет по id", true, service.getById(id)));}@PostMapping("/add") public ResponseEntity<BaseResponse> add(@RequestBody SubjectEntity subject) {try {SubjectValidation.*validation*(subject);service.save(subject);return ResponseEntity.*ok*(new BaseResponse("Предмет добавлена", true));} catch (SubjectExceptionValidation exc) {return ResponseEntity.*badRequest*().body(new BaseResponse(exc.getMessage(), false));} catch (Exception e) {return ResponseEntity.*badRequest*().body(new BaseResponse(e.getMessage(), false));}}@PutMapping("/edit") public ResponseEntity<BaseResponse> edit(@RequestBody SubjectEntity subject) {try {SubjectValidation.*validation*(subject);service.save(subject);return ResponseEntity.*ok*(new BaseResponse("Предмет отредактирован", true));} catch (SubjectExceptionValidation e) {return ResponseEntity.*badRequest*().body(new BaseResponse(e.getMessage(), false));} catch (Exception e) {return ResponseEntity.*badRequest*().body(new BaseResponse(e.getMessage(), false));}}@DeleteMapping("/delete{id}") public ResponseEntity<BaseResponse> delete(@PathVariable long id) {try {service.delete(id);return ResponseEntity.*ok*(new BaseResponse("Предмет удалён", true));} catch (Exception e) {return ResponseEntity.*badRequest*().body(new BaseResponse(e.getMessage(), false));}}}

TeacherController.java

package practice.server.practiceServer.controller;import lombok.AllArgsConstructor;import org.springframework.http.ResponseEntity;import org.springframework.web.bind.annotation.\*;import practice.server.practiceServer.entity.TeacherEntity;import practice.server.practiceServer.exception.TeacherExceptionValidation;import practice.server.practiceServer.response.BaseResponse;import practice.server.practiceServer.response.TeacherListResponse;import practice.server.practiceServer.response.TeacherResponse;import practice.server.practiceServer.service.TeacherService;import practice.server.practiceServer.utils.TeacherValidation;import java.util.List;@RestController @RequestMapping("bipapi/teachers") @AllArgsConstructor public class TeacherController { public TeacherService service;@GetMapping("/all") public ResponseEntity<BaseResponse> getAll() {return ResponseEntity.*ok*(new TeacherListResponse("Список всех учителей:", true, service.getAll()));}@GetMapping("/byId") public ResponseEntity<BaseResponse> getById(@RequestParam Long id) {return ResponseEntity.*ok*(new TeacherResponse("Учитель по id", true, service.getById(id)));}@PostMapping("/add") public ResponseEntity<BaseResponse> add(@RequestBody TeacherEntity teacher) {try {TeacherValidation.*validation*(teacher);service.save(teacher);return ResponseEntity.*ok*(new BaseResponse("Учитель добавлен", true));} catch (TeacherExceptionValidation exc) {return ResponseEntity.*badRequest*().body(new BaseResponse(exc.getMessage(), false));} catch (Exception e) {return ResponseEntity.*badRequest*().body(new BaseResponse(e.getMessage(), false));}}@PutMapping("/edit") public ResponseEntity<BaseResponse> edit(@RequestBody TeacherEntity teacher) {try {TeacherValidation.*validation*(teacher);service.save(teacher);return ResponseEntity.*ok*(new BaseResponse("Учитель отредактирован", true));} catch (TeacherExceptionValidation e) {return ResponseEntity.*badRequest*().body(new BaseResponse(e.getMessage(), false));} catch (Exception e) {return ResponseEntity.*badRequest*().body(new BaseResponse(e.getMessage(), false));}}@DeleteMapping("/delete{id}") public ResponseEntity<BaseResponse> delete(@PathVariable long id) {try {service.delete(id);return ResponseEntity.*ok*(new BaseResponse("Учитель удалён", true));} catch (Exception e) {return ResponseEntity.*badRequest*().body(new BaseResponse(e.getMessage(), false));}}}

GroupEntity.java

package practice.server.practiceServer.entity;import com.fasterxml.jackson.annotation.JsonIgnore;import lombok.AllArgsConstructor;import lombok.Data;import lombok.NoArgsConstructor;import lombok.RequiredArgsConstructor;import javax.persistence.\*;import javax.validation.constraints.NotBlank;import javax.validation.constraints.NotNull;import java.util.List;@Entity @Data @Table(name = "Groups") @AllArgsConstructor @NoArgsConstructor public class GroupEntity { @Id @GeneratedValue(strategy = GenerationType.*IDENTITY*) private Long id;@NotBlank private String groupName;@NotNull @JsonIgnore @OneToMany(cascade = CascadeType.*ALL*, mappedBy = "groupInLoad") private List<LoadEntity> groupInLoad;public GroupEntity(String groupName) {this.groupName = groupName;}public GroupEntity(Long id, String groupName) {this.id = id;this.groupName = groupName;}}

LoadEntity.java

package practice.server.practiceServer.entity;import lombok.AllArgsConstructor;import lombok.Data;import lombok.NoArgsConstructor;import javax.persistence.\*;import javax.validation.constraints.NotBlank;import javax.validation.constraints.NotNull;@Entity @Data @Table(name = "LoadOfTeachers") @AllArgsConstructor @NoArgsConstructor public class LoadEntity { @Id @GeneratedValue(strategy = GenerationType.*IDENTITY*) private Long id;@NotBlank private Integer semesterFirstHours;@NotBlank private Integer semesterSecondHours;@NotBlank private Integer examHours;@NotBlank private Integer courseworkHours;@NotBlank private Integer diplomaHours;@NotBlank private Integer consultationHours;@NotNull @ManyToOne private TeacherEntity teacherInLoad;@NotNull @ManyToOne private SubjectEntity subjectInLoad;@NotNull @ManyToOne private GroupEntity groupInLoad;public LoadEntity(Integer semesterFirstHours, Integer semesterSecondHours, Integer examHours, Integer courseworkHours, Integer diplomaHours, Integer consultationHours, TeacherEntity initials, SubjectEntity subjectName, GroupEntity groupName) {this.semesterFirstHours = semesterFirstHours;this.semesterSecondHours = semesterSecondHours;this.examHours = examHours;this.courseworkHours = courseworkHours;this.diplomaHours = diplomaHours;this.consultationHours = consultationHours;this.teacherInLoad = initials;this.subjectInLoad = subjectName;this.groupInLoad = groupName;}}

SubjectEntity.java

package practice.server.practiceServer.entity;import com.fasterxml.jackson.annotation.JsonIgnore;import lombok.AllArgsConstructor;import lombok.Data;import lombok.NoArgsConstructor;import javax.persistence.\*;import javax.validation.constraints.NotBlank;import javax.validation.constraints.NotNull;import java.util.List;@Entity @Data @Table(name = "Subject") @AllArgsConstructor @NoArgsConstructor public class SubjectEntity { @Id @GeneratedValue(strategy = GenerationType.*IDENTITY*) private Long id;@NotBlank private String subjectName;@NotNull @JsonIgnore @OneToMany(cascade = CascadeType.*ALL*, mappedBy = "subjectInLoad") private List<LoadEntity> subjectInLoad;public SubjectEntity(Long id, String subjectName) {this.id = id;this.subjectName = subjectName;}public SubjectEntity(String subjectName) {this.subjectName = subjectName;}}

TeacherEntity.java

package practice.server.practiceServer.entity;import com.fasterxml.jackson.annotation.JsonIgnore;import lombok.AllArgsConstructor;import lombok.Data;import lombok.NoArgsConstructor;import javax.persistence.\*;import javax.validation.constraints.NotBlank;import javax.validation.constraints.NotNull;import java.util.List;@Entity @Data @Table(name = "Teachers") @AllArgsConstructor @NoArgsConstructor public class TeacherEntity { @Id @GeneratedValue(strategy = GenerationType.*IDENTITY*) private Long id;@NotBlank private String initials;@NotNull @JsonIgnore @OneToMany(cascade = CascadeType.*ALL*, mappedBy = "teacherInLoad") private List<LoadEntity> teacherInLoad;public TeacherEntity(Long id, String initials) {this.id = id;this.initials = initials;}public TeacherEntity(String initials) {this.initials = initials;}}

GroupValidationException.java

package practice.server.practiceServer.exception;import javax.validation.ValidationException;public class GroupExceptionValidation extends ValidationException { public GroupExceptionValidation(String message) {super(message);}}

LoadValidationException.java

package practice.server.practiceServer.exception;import javax.validation.ValidationException;public class LoadExceptionValidation extends ValidationException { public LoadExceptionValidation(String message) {super(message);}}

SubjectValidationException.java

package practice.server.practiceServer.exception;import javax.validation.ValidationException;public class SubjectExceptionValidation extends ValidationException { public SubjectExceptionValidation(String message) {super(message);}}

TeacherValidationException.java

package practice.server.practiceServer.exception;import javax.validation.ValidationException;public class TeacherExceptionValidation extends ValidationException { public TeacherExceptionValidation(String message) {super(message);}}

LoadModel.java

package practice.server.practiceServer.model;import lombok.AllArgsConstructor;import lombok.Data;import lombok.NoArgsConstructor;@Data @AllArgsConstructor @NoArgsConstructor public class LoadModel { private Long id;private Integer semesterFirstHours;private Integer semesterSecondHours;private Integer examHours;private Integer courseworkHours;private Integer diplomaHours;private Integer consultationHours;private String teacherInLoad;private String subjectInLoad;private String groupInLoad;public LoadModel(Integer semesterFirstHours, Integer semesterSecondHours, Integer examHours, Integer courseworkHours, Integer diplomaHours, Integer consultationHours, String initials, String subjectName, String groupName) {this.semesterFirstHours = semesterFirstHours;this.semesterSecondHours = semesterSecondHours;this.examHours = examHours;this.courseworkHours = courseworkHours;this.diplomaHours = diplomaHours;this.consultationHours = consultationHours;this.teacherInLoad = initials;this.subjectInLoad = subjectName;this.groupInLoad = groupName;}}

GroupRepo.java

package practice.server.practiceServer.repo;import org.springframework.data.jpa.repository.Modifying;import org.springframework.data.jpa.repository.Query;import org.springframework.data.repository.CrudRepository;import org.springframework.transaction.annotation.Transactional;import practice.server.practiceServer.entity.GroupEntity;import java.util.List;import java.util.Optional;public interface GroupRepo extends CrudRepository<GroupEntity, Long> { @Override <S extends GroupEntity> List<S> saveAll(Iterable<S> entities);@Override List<GroupEntity> findAll();GroupEntity findByGroupName(String groupName);@Modifying @Query(value = "truncate table groups", nativeQuery = true) void truncateMyTable();}class GroupRepoReworked implements GroupRepo { @Override public <S extends GroupEntity> S save(S entity) {return null;}@Override public <S extends GroupEntity> List<S> saveAll(Iterable<S> entities) {return null;}@Override public Optional<GroupEntity> findById(Long aLong) {return Optional.*empty*();}@Override public boolean existsById(Long aLong) {return false;}@Override public List<GroupEntity> findAll() {return null;}@Override public GroupEntity findByGroupName(String groupname) {return null;}@Override public void truncateMyTable() {}@Override public Iterable<GroupEntity> findAllById(Iterable<Long> longs) {return null;}@Override public long count() {return 0;}@Override public void deleteById(Long aLong) {}@Override public void delete(GroupEntity entity) {}@Override public void deleteAllById(Iterable<? extends Long> longs) {}@Override public void deleteAll(Iterable<? extends GroupEntity> entities) {}@Override public void deleteAll() {}}

LoadRepo.java

package practice.server.practiceServer.repo;import org.springframework.data.jpa.repository.Modifying;import org.springframework.data.jpa.repository.Query;import org.springframework.data.repository.CrudRepository;import practice.server.practiceServer.entity.GroupEntity;import practice.server.practiceServer.entity.LoadEntity;import java.util.List;import java.util.Optional;public interface LoadRepo extends CrudRepository<LoadEntity, Long> { @Override <S extends LoadEntity> List<S> saveAll(Iterable<S> entities);@Override List<LoadEntity> findAll();List<LoadEntity> findByGroupInLoad\_GroupNameOrTeacherInLoad\_InitialsOrSubjectInLoad\_SubjectName(String groupname, String initials, String subjectname);@Modifying @Query(value = "truncate table load\_of\_teachers", nativeQuery = true) void truncateMyTable();}class LoadRepoReworked implements LoadRepo { @Override public <S extends LoadEntity> S save(S entity) {return null;}@Override public <S extends LoadEntity> List<S> saveAll(Iterable<S> entities) {return null;}@Override public Optional<LoadEntity> findById(Long aLong) {return Optional.*empty*();}@Override public boolean existsById(Long aLong) {return false;}@Override public List<LoadEntity> findAll() {return null;}@Override public List<LoadEntity> findByGroupInLoad\_GroupNameOrTeacherInLoad\_InitialsOrSubjectInLoad\_SubjectName(String groupname, String initials, String subjectname) {return null;}@Override public void truncateMyTable() {}@Override public Iterable<LoadEntity> findAllById(Iterable<Long> longs) {return null;}@Override public long count() {return 0;}@Override public void deleteById(Long aLong) {}@Override public void delete(LoadEntity entity) {}@Override public void deleteAllById(Iterable<? extends Long> longs) {}@Override public void deleteAll(Iterable<? extends LoadEntity> entities) {}@Override public void deleteAll() {}}

SubjectRepo.java

package practice.server.practiceServer.repo;import org.springframework.data.jpa.repository.Modifying;import org.springframework.data.jpa.repository.Query;import org.springframework.data.repository.CrudRepository;import practice.server.practiceServer.entity.SubjectEntity;import java.util.List;import java.util.Optional;public interface SubjectRepo extends CrudRepository<SubjectEntity, Long> { @Override <S extends SubjectEntity> List<S> saveAll(Iterable<S> entities);@Override List<SubjectEntity> findAll();List<SubjectEntity> findBySubjectName(String subjectName);@Modifying @Query(value = "truncate table subject", nativeQuery = true) void truncateMyTable();}class SubjectRepoReworkd implements SubjectRepo { @Override public <S extends SubjectEntity> S save(S entity) {return null;}@Override public <S extends SubjectEntity> List<S> saveAll(Iterable<S> entities) {return null;}@Override public Optional<SubjectEntity> findById(Long aLong) {return Optional.*empty*();}@Override public boolean existsById(Long aLong) {return false;}@Override public List<SubjectEntity> findAll() {return null;}@Override public List<SubjectEntity> findBySubjectName(String subjectName) {return null;}@Override public void truncateMyTable() {}@Override public Iterable<SubjectEntity> findAllById(Iterable<Long> longs) {return null;}@Override public long count() {return 0;}@Override public void deleteById(Long aLong) {}@Override public void delete(SubjectEntity entity) {}@Override public void deleteAllById(Iterable<? extends Long> longs) {}@Override public void deleteAll(Iterable<? extends SubjectEntity> entities) {}@Override public void deleteAll() {}}

TeacherRepo.java

package practice.server.practiceServer.repo;import org.springframework.data.jpa.repository.Modifying;import org.springframework.data.jpa.repository.Query;import org.springframework.data.repository.CrudRepository;import practice.server.practiceServer.entity.TeacherEntity;import java.util.List;import java.util.Optional;public interface TeacherRepo extends CrudRepository<TeacherEntity, Long> { @Override <S extends TeacherEntity> List<S> saveAll(Iterable<S> entities);@Override List<TeacherEntity> findAll();TeacherEntity findByInitials(String initials);@Modifying @Query(value = "truncate table teachers", nativeQuery = true) void truncateMyTable();}class TeacherRepoReworked implements TeacherRepo { @Override public <S extends TeacherEntity> S save(S entity) {return null;}@Override public <S extends TeacherEntity> List<S> saveAll(Iterable<S> entities) {return null;}@Override public Optional<TeacherEntity> findById(Long aLong) {return Optional.*empty*();}@Override public boolean existsById(Long aLong) {return false;}@Override public List<TeacherEntity> findAll() {return null;}@Override public TeacherEntity findByInitials(String initials) {return null;}@Override public void truncateMyTable() {}@Override public Iterable<TeacherEntity> findAllById(Iterable<Long> longs) {return null;}@Override public long count() {return 0;}@Override public void deleteById(Long aLong) {}@Override public void delete(TeacherEntity entity) {}@Override public void deleteAllById(Iterable<? extends Long> longs) {}@Override public void deleteAll(Iterable<? extends TeacherEntity> entities) {}@Override public void deleteAll() {}}

BaseResponse.java

package practice.server.practiceServer.response;import lombok.AllArgsConstructor;import lombok.Data;@Data @AllArgsConstructor public class BaseResponse { protected String message;protected boolean success;}

GroupListResponse.java

package practice.server.practiceServer.response;import lombok.Getter;import lombok.Setter;import practice.server.practiceServer.entity.GroupEntity;import java.util.List;@Getter @Setter public class GroupListResponse extends BaseResponse { private List<GroupEntity> data;public GroupListResponse(String message, boolean success, List<GroupEntity> data) {super(message, success);this.data = data;}}

GroupResponse.java

package practice.server.practiceServer.response;import lombok.Getter;import lombok.Setter;import practice.server.practiceServer.entity.GroupEntity;@Getter @Setter public class GroupResponse extends BaseResponse { private GroupEntity data;public GroupResponse(String message, boolean success, GroupEntity data) {super(message, success);this.data = data;}}

LoadListResponse.java

package practice.server.practiceServer.response;import lombok.Getter;import lombok.Setter;import practice.server.practiceServer.entity.LoadEntity;import practice.server.practiceServer.model.LoadModel;import java.util.List;@Getter @Setter public class LoadListResponse extends BaseResponse { private List<LoadModel> data;public LoadListResponse(String message, boolean success, List<LoadModel> data) {super(message, success);this.data = data;}}

LoadResponse.java

package practice.server.practiceServer.response;import lombok.Getter;import lombok.Setter;import practice.server.practiceServer.entity.LoadEntity;import practice.server.practiceServer.model.LoadModel;import java.util.List;@Getter @Setter public class LoadResponse extends BaseResponse { private LoadModel data;public LoadResponse(String message, boolean success, LoadModel data) {super(message, success);this.data = data;}}

SubjectListResponse.java

package practice.server.practiceServer.response;import lombok.Getter;import lombok.Setter;import practice.server.practiceServer.entity.SubjectEntity;import java.util.List;@Getter @Setter public class SubjectListResponse extends BaseResponse { private List<SubjectEntity> data;public SubjectListResponse(String message, boolean success, List<SubjectEntity> data) {super(message, success);this.data = data;}}

SubjectListResponse.java

package practice.server.practiceServer.response;import lombok.Getter;import lombok.Setter;import practice.server.practiceServer.entity.SubjectEntity;import java.util.List;@Getter @Setter public class SubjectResponse extends BaseResponse { private SubjectEntity data;public SubjectResponse(String message, boolean success, SubjectEntity data) {super(message, success);this.data = data;}}

TeacherListResponse.java

package practice.server.practiceServer.response;import lombok.Getter;import lombok.Setter;import practice.server.practiceServer.entity.TeacherEntity;import java.util.List;@Getter @Setter public class TeacherListResponse extends BaseResponse { private List<TeacherEntity> data;public TeacherListResponse(String message, boolean success, List<TeacherEntity> data) {super(message, success);this.data = data;}}

TeacherResponse.java

package practice.server.practiceServer.response;import lombok.Getter;import lombok.Setter;import practice.server.practiceServer.entity.TeacherEntity;import java.util.List;@Getter @Setter public class TeacherResponse extends BaseResponse { private TeacherEntity data;public TeacherResponse(String message, boolean success, TeacherEntity data) {super(message, success);this.data = data;}}

GroupService.java

package practice.server.practiceServer.service;import lombok.AllArgsConstructor;import lombok.Data;import org.springframework.stereotype.Service;import org.springframework.transaction.annotation.Transactional;import practice.server.practiceServer.entity.GroupEntity;import practice.server.practiceServer.repo.GroupRepo;import java.util.List;@Service @Data @AllArgsConstructor public class GroupService { public GroupRepo repo;public void saveAll(List<GroupEntity> groupList) {repo.saveAll(groupList);}public void save(GroupEntity group) {repo.save(group);}public void delete(Long id) {repo.deleteById(id);}public List<GroupEntity> getAll() {return repo.findAll();}public GroupEntity findByName(String name) {return repo.findByGroupName(name);}public GroupEntity getById(Long id) {return repo.findById(id).isPresent()? repo.findById(id).get() : null;}public void deleteAll() {repo.deleteAll();}}

LoadService.java

package practice.server.practiceServer.service;import lombok.AllArgsConstructor;import lombok.Data;import org.springframework.stereotype.Service;import org.springframework.transaction.annotation.Transactional;import practice.server.practiceServer.entity.LoadEntity;import practice.server.practiceServer.repo.LoadRepo;import java.util.ArrayList;import java.util.List;@Service @Data @AllArgsConstructor public class LoadService { public LoadRepo repo;public void saveAll(List<LoadEntity> groupList) {repo.saveAll(groupList);}public void save(LoadEntity group) {repo.save(group);}public void delete(Long id) {repo.deleteById(id);}public List<LoadEntity> getAll() {return repo.findAll();}public LoadEntity getById(Long id) {return repo.findById(id).isPresent() ? repo.findById(id).get() : null;}public List<LoadEntity> getLoadBy(String groupname, String initials, String subjectname) {return repo.findByGroupInLoad\_GroupNameOrTeacherInLoad\_InitialsOrSubjectInLoad\_SubjectName(groupname, initials, subjectname);}public List<Integer> finalLoadOfTeacher(List<LoadEntity> loads) {List<Integer> summs = new ArrayList<>();Integer semester1 = 0;Integer semester2 = 0;Integer exam = 0;Integer coursework = 0;Integer diploma = 0;Integer consultations = 0;for (int i = 0; i < loads.size(); i++) {semester1 += loads.get(i).getSemesterFirstHours();semester2 += loads.get(i).getSemesterSecondHours();exam += loads.get(i).getExamHours();coursework += loads.get(i).getCourseworkHours();diploma += loads.get(i).getDiplomaHours();consultations += loads.get(i).getConsultationHours();}summs.add(semester1);summs.add(semester2);summs.add(exam);summs.add(coursework);summs.add(diploma);summs.add(consultations);return summs;}@Transactional public void truncateMyTable() {repo.truncateMyTable();}}

SubjectService.java

package practice.server.practiceServer.service;import lombok.AllArgsConstructor;import lombok.Data;import org.springframework.stereotype.Service;import org.springframework.transaction.annotation.Transactional;import practice.server.practiceServer.entity.SubjectEntity;import practice.server.practiceServer.repo.SubjectRepo;import java.util.List;@Service @Data @AllArgsConstructor public class SubjectService { public SubjectRepo repo;public void saveAll(List<SubjectEntity> groupList) {repo.saveAll(groupList);}public void save(SubjectEntity group) {repo.save(group);}public void delete(Long id) {repo.deleteById(id);}public List<SubjectEntity> getAll() {return repo.findAll();}public List<SubjectEntity> findByName(String name) {return repo.findBySubjectName(name);}public SubjectEntity getById(Long id) {return repo.findById(id).isPresent() ? repo.findById(id).get() : null;}public void deleteAll() {repo.deleteAll();}}

TeacherService.java

package practice.server.practiceServer.service;import lombok.AllArgsConstructor;import lombok.Data;import org.springframework.stereotype.Service;import practice.server.practiceServer.entity.TeacherEntity;import practice.server.practiceServer.repo.TeacherRepo;import java.util.List;@Service @Data @AllArgsConstructor public class TeacherService { public TeacherRepo repo;public void saveAll(List<TeacherEntity> groupList) {repo.saveAll(groupList);}public void save(TeacherEntity group) {repo.save(group);}public void delete(Long id) {repo.deleteById(id);}public List<TeacherEntity> getAll() {return repo.findAll();}public TeacherEntity findByName(String name) {return repo.findByInitials(name);}public TeacherEntity getById(Long id) {return repo.findById(id).isPresent() ? repo.findById(id).get() : null;}public void deleteAll() {repo.deleteAll();}}

GroupValidation.java

package practice.server.practiceServer.utils;import practice.server.practiceServer.entity.GroupEntity;import practice.server.practiceServer.exception.GroupExceptionValidation;import java.util.regex.Pattern;public class GroupValidation { public static void validation(GroupEntity entity) {if (entity.getGroupName() == null || entity.getGroupName().isBlank()) throw new GroupExceptionValidation("Группа должна быть заполнена");Pattern pattern = Pattern.*compile*("[0-9]\*[А-Я]+-[0-9]+[А-Я]\*");if (!pattern.matcher(entity.getGroupName()).find()) throw new GroupExceptionValidation("Группа должна быть заполнена по образцу");}}

LoadValidation.java

package practice.server.practiceServer.utils;import practice.server.practiceServer.entity.LoadEntity;import practice.server.practiceServer.exception.LoadExceptionValidation;import java.util.regex.Pattern;public class LoadValidation { public static void validation(LoadEntity entity) {if (entity.getTeacherInLoad() == null) throw new LoadExceptionValidation("Учитель должен быть заполнен");if (entity.getGroupInLoad() == null) throw new LoadExceptionValidation("Группа должна быть заполнена");if (entity.getSubjectInLoad() == null) throw new LoadExceptionValidation("Предмет должен быть заполнен");Pattern pattern = Pattern.*compile*("^[0-9]+$");if (entity.getSemesterFirstHours() < 0) throw new LoadExceptionValidation("Данные за первый семестр должны быть заполнены");if (!pattern.matcher(entity.getSemesterFirstHours().toString()).find()) throw new LoadExceptionValidation("Данные за первый семестр должны быть цифрами");if (entity.getSemesterSecondHours() < 0) throw new LoadExceptionValidation("Данные за второй семестр должны быть заполнены");if (!pattern.matcher(entity.getSemesterSecondHours().toString()).find()) throw new LoadExceptionValidation("Данные за второй семестр должны быть цифрами");if (entity.getConsultationHours() < 0) throw new LoadExceptionValidation("Данные за консультацию должны быть заполнены");if (!pattern.matcher(entity.getConsultationHours().toString()).find()) throw new LoadExceptionValidation("Данные за консультацию должны быть цифрами");if (entity.getCourseworkHours() < 0) throw new LoadExceptionValidation("Данные за курсовые должны быть заполнены");if (!pattern.matcher(entity.getCourseworkHours().toString()).find()) throw new LoadExceptionValidation("Данные за курсовые должны быть цифрами");if (entity.getDiplomaHours() < 0) throw new LoadExceptionValidation("Данные за диплом должны быть заполнены");if (!pattern.matcher(entity.getDiplomaHours().toString()).find()) throw new LoadExceptionValidation("Данные за диплом должны быть цифрами");if (entity.getExamHours() < 0) throw new LoadExceptionValidation("Данные за экзамен должны быть заполнены");if (!pattern.matcher(entity.getExamHours().toString()).find()) throw new LoadExceptionValidation("Данные за экзамен должны быть цифрами");}}

SubjectValidation.java

package practice.server.practiceServer.utils;import practice.server.practiceServer.entity.SubjectEntity;import practice.server.practiceServer.exception.SubjectExceptionValidation;import java.util.regex.Pattern;public class SubjectValidation { public static void validation(SubjectEntity entity) {if (entity.getSubjectName() == null) throw new SubjectExceptionValidation("Предмет должен быть заполнен");}}

TeacherValidation.java

package practice.server.practiceServer.utils;import practice.server.practiceServer.entity.TeacherEntity;import practice.server.practiceServer.exception.TeacherExceptionValidation;import java.util.regex.Pattern;public class TeacherValidation { public static void validation(TeacherEntity entity) {if (entity.getInitials() == null) throw new TeacherExceptionValidation("Учитель должен быть заполнен");Pattern pattern = Pattern.*compile*("[А-Я][а-я]+\\s+[А-Я]\\.[А-Я]\\.");if (!pattern.matcher(entity.getInitials()).find()) throw new TeacherExceptionValidation("Учитель должен быть заполнен по шаблону");}}

PracticeServerApplication.java

package practice.server.practiceServer;import org.springframework.boot.SpringApplication;import org.springframework.boot.autoconfigure.SpringBootApplication;@SpringBootApplication public class PracticeServerApplication { public static void main(String[] args) {SpringApplication.*run*(PracticeServerApplication.class, args);}}

AddEditController.java

package client.bip.cleintdontdeletefuck.controller;import client.bip.cleintdontdeletefuck.entity.GroupEntity;import client.bip.cleintdontdeletefuck.entity.LoadEntity;import client.bip.cleintdontdeletefuck.entity.SubjectEntity;import client.bip.cleintdontdeletefuck.entity.TeacherEntity;import javafx.event.ActionEvent;import javafx.fxml.FXML;import javafx.scene.control.Button;import javafx.scene.control.ComboBox;import javafx.scene.control.Label;import javafx.scene.control.TextField;import javafx.scene.layout.AnchorPane;import javafx.scene.layout.GridPane;import javafx.stage.Stage;import java.io.IOException;import static client.bip.cleintdontdeletefuck.controller.MainController.\*;import static client.bip.cleintdontdeletefuck.ClientApplication.\*;public class AddEditController { @FXML private Button cancelButton;@FXML private Label changeTheLabel;@FXML private TextField consultationsInput;@FXML private TextField courseworkInput;@FXML private TextField diplomaInput;@FXML private TextField examInput;@FXML private TextField firstSemseterInput;@FXML private ComboBox<String> groupComboBox;@FXML private TextField changingInputForSmallTables;@FXML private AnchorPane loadAnchorPane;@FXML private TextField secondSemesterInput;@FXML private ComboBox<String> subjectComboBox;@FXML private ComboBox<String> teacherComboBox;@FXML private GridPane smallTablesGridPane;@FXML void initialize() {switch (*whatTableIsSelected*) {case "groups" -> {loadAnchorPane.setVisible(false);smallTablesGridPane.setVisible(true);if ("".equals(*selectedFromComboBoxSelectTable*)) break;changeTheLabel.setText("Введите название группы");changingInputForSmallTables.setText(*selectedFromComboBoxSelectTable*);}case "loads" -> {loadAnchorPane.setVisible(true);smallTablesGridPane.setVisible(false);if (*selectedFromLoadTable* == null) break;consultationsInput.setText(*selectedFromLoadTable*.getConsultationHours().toString());courseworkInput.setText(*selectedFromLoadTable*.getCourseworkHours().toString());examInput.setText(*selectedFromLoadTable*.getExamHours().toString());diplomaInput.setText(*selectedFromLoadTable*.getDiplomaHours().toString());firstSemseterInput.setText(*selectedFromLoadTable*.getSemesterFirstHours().toString());secondSemesterInput.setText(*selectedFromLoadTable*.getSemesterSecondHours().toString());teacherComboBox.setValue(*selectedFromLoadTable*.getTeacherInLoad());subjectComboBox.setValue(*selectedFromLoadTable*.getSubjectInLoad());groupComboBox.setValue(*selectedFromLoadTable*.getGroupInLoad());}case "subjects" -> {loadAnchorPane.setVisible(false);smallTablesGridPane.setVisible(true);if ("".equals(*selectedFromComboBoxSelectTable*)) break;changeTheLabel.setText("Введите название предмета");changingInputForSmallTables.setText(*selectedFromComboBoxSelectTable*);}case "teachers" -> {loadAnchorPane.setVisible(false);smallTablesGridPane.setVisible(true);if ("".equals(*selectedFromComboBoxSelectTable*)) break;changeTheLabel.setText("Введите инициалы учителя");changingInputForSmallTables.setText(*selectedFromComboBoxSelectTable*);}}}@FXML void cancel(ActionEvent event) {Stage stage = (Stage) cancelButton.getScene().getWindow();stage.close();}@FXML void save(ActionEvent event) {try {switch (*whatTableIsSelected*) {case "groups" -> {GroupEntity group = new GroupEntity();group.setId(*forSelectedComboBoxTable*);group.setGroupName(changingInputForSmallTables.getText());if (!"".equals(*selectedFromComboBoxSelectTable*)) {System.*out*.println(*http*.put(*api*.concat("groups/edit"), *gson*.toJson(group, GroupEntity.class)));*infoMessage*("Запись успешно обновлена!");return;}*http*.post(*api*.concat("groups/add"), *gson*.toJson(group, GroupEntity.class));*infoMessage*("Запись успешно добавлена!");*selectedFromComboBoxSelectTable* = group.getGroupName();} case "loads" -> {LoadEntity load = new LoadEntity(Integer.*parseInt*(firstSemseterInput.getText()), Integer.*parseInt*(secondSemesterInput.getText()), Integer.*parseInt*(examInput.getText()), Integer.*parseInt*(courseworkInput.getText()), Integer.*parseInt*(diplomaInput.getText()), Integer.*parseInt*(consultationsInput.getText()), teacherComboBox.getSelectionModel().getSelectedItem(), subjectComboBox.getSelectionModel().getSelectedItem(), groupComboBox.getSelectionModel().getSelectedItem());if (*selectedFromLoadTable* != null) {*http*.put(*api*.concat("loads/edit"), *gson*.toJson(load, LoadEntity.class));*infoMessage*("Запись успешно обновлена!");return;}*http*.post(*api*.concat("loads/add"), *gson*.toJson(load, LoadEntity.class));*infoMessage*("Запись успешно добавлена!");*selectedFromLoadTable* = load;}case "subjects" -> {SubjectEntity subject = *forSelectedComboBoxTable* == -1? new SubjectEntity(changingInputForSmallTables.getText()) : new SubjectEntity(*forSelectedComboBoxTable*, changingInputForSmallTables.getText());if (!"".equals(*selectedFromComboBoxSelectTable*)) {*http*.put(*api*.concat("subjects/edit"), *gson*.toJson(subject, SubjectEntity.class));*infoMessage*("Запись успешно обновлена!");return;}*http*.post(*api*.concat("subjects/add"), *gson*.toJson(subject, SubjectEntity.class));*infoMessage*("Запись успешно добавлена!");*selectedFromComboBoxSelectTable* = subject.getSubjectName();}case "teachers" -> {TeacherEntity teachers = *forSelectedComboBoxTable* == -1? new TeacherEntity(changingInputForSmallTables.getText()) : new TeacherEntity(*forSelectedComboBoxTable*, changingInputForSmallTables.getText());if (!"".equals(*selectedFromComboBoxSelectTable*)) {*http*.put(*api*.concat("teachers/edit"), *gson*.toJson(teachers, TeacherEntity.class));*infoMessage*("Запись успешно обновлена!");return;}*http*.post(*api*.concat("teachers/add"), *gson*.toJson(teachers, TeacherEntity.class));*infoMessage*("Запись успешно добавлена!");*selectedFromComboBoxSelectTable* = teachers.getInitials();}}} catch (IOException e) {*errorMessage*("Сервер выключен. Пожалуйста, попробуйте позже!");}}}

LogicController.java

package client.bip.cleintdontdeletefuck.controller;import client.bip.cleintdontdeletefuck.entity.\*;import org.apache.poi.ss.usermodel.\*;import org.apache.poi.xssf.usermodel.XSSFSheet;import org.apache.poi.xssf.usermodel.XSSFWorkbook;import java.io.\*;import java.util.\*;import static client.bip.cleintdontdeletefuck.controller.MainController.\*;public class LogicController { public static List<LoadEntity> *workload* = new ArrayList<>();public static void getValues(String filePath) {try {FileInputStream fileIN = new FileInputStream(filePath);XSSFWorkbook workbook = new XSSFWorkbook(fileIN);XSSFSheet sheet = workbook.getSheetAt(0);List<Row> rowIterator = new ArrayList<>();for (Row row : sheet) rowIterator.add(row);for (int i = 5; i < rowIterator.size(); i++) {Row thisRow = rowIterator.get(i);if (thisRow.getCell(0) == null || thisRow.getCell(0).getStringCellValue().isEmpty() || thisRow.getCell(0).getStringCellValue().isBlank())continue;System.*out*.println(thisRow.getRowNum());/\* Инициалы учителя\*/String teacher = thisRow.getCell(0).getStringCellValue().trim();/\* Предмет\*/String subject = thisRow.getCell(1).getStringCellValue().trim().replace("ё", "е");/\* Группа\*/String group = thisRow.getCell(2).getStringCellValue().trim();/\* Экзамен\*/int exam = 0;/\* Диплом\*/int diplom\_hours = 0;/\* Консультации\*/int consultations = 0;/\* Курсовые\*/int coursework = 0;/\* Семестры\*/int term = thisRow.getCell(4).getNumericCellValue() == 1 ? 1 : 2;/\* Первый\*/int semesterFirst = 0;/\* Второй\*/int semesterSecond = 0;if (thisRow.getCell(5).getStringCellValue().equalsIgnoreCase("э")) {exam = (int) thisRow.getCell(6).getNumericCellValue();consultations = (int) thisRow.getCell(8).getNumericCellValue();}coursework = (int) thisRow.getCell(9).getNumericCellValue();if (term == 1) semesterFirst = (int) thisRow.getCell(10).getNumericCellValue(); else semesterSecond = (int) thisRow.getCell(11).getNumericCellValue();diplom\_hours = (int) thisRow.getCell(7).getNumericCellValue();*workload*.add(new LoadEntity(semesterFirst, semesterSecond, exam, coursework, diplom\_hours, consultations, teacher, subject, group));}*semestersConnect*();fileIN.close();} catch (Exception e) {e.printStackTrace();}}public static void semestersConnect() {*workload*.sort(Comparator.*comparing*(LoadEntity::getTeacherInLoad));for (int i = 0; i < *workload*.size(); i++) {LoadEntity load = *workload*.get(i);if (i + 1 >= *workload*.size()) return;LoadEntity nextLoad = *workload*.get(i + 1);if (!nextLoad.getTeacherInLoad().equals(load.getTeacherInLoad()))continue;if (!nextLoad.getSubjectInLoad().equals(load.getSubjectInLoad())) continue;if (load.getTeacherInLoad().isBlank() || load.getTeacherInLoad().isEmpty() || load.getTeacherInLoad() == null) {*workload*.remove(load);continue;}load.setConsultationHours(load.getConsultationHours() + nextLoad.getConsultationHours());load.setExamHours(load.getExamHours() + nextLoad.getExamHours());load.setDiplomaHours(load.getDiplomaHours() + nextLoad.getDiplomaHours());load.setCourseworkHours(load.getCourseworkHours() + nextLoad.getCourseworkHours());load.setSemesterFirstHours(load.getSemesterFirstHours() + nextLoad.getSemesterFirstHours());load.setSemesterSecondHours(load.getSemesterSecondHours() + nextLoad.getSemesterSecondHours());*workload*.remove(nextLoad);}}public static void setValues(String filePath, String sheetName, String whatIsNeed) {try {FileInputStream fileIN = new FileInputStream(filePath);XSSFWorkbook wb = new XSSFWorkbook(fileIN);fileIN.close();OutputStream fileOUT = new FileOutputStream(filePath);Sheet sheet = wb.getSheet(sheetName);List<String> header = new ArrayList<>();header.add("№ п/п");header.add("ФИО");header.add("Читаемые дисциплины");header.add("Диплом");header.add("Консультация");header.add("Экзамен");header.add("Курсовая");header.add("1 сем");header.add("2 сем");header.add("ВСЕГО");sheet.autoSizeColumn(0, true);sheet.setColumnWidth(1, 8 \* 1000);sheet.setColumnWidth(2, 15 \* 1000);sheet.autoSizeColumn(3, true);sheet.autoSizeColumn(4, true);sheet.autoSizeColumn(5, true);sheet.autoSizeColumn(6, true);sheet.autoSizeColumn(7, true);sheet.autoSizeColumn(8, true);sheet.autoSizeColumn(9, true);/\* "№ п/п", "ФИО", "Читаемые дисциплины", "Диплом", "Консультация", "Экзамен", "Курсовая", "1 сем", "2 сем", "Итого по дисциплинам", "ВСЕГО"\*/Row row = sheet.createRow(0);for (int i = 0; i < header.size(); i++) {Cell cell = row.createCell(i);cell.setCellValue(header.get(i));}List<String> cache = new ArrayList<>();Integer numberOfTeacher = 0;int cellNum = 1; /\* номер ячейки с начала таблицы (без учёта шапки)\*/for (int i = 0; i < *workload*.size(); i++) {LoadEntity load = *workload*.get(i);if (!"".equals(*whatTeacherNeedToCount*) && !(*whatTeacherNeedToCount* == null)) load.setTeacherInLoad(whatTeacherNeedToCount);if (cache.contains(load.getTeacherInLoad())) continue;numberOfTeacher++;List<LoadEntity> sameTeacher = load.listOfAllSubjectsThisTeacher(*workload*);List<Integer> allNumbers = load.semesterHours(sameTeacher);Integer totalHours = load.totalHours(allNumbers);Row rowTable = sheet.createRow(cellNum);CellStyle myCellStyle = wb.createCellStyle();BorderStyle bs\_medium = BorderStyle.*THIN*;myCellStyle.setBorderTop(bs\_medium);Cell rowNumber = rowTable.createCell(0);rowNumber.setCellValue(numberOfTeacher);rowNumber.setCellStyle(myCellStyle);Cell initials = rowTable.createCell(1);initials.setCellValue(load.getTeacherInLoad());initials.setCellStyle(myCellStyle);cache.add(load.getTeacherInLoad());Cell subjects = rowTable.createCell(2);subjects.setCellValue(load.getSubjectInLoad());subjects.setCellStyle(myCellStyle);if ('1' == whatIsNeed.charAt(0)) {Cell diploma = rowTable.createCell(3);diploma.setCellValue(allNumbers.get(1));diploma.setCellStyle(myCellStyle);} else totalHours -= allNumbers.get(1);if ('1' == whatIsNeed.charAt(1)){Cell consultations = rowTable.createCell(4);consultations.setCellValue(allNumbers.get(4));consultations.setCellStyle(myCellStyle);} else totalHours -= allNumbers.get(4);if ('1' == whatIsNeed.charAt(2)) {Cell exam = rowTable.createCell(5);exam.setCellValue(allNumbers.get(5));exam.setCellStyle(myCellStyle);} else totalHours -= allNumbers.get(5);if ('1' == whatIsNeed.charAt(3)){Cell coursework = rowTable.createCell(6);coursework.setCellValue(allNumbers.get(3));coursework.setCellStyle(myCellStyle);} else totalHours -= allNumbers.get(3);if ('1' == whatIsNeed.charAt(4)) {Cell semesterFirst = rowTable.createCell(7);semesterFirst.setCellValue(load.getSemesterFirstHours());semesterFirst.setCellStyle(myCellStyle);} else totalHours -= allNumbers.get(1);if ('1' == whatIsNeed.charAt(5)) {Cell semesterSecond = rowTable.createCell(8);semesterSecond.setCellValue(load.getSemesterSecondHours());semesterSecond.setCellStyle(myCellStyle);} else totalHours -= allNumbers.get(2);Cell total = rowTable.createCell(9);total.setCellValue(totalHours);total.setCellStyle(myCellStyle);if (sameTeacher.size() <= 1) {cellNum++;continue;}for (int d = 1; d < sameTeacher.size(); d++) {Row rowDist = sheet.createRow(cellNum + 1);rowDist.createCell(2).setCellValue(sameTeacher.get(d).getSubjectInLoad());rowDist.createCell(7).setCellValue(sameTeacher.get(d).getSemesterFirstHours());rowDist.createCell(8).setCellValue(sameTeacher.get(d).getSemesterSecondHours());cellNum++;}cellNum++;}wb.write(fileOUT);fileOUT.close();} catch (Exception e) {e.printStackTrace();}}public static List<FinalLoadEntity> convertToServerLoadEntity() {List<FinalLoadEntity> server = new ArrayList<>();for (LoadEntity toServer: *workload*) server.add(new FinalLoadEntity(new TeacherEntity(toServer.getTeacherInLoad()), new SubjectEntity(toServer.getSubjectInLoad()), new GroupEntity(toServer.getGroupInLoad()), toServer.getExamHours(), toServer.getDiplomaHours(), toServer.getConsultationHours(), toServer.getCourseworkHours(), toServer.getSemesterFirstHours(), toServer.getSemesterSecondHours()));return server;}public static void createSheet(String filePath, String sheetName) {try {File file = new File(filePath);XSSFWorkbook wb = new XSSFWorkbook();if (file.exists()) {FileInputStream fileIN = new FileInputStream(filePath);wb = new XSSFWorkbook(fileIN);fileIN.close();}FileOutputStream fileOut = new FileOutputStream(filePath);if (wb.getSheet(sheetName) == null) wb.createSheet(sheetName);wb.write(fileOut);fileOut.close();} catch (Exception err) {System.*out*.println("ERROR: " + err.getMessage());}}}

MainController.java

package client.bip.cleintdontdeletefuck.controller;import client.bip.cleintdontdeletefuck.ClientApplication;import client.bip.cleintdontdeletefuck.entity.GroupEntity;import client.bip.cleintdontdeletefuck.entity.LoadEntity;import client.bip.cleintdontdeletefuck.entity.SubjectEntity;import client.bip.cleintdontdeletefuck.entity.TeacherEntity;import client.bip.cleintdontdeletefuck.utils.HTTPUtils;import com.google.gson.Gson;import com.google.gson.JsonArray;import com.google.gson.JsonObject;import javafx.collections.FXCollections;import javafx.collections.ObservableList;import javafx.event.ActionEvent;import javafx.fxml.FXML;import javafx.scene.control.CheckBox;import javafx.scene.control.ComboBox;import javafx.scene.control.TableColumn;import javafx.scene.control.TableView;import javafx.scene.control.cell.PropertyValueFactory;import javafx.scene.layout.AnchorPane;import javafx.scene.layout.GridPane;import java.io.File;import java.io.IOException;import java.util.Comparator;import static client.bip.cleintdontdeletefuck.ClientApplication.\*;import static client.bip.cleintdontdeletefuck.controller.LogicController.\*;public class MainController { public static String *fileDirectory* = "D:\\bipApi\\cleintDONTDELETEFUCK\\src\\workload.xlsx";public static String *api* = "http://localhost:28242/bipapi/";public static Gson *gson* = new Gson();public static HTTPUtils *http* = new HTTPUtils();public static String *whatNeedToSum* = "111111";public static ObservableList<String> *toComboBox* = FXCollections.*observableArrayList*();public static ObservableList<SubjectEntity> *subjectComboBox* = FXCollections.*observableArrayList*();public static ObservableList<TeacherEntity> *teachersComboBox* = FXCollections.*observableArrayList*();public static ObservableList<GroupEntity> *groupsComboBox* = FXCollections.*observableArrayList*();public static ObservableList<LoadEntity> *loadEntityTableList* = FXCollections.*observableArrayList*();public static String *whatTableIsSelected* = "";public static String *whatTeacherNeedToCount* = "";public static String *whatGroupNeedToCount* = "";public static String *whatSubjectNeedToCount* = "";public static String *selectedFromComboBoxSelectTable* = "";public static LoadEntity *selectedFromLoadTable*;public static long *forSelectedComboBoxTable* = -1;@FXML private ComboBox<String> SubjectComboBox;@FXML private ComboBox<String> comboBoxSelectTable;@FXML private TableColumn<LoadEntity, Integer> consultationsColumn;@FXML private TableColumn<LoadEntity, Integer> courseworkColumn;@FXML private TableColumn<LoadEntity, Integer> diplomaColumn;@FXML private TableColumn<LoadEntity, Integer> examColumn;@FXML private AnchorPane finalLoadProperties;@FXML private TableColumn<LoadEntity, String> groupColumn;@FXML private ComboBox<String> groupComboBox;@FXML private TableView<LoadEntity> loadTable;@FXML private TableColumn<LoadEntity, Integer> semesterFirstColumn;@FXML private TableColumn<LoadEntity, Integer> semesterSecondColumn;@FXML private GridPane smallTablesInside;@FXML private TableColumn<LoadEntity, String> subjectColumn;@FXML private GridPane tablesGridPane;@FXML private TableColumn<LoadEntity, String> teacherColumn;@FXML private ComboBox<String> teacherComboBox;@FXML private CheckBox withoutConsultations;@FXML private CheckBox withoutCoursework;@FXML private CheckBox withoutDiploma;@FXML private CheckBox withoutExam;@FXML private CheckBox withoutFirstSemester;@FXML private CheckBox withoutSecondSemester;@FXML void addButton(ActionEvent event){comboBoxSelectTable.getSelectionModel().clearSelection();loadTable.getSelectionModel().clearSelection();*showPersonEditDialog*("addEdit.fxml", "Добавление записи");if ("groups".equals(*whatTableIsSelected*) || "subjects".equals(*whatTableIsSelected*) || "teachers".equals(*whatTableIsSelected*)) {initializeComboboxSelectTable();return;}initializeLoadTable();}@FXML void changeFinalEdit(ActionEvent event) {*whatNeedToSum* = "";if (withoutDiploma.isSelected()) *whatNeedToSum* = *whatNeedToSum*.concat("1"); else *whatNeedToSum* = *whatNeedToSum*.concat("0");if (withoutConsultations.isSelected()) *whatNeedToSum* = *whatNeedToSum*.concat("1"); else *whatNeedToSum* = *whatNeedToSum*.concat("0");if (withoutExam.isSelected()) *whatNeedToSum* = *whatNeedToSum*.concat("1"); else *whatNeedToSum* = *whatNeedToSum*.concat("0");if (withoutCoursework.isSelected()) *whatNeedToSum* = *whatNeedToSum*.concat("1"); else *whatNeedToSum* = *whatNeedToSum*.concat("0");if (withoutFirstSemester.isSelected()) *whatNeedToSum* = *whatNeedToSum*.concat("1"); else *whatNeedToSum* = *whatNeedToSum*.concat("0");if (withoutSecondSemester.isSelected()) *whatNeedToSum* = *whatNeedToSum*.concat("1"); else *whatNeedToSum* = *whatNeedToSum*.concat("0");}@FXML void deleteButton(ActionEvent event) {try {switch (*whatTableIsSelected*) {case "groups" -> {if (comboBoxSelectTable.getSelectionModel().isEmpty()) {*warningMessage*("Для удаления нужно выбрать запись!");return;}for (GroupEntity groups : *groupsComboBox*) if (groups.getGroupName().equals(comboBoxSelectTable.getSelectionModel().getSelectedItem())) {*http*.delete(*api*, "groups/delete".concat(groups.getId().toString()));*groupsComboBox*.clear();initializeComboboxSelectTable();return;}}case "loads" -> {if (loadTable.getSelectionModel().isEmpty()) {*warningMessage*("Для удаления нужно выбрать запись!");return;}*http*.delete(*api*, "loads/delete".concat(loadTable.getSelectionModel().getSelectedItem().getId().toString()));*loadEntityTableList*.clear();initializeLoadTable();}case "subjects" -> {if (comboBoxSelectTable.getSelectionModel().isEmpty()) {*warningMessage*("Для удаления нужно выбрать запись!");return;}for (SubjectEntity subject : *subjectComboBox*) if (subject.getSubjectName().equals(comboBoxSelectTable.getSelectionModel().getSelectedItem())) {*http*.delete(*api*, "subjects/delete".concat(subject.getId().toString()));*subjectComboBox*.clear();initializeComboboxSelectTable();return;}}case "teachers" -> {if (comboBoxSelectTable.getSelectionModel().isEmpty()) {*warningMessage*("Для удаления нужно выбрать запись!");return;}for (TeacherEntity teacher : *teachersComboBox*) if (teacher.getInitials().equals(comboBoxSelectTable.getSelectionModel().getSelectedItem())) {*http*.delete(*api*, "teachers/delete".concat(teacher.getId().toString()));*teachersComboBox*.clear();initializeComboboxSelectTable();return;}}}} catch (IOException e) {*errorMessage*("Сервер выключен. Пожалуйста, попробуйте позже!");}}@FXML void editButton(ActionEvent event) {if (comboBoxSelectTable.getSelectionModel().isEmpty() && loadTable.getSelectionModel().isEmpty()) {*warningMessage*("Для редактирования нужно выбрать запись!");return;}switch (*whatTableIsSelected*) {case "groups" -> {*selectedFromComboBoxSelectTable* = comboBoxSelectTable.getSelectionModel().getSelectedItem();for (GroupEntity group : *groupsComboBox*) if (*selectedFromComboBoxSelectTable*.equals(group.getGroupName())) *forSelectedComboBoxTable* = group.getId();}case "loads" -> *selectedFromLoadTable* = loadTable.getSelectionModel().getSelectedItem();case "subjects" -> {*selectedFromComboBoxSelectTable* = comboBoxSelectTable.getSelectionModel().getSelectedItem();for (SubjectEntity subject : *subjectComboBox*) if (*selectedFromComboBoxSelectTable*.equals(subject.getSubjectName())) *forSelectedComboBoxTable* = subject.getId();}case "teachers" -> {*selectedFromComboBoxSelectTable* = comboBoxSelectTable.getSelectionModel().getSelectedItem();for (TeacherEntity teacher : *teachersComboBox*) if (*selectedFromComboBoxSelectTable*.equals(teacher.getInitials())) *forSelectedComboBoxTable* = teacher.getId();}}*showPersonEditDialog*("addEdit.fxml", "Редактирование записи");if (*whatTableIsSelected*.equals("groups") || *whatTableIsSelected*.equals("subjects") || *whatTableIsSelected*.equals("teachers")) {comboBoxSelectTable.getSelectionModel().clearSelection();*selectedFromComboBoxSelectTable* = "";initializeComboboxSelectTable();} else {loadTable.getSelectionModel().clearSelection();*selectedFromLoadTable* = null;initializeLoadTable();}}@FXML void finalLoadOpenButton(ActionEvent event){if (finalLoadProperties.isVisible()) {finalLoadProperties.setVisible(false);return;}initializeFinalLoadProperies();finalLoadProperties.setVisible(true);}private void initializeFinalLoadProperies() {*toComboBox*.clear();loadTable.setVisible(false);try {JsonObject baseTeachers = *gson*.fromJson(*http*.get(*api*, "teachers/all"), JsonObject.class);JsonArray dataTeacher = baseTeachers.getAsJsonArray("data");for (int i = 0; i < dataTeacher.size(); i++) {TeacherEntity teacher = *gson*.fromJson(dataTeacher.get(i).toString(), TeacherEntity.class);*toComboBox*.add(teacher.getInitials());}} catch (IOException e) {*errorMessage*("Сервер выключен. Пожалуйста, попробуйте позже!");}*toComboBox*.sort(Comparator.*comparing*(String::toString));teacherComboBox.getItems().setAll(*toComboBox*);}@FXML void workWithExcel(ActionEvent event) {String fileDirectoryFromUser = ClientApplication.*dialogMessage*(*fileDirectory*);if (!new File(fileDirectoryFromUser).exists()) {ClientApplication.*errorMessage*("Для считывания с файла он должен существовать!");return;}*getValues*(fileDirectoryFromUser);try {*http*.post(*api*.concat("fromexcel/conversion"), *gson*.toJson(*convertToServerLoadEntity*()));boolean answer = *applyMessage*("Хотите использовать заданные параметры для подсчёта нагрузки?");if (answer) {*createSheet*(fileDirectoryFromUser, "workloadForBip");*whatTeacherNeedToCount* = teacherComboBox.getSelectionModel().getSelectedItem();*setValues*(fileDirectoryFromUser, "workloadForBip", *whatNeedToSum*);} else {*createSheet*(fileDirectoryFromUser, "workloadForBip");*whatTeacherNeedToCount* = "";*setValues*(fileDirectoryFromUser, "workloadForBip", "111111");}initializeFinalLoadProperies();} catch (IOException e) {*errorMessage*("Сервер выключен. Пожалуйста, попробуйте позже!");}}@FXML void groupsOpenButton(ActionEvent event) {if (smallTablesInside.isVisible() && *whatTableIsSelected*.equals("groups")) {smallTablesInside.setVisible(false);comboBoxSelectTable.setVisible(false);return;}*whatTableIsSelected* = "groups";comboBoxSelectTable.setVisible(true);initializeComboboxSelectTable();}private void initializeComboboxSelectTable() {try {switch (*whatTableIsSelected*) {case "groups" -> {*toComboBox*.clear();comboBoxSelectTable.getEditor().clear();smallTablesInside.setVisible(true);JsonObject base = *gson*.fromJson(*http*.get(*api*, "groups/all"), JsonObject.class);JsonArray dataArr = base.getAsJsonArray("data");for (int i = 0; i < dataArr.size(); i++) {GroupEntity subject = *gson*.fromJson(dataArr.get(i).toString(), GroupEntity.class);*groupsComboBox*.add(subject);*toComboBox*.add(subject.getGroupName());}*toComboBox*.sort(Comparator.*comparing*(String::toString));comboBoxSelectTable.getItems().setAll(*toComboBox*);*toComboBox*.clear();}case "teachers" -> {JsonObject base = *gson*.fromJson(*http*.get(*api*, "teachers/all"), JsonObject.class);JsonArray dataArr = base.getAsJsonArray("data");for (int i = 0; i < dataArr.size(); i++) {TeacherEntity teacher = *gson*.fromJson(dataArr.get(i).toString(), TeacherEntity.class);*toComboBox*.add(teacher.getInitials());*teachersComboBox*.add(teacher);}*toComboBox*.sort(Comparator.*comparing*(String::toString));comboBoxSelectTable.getItems().setAll(*toComboBox*);*toComboBox*.clear();}case "subjects" -> {JsonObject base = *gson*.fromJson(*http*.get(*api*, "subjects/all"), JsonObject.class);JsonArray dataArr = base.getAsJsonArray("data");for (int i = 0; i < dataArr.size(); i++) {SubjectEntity subject = *gson*.fromJson(dataArr.get(i).toString(), SubjectEntity.class);*toComboBox*.add(subject.getSubjectName());*subjectComboBox*.add(subject);}*toComboBox*.sort(Comparator.*comparing*(String::toString));comboBoxSelectTable.getItems().setAll(*toComboBox*);*toComboBox*.clear();}}} catch (IOException e) {*errorMessage*("Сервер выключен. Пожалуйста, попробуйте позже!");}}@FXML void loadOpenButton(ActionEvent event) {*whatTableIsSelected* = "loads";*loadEntityTableList*.clear();if (loadTable.isVisible()) {loadTable.setVisible(false);smallTablesInside.setVisible(false);return;}smallTablesInside.setVisible(true);loadTable.setVisible(true);initializeLoadTable();teacherColumn.setCellValueFactory(new PropertyValueFactory<>("teacherInLoad"));subjectColumn.setCellValueFactory(new PropertyValueFactory<>("subjectInLoad"));groupColumn.setCellValueFactory(new PropertyValueFactory<>("groupInLoad"));semesterFirstColumn.setCellValueFactory(new PropertyValueFactory<>("semesterFirstHours"));semesterSecondColumn.setCellValueFactory(new PropertyValueFactory<>("semesterSecondHours"));examColumn.setCellValueFactory(new PropertyValueFactory<>("examHours"));courseworkColumn.setCellValueFactory(new PropertyValueFactory<>("courseworkHours"));diplomaColumn.setCellValueFactory(new PropertyValueFactory<>("diplomaHours"));consultationsColumn.setCellValueFactory(new PropertyValueFactory<>("consultationHours"));loadTable.getItems().setAll(*loadEntityTableList*);*loadEntityTableList*.clear();}private void initializeLoadTable() {try {JsonObject base = *gson*.fromJson(*http*.get(*api*, "loads/all"), JsonObject.class);JsonArray dataArr = base.getAsJsonArray("data");for (int i = 0; i < dataArr.size(); i++) {LoadEntity subject = *gson*.fromJson(dataArr.get(i).toString(), LoadEntity.class);*loadEntityTableList*.add(subject);}} catch (IOException e) {*errorMessage*("Сервер выключен. Пожалуйста, попробуйте позже!");}}@FXML void subjectsOpenButton(ActionEvent event) {if (smallTablesInside.isVisible() && *whatTableIsSelected*.equals("subjects")) {smallTablesInside.setVisible(false);comboBoxSelectTable.setVisible(false);return;}*whatTableIsSelected* = "subjects";*toComboBox*.clear();comboBoxSelectTable.getEditor().clear();smallTablesInside.setVisible(true);comboBoxSelectTable.setVisible(true);initializeComboboxSelectTable();}@FXML void tablesOpenButton(ActionEvent event) {if (tablesGridPane.isVisible()) tablesGridPane.setVisible(false); else tablesGridPane.setVisible(true);}@FXML void teachersOpernButton(ActionEvent event) {if (smallTablesInside.isVisible() && *whatTableIsSelected*.equals("teachers")) {smallTablesInside.setVisible(false);comboBoxSelectTable.setVisible(false);return;}*whatTableIsSelected* = "teachers";*toComboBox*.clear();comboBoxSelectTable.getEditor().clear();smallTablesInside.setVisible(true);comboBoxSelectTable.setVisible(true);initializeComboboxSelectTable();}}

FinalLoadEntity.java

package client.bip.cleintdontdeletefuck.entity;import lombok.Data;import lombok.NoArgsConstructor;@Data @NoArgsConstructor public class FinalLoadEntity { private TeacherEntity teacherInLoad;private SubjectEntity subjectInLoad;private GroupEntity groupInLoad;private Integer examHours;private Integer diplomaHours;private Integer consultationHours;private Integer courseworkHours;private Integer semesterFirstHours;private Integer semesterSecondHours;public FinalLoadEntity(TeacherEntity teacherInLoad, SubjectEntity subject, GroupEntity group, Integer exam, Integer diploma, Integer consultation, Integer coursework, Integer semesterFirst, Integer semesterSecond) {this.subjectInLoad = subject;this.groupInLoad = group;this.semesterFirstHours = semesterFirst;this.semesterSecondHours = semesterSecond;this.teacherInLoad = teacherInLoad;this.examHours = exam;this.diplomaHours = diploma;this.consultationHours = consultation;this.courseworkHours = coursework;}}

GroupEntity.java

package client.bip.cleintdontdeletefuck.entity;import lombok.Data;import lombok.NoArgsConstructor;@Data @NoArgsConstructor public class GroupEntity { private Long id;private String groupName;public GroupEntity(String groupName) {this.groupName = groupName;}public GroupEntity(Long id, String groupName) {this.id = id;this.groupName = groupName;}}

LoadEntity.java

package client.bip.cleintdontdeletefuck.entity;import lombok.AllArgsConstructor;import lombok.Data;import lombok.NoArgsConstructor;import java.util.ArrayList;import java.util.List;@Data @AllArgsConstructor @NoArgsConstructor public class LoadEntity { private Long id;private Integer semesterFirstHours;private Integer semesterSecondHours;private Integer examHours;private Integer courseworkHours;private Integer diplomaHours;private Integer consultationHours;private String teacherInLoad;private String subjectInLoad;private String groupInLoad;public LoadEntity(Integer semesterFirstHours, Integer semesterSecondHours, Integer examHours, Integer courseworkHours, Integer diplomaHours, Integer consultationHours, String initials, String subjectName, String groupName) {this.semesterFirstHours = semesterFirstHours;this.semesterSecondHours = semesterSecondHours;this.examHours = examHours;this.courseworkHours = courseworkHours;this.diplomaHours = diplomaHours;this.consultationHours = consultationHours;this.teacherInLoad = initials;this.subjectInLoad = subjectName;this.groupInLoad = groupName;}public List<LoadEntity> listOfAllSubjectsThisTeacher(List<LoadEntity> workload) {List<LoadEntity> loads = new ArrayList<>();for (LoadEntity load : workload) if (this.teacherInLoad.equals(load.teacherInLoad)) loads.add(load);return loads;}public Integer totalHours(List<Integer> semesters) {return semesters.get(0) + semesters.get(1) + semesters.get(2) + semesters.get(3) + semesters.get(4) + semesters.get(5);}public List<Integer> semesterHours(List<LoadEntity> loads) {List<Integer> allNumbers = new ArrayList<>();allNumbers.add(0);allNumbers.add(0);allNumbers.add(0);allNumbers.add(0);allNumbers.add(0);allNumbers.add(0);for (LoadEntity load: loads) {allNumbers.set(0, allNumbers.get(0) + load.semesterFirstHours);allNumbers.set(1, allNumbers.get(1) + load.diplomaHours);allNumbers.set(2, allNumbers.get(2) + load.semesterSecondHours);allNumbers.set(3, allNumbers.get(3) + load.courseworkHours);allNumbers.set(4, allNumbers.get(4) + load.consultationHours);allNumbers.set(5, allNumbers.get(5) + load.examHours);}return allNumbers;}}

SubjectEntity.java

package client.bip.cleintdontdeletefuck.entity;import lombok.Data;import lombok.NoArgsConstructor;@Data @NoArgsConstructor public class SubjectEntity { private Long id;private String subjectName;public SubjectEntity(Long id, String subjectName) {this.id = id;this.subjectName = subjectName;}public SubjectEntity(String subjectName) {this.subjectName = subjectName;}}

TeacherEntity.java

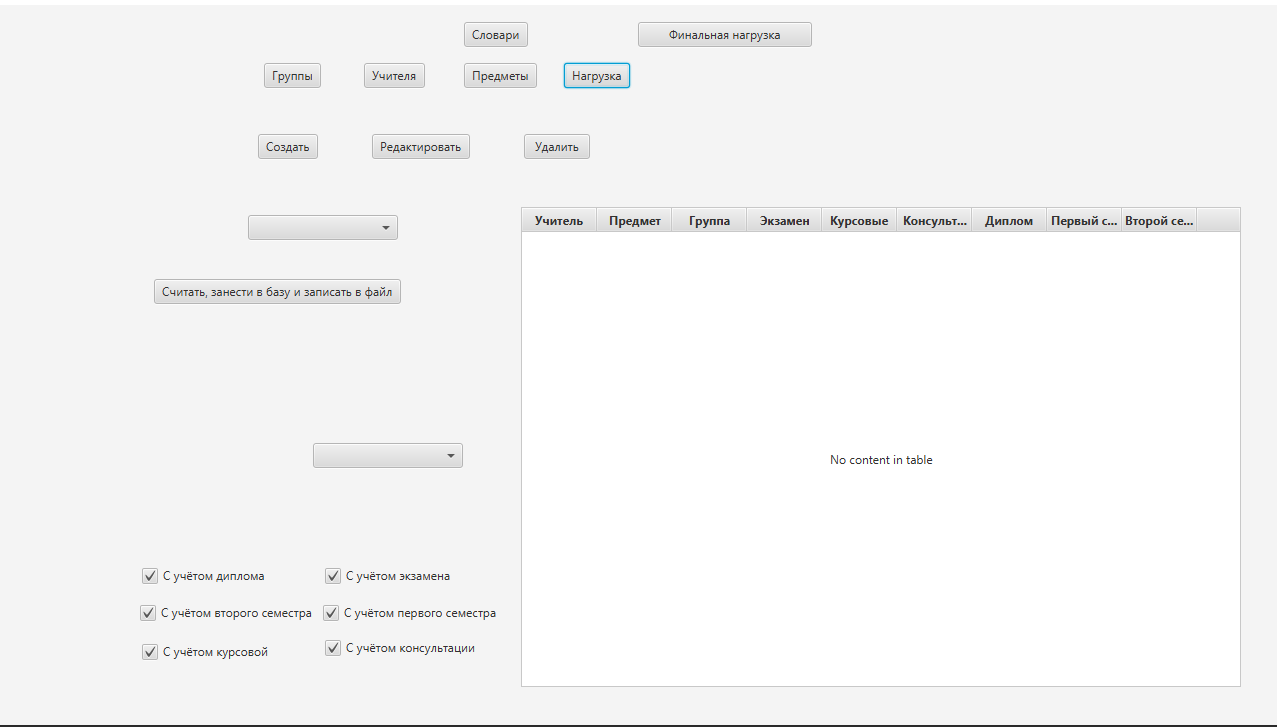
package client.bip.cleintdontdeletefuck.entity;import lombok.Data;import lombok.NoArgsConstructor;@Data @NoArgsConstructor public class TeacherEntity { private Long id;private String initials;public TeacherEntity(Long id, String initials) {this.id = id;this.initials = initials;}public TeacherEntity(String initials) {this.initials = initials;}}

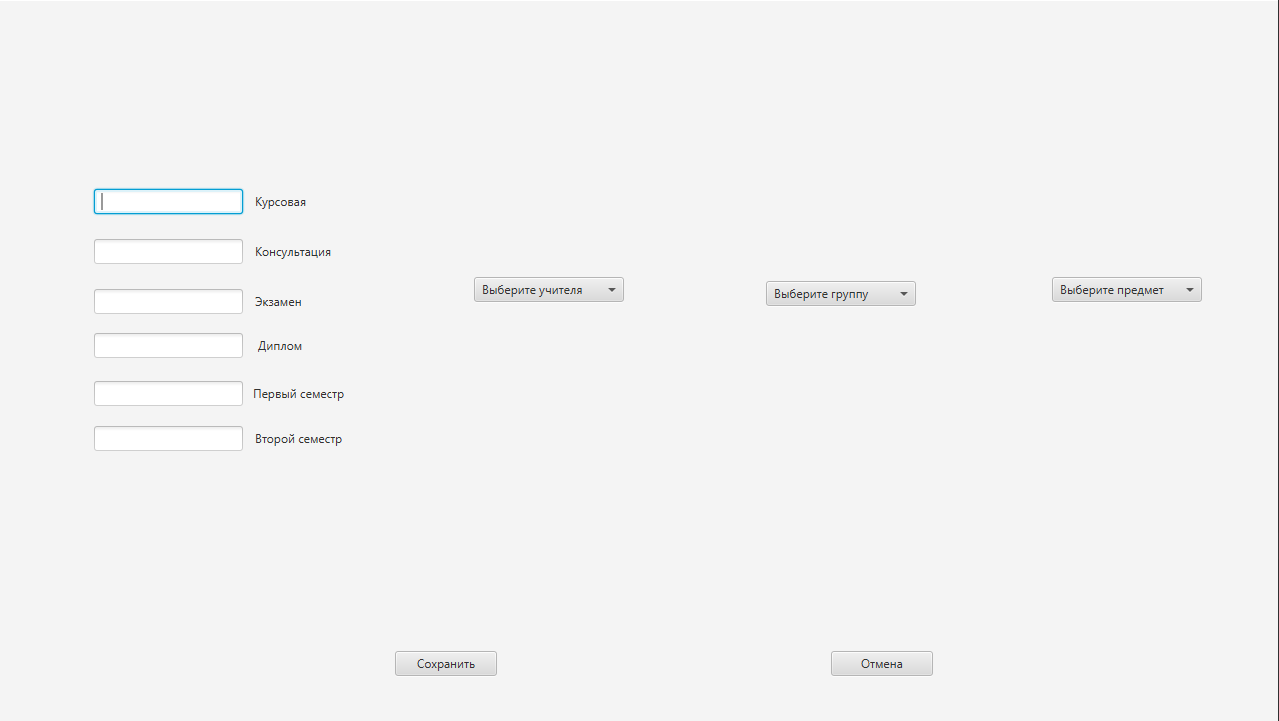
HTTPUtils.java

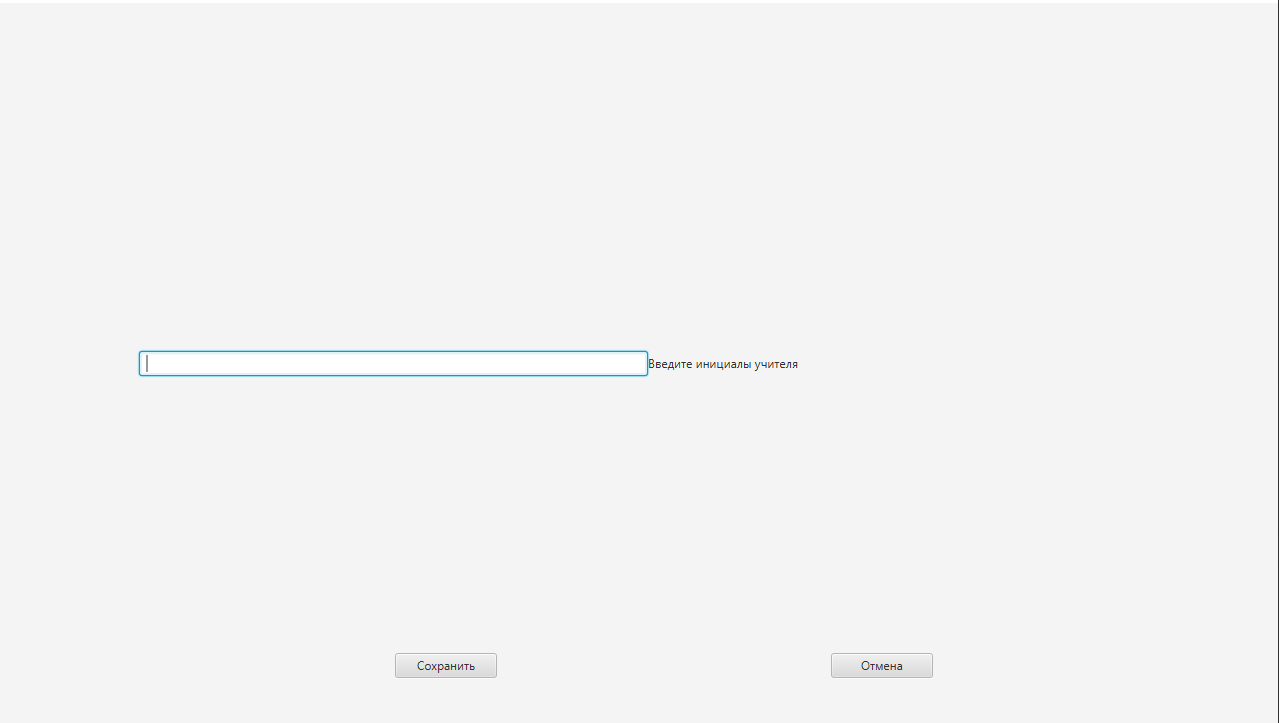
package client.bip.cleintdontdeletefuck.utils;import okhttp3.\*;import java.io.IOException;public class HTTPUtils { OkHttpClient client = new OkHttpClient();public String post(String url, String json) throws IOException {RequestBody body = RequestBody.*create*(json, MediaType.*get*("application/json; charset=utf-8"));Request request = new Request.Builder().url(url).post(body).build();try (Response response = client.newCall(request).execute()) {return response.body().string();}}public String get(String url, String args) throws IOException {Request req = new Request.Builder().url(url + args).build();try (Response response = client.newCall(req).execute()) {return response.body().string();}}public String put(String url, String json) throws IOException {RequestBody body = RequestBody.*create*(json, MediaType.*get*("application/json; charset=utf-8"));Request request = new Request.Builder().url(url).put(body).build();try (Response response = client.newCall(request).execute()) {return response.body().string();}}public String delete(String url, String args) throws IOException {Request req = new Request.Builder().url(url + args).delete().build();try (Response response = client.newCall(req).execute()) {return response.body().string();}}}

ClienApplication.java

package client.bip.cleintdontdeletefuck;import javafx.application.Application;import javafx.fxml.FXMLLoader;import javafx.scene.Scene;import javafx.scene.control.Alert;import javafx.scene.control.Button;import javafx.scene.control.ButtonType;import javafx.scene.control.TextInputDialog;import javafx.stage.Modality;import javafx.stage.Stage;import java.io.IOException;public class ClientApplication extends Application { public static void showPersonEditDialog(String type, String title) {try {FXMLLoader loader = new FXMLLoader(ClientApplication.class.getResource(type));Stage dialogStage = new Stage();dialogStage.setTitle(title);if ("main.fxml".equals(type)) dialogStage.initModality(Modality.*NONE*); else dialogStage.initModality(Modality.*APPLICATION\_MODAL*);dialogStage.setScene(new Scene(loader.load()));if ("main.fxml".equals(type)) dialogStage.show(); else dialogStage.showAndWait();} catch (IOException e) {e.printStackTrace();}}public static void infoMessage(String message) {Alert info = new Alert(Alert.AlertType.*INFORMATION*, message, ButtonType.*OK*);info.initModality(Modality.*WINDOW\_MODAL*);info.show();}public static boolean applyMessage(String message) {Alert info = new Alert(Alert.AlertType.*INFORMATION*, message, ButtonType.*YES*, ButtonType.*NO*);info.initModality(Modality.*WINDOW\_MODAL*);info.showAndWait();return info.getResult().equals(ButtonType.*YES*);}public static void warningMessage(String message) {Alert alert = new Alert(Alert.AlertType.*WARNING*, message, ButtonType.*OK*);alert.initModality(Modality.*WINDOW\_MODAL*);alert.show();}public static void errorMessage(String message) {Alert error = new Alert(Alert.AlertType.*ERROR*, message, ButtonType.*OK*);error.initModality(Modality.*WINDOW\_MODAL*);error.show();}public static String dialogMessage(String defaultValue) {TextInputDialog dialog = new TextInputDialog(defaultValue);dialog.initModality(Modality.*APPLICATION\_MODAL*);dialog.showAndWait();return dialog.getEditor().getText();}@Override public void start(Stage stage) throws IOException {FXMLLoader fxmlLoader = new FXMLLoader(ClientApplication.class.getResource("main.fxml"));Scene scene = new Scene(fxmlLoader.load());stage.setTitle("Hello!");stage.setScene(scene);stage.show();}public static void main(String[] args) {*launch*();}}

Главное окно

Форма создания или редактирования записи в таблице LoadOfTeachers



Форма добавления или изменения записи в таблицах Teachers, Groups, Subjects