|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ \_\_\_\_\_\_\_**ИНФОРМАТИКА И СИСТЕМ УПРАВЛЕНИЯ**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА \_\_\_\_\_\_\_\_**КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.03 Прикладная информатика**

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К КУРСОВОЙ РАБОТЕ***

***НА ТЕМУ:***

***\_\_\_\_\_\_\_\_\_ПРОГРАММА ТЕСТ-ТЕСТЫ\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_***

Студент \_\_ИУ6-И55Б\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** Ле Куанг Ньуе

(Группа) (Подпись, дата) (И.О.Фамилия) (И.О.Фамилия)

Руководитель курсовой работы **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  Cкворцова М. А.

(Подпись, дата) (И.О. Фамилия)

*2022 г.*



**РЕФЕРАТ**

РПЗ 28 с., 25 рис., 7 табл., 5 источников, 3 приложения.

Ключевые слова: MVC.

Объектом разработки является клиент-серверное веб - приложение, конечной целью которого является предоставление пользователям (преподавателю и студентам) возможности работы с экзаменом через онлайн-систему.

Цель работы – разработка веб-приложения, обеспечивающего:

* возможность прохождения теста за определенной время для студента;
* рабочее место преподавателя.

В результате разработки было спроектировано и реализовано веб-приложение.

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 5](#_Toc119837886)

[Глава 1. Анализ задания и выбор технологии, языка и среды разработки 6](#_Toc119837887)

[Глава 2. Проектирование и реализация программного продукта 8](#_Toc119837888)

[**1.** **Выбор методов решения задания** 8](#_Toc119837889)

[**2.** **Разработка диаграммы вариантов использования** 9](#_Toc119837890)

[**3.** **Разработка диаграммы классов программы** 12](#_Toc119837891)

[**4.** **Процесс обработки информации и анализ структуры данных** 13](#_Toc119837892)

[**5.** **Разработка интерфейса пользователей** 16](#_Toc119837893)

[**6.** **Описание форм интерфейса** 18](#_Toc119837894)

[Глава 3. Тестирование программного продукта 24](#_Toc119837895)

[**1.** **Анализ причинно-следственных связей** 24](#_Toc119837896)

[**2.** **Эквивалентное разбиение** 28](#_Toc119837897)

[**3.** **Анализ граничных условий** 30](#_Toc119837898)

[ЗАКЛЮЧЕНИЕ 32](#_Toc119837899)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 33](#_Toc119837900)

[ПРИЛОЖЕНИЯ А Техническое задание 34](#_Toc119837901)

[ПРИЛОЖЕНИЯ Б Руководство пользовотелей 42](#_Toc119837925)

[ПРИЛОЖЕНИЯ В Листинг кода 55](#_Toc119837926)

# **ВВЕДЕНИЕ**

Настоящее техническое задание распространяется на разработку программы «тест-тесты», используемой для смостоятельного изучения русской лексики и грамматики, предназначенной для русского преподавателя и иностранных студентов.

Русский язык - один из самых сложных языков в мире. Повторение и практика - лучший способ улучшить русский язык. Поэтому вышеуказанная программа родилась, чтобы помочь российским преподавателям и иностранным студентам иметь наиболее удобный способ обучения и тестирования.

Работа заключается в создании веб-приложения для обработки рабочих мест преподавателя и студента в экзамене. Каждому пользователю будут назначены функции, соответствующие его положению в экзамене с множественными выборами. Функции каждого пользователя будут представлены в разделе 4.1.1 ТЗ.

Разработанный сайт будет иметь удобный интерфейс и, больше всего, легкое использование.

# **Глава 1.** Анализ задания и выбор технологии, языка и среды разработки

Технологией программирования называют совокупность методов и средств, используемых в процессе разработки программного обеспечения.

Для написания любой программы необходимо выбирать язык программирования и среду разработки. Для разработки программы принято решение использовать объектно-ориентированный подход (объектный подход). Использование объектного подхода существенно повышает уровень унификации разработки и пригодность для повторного использования, что ведет к созданию среды разработки и переходу к сборочному созданию моделей. Объектный подход фокусируется на обоих аспектах системы (данных и действиях), так и он представляет собой способ мышления, который сопоставляет компоненты с реальными объектами. При таком подходе система разделяется соответственно на небольшие части, называемые объектами. Каждый объект включает в себя все данные и действия, связанные с этим объектом. Объекты в системе относительно независимы друг от друга, и программное обеспечение создается путем объединения этих объектов посредством их отношений и взаимодействий. Этот подход используется по следующим основным причинам:

- Ближе к реальному миру;

- Бозможность повторного использования;

- Инкапсуляция и сокрытие информации, следовательно, более надёжная система.

Cистема основана на языке программирования JAVA (для серверной части) и jQuery (для интерфейсной части) и объединена с помощью баз данных MySQL.

Для Back End использован язык программирования JAVA с применением Spring Framework. Spring Framework (коротко Spring) — универсальный фреймворк с открытым исходным кодом для Java-платформы. Spring обеспечивает комплексную модель разработки и конфигурации для современных приложений на Java. С помощью Spring Framework разработчики могут создавать высокопроизводительный, простой для тестирования и многократный используемый код.

Для интерфейса использованы HTML, CSS и Javascript c библиотекой Bootstrap. Bootstrap – библиотека шаблонов CSS, распространяемая свободно. Bootstrap - это просто встроенная библиотека, поэтому ее использование поможет разработчикам быстро разработать веб-сайт.

Для баз данных использована MySQL – система управления реляционными базами данных с открытым исходным кодом, поддерживаемая Oracle. Используется MySQL по следующим плюсам:

- Сервер MySQL очень гибкий. Кроме того, поскольку MySQL является системой управления базами данных с открытым исходным кодом, она позволяет настраивать соответствующие требования к серверу базы данных;

- MySQL — это высокоскоростная, стабильная и удобная база данных;

- Эта СУБД подходит для приложений, имеющих доступ к базам данных в Интернете, которым требуется высокий уровень безопасности;

- MySQl поддерживает множество функций SQL, мощных и расширяемых для обработки большего количества данных по мере необходимости;

- MySQL работает в нескольких операционных системах и на нескольких языках, включая PHP, PERL, JAVA, C, C ++, С#...

# **Глава 2. Проектирование и реализация программного продукта**

1. **Выбор методов решения задания**

Создание программы в направлении веб-API, разделение на back end и front end, что помогает уменьшить взаимозависимость двух частей:

- База данных: Использование реляционной базы данных SQL (MySQL);

- Back end: использована Spring Boot для написания API, использован JPA для сопоставления класса (объектно-ориентированный подход) в коде с таблицей (реляционный подход) SQL;

- Front end: интерфейс cоздано с помощью HTML, CSS. Взаимодействие с Backend API с помощью библиотеки jQuery (в частности, AJAX).

Model – View – Controller (MVC) - это шаблон проектирования, целью которого является разделение на интерфейс и код для упрощения управления, разработки и обслуживания. MVC делит программное приложение на 3 взаимодействующие части:

- Модель - это класс, содержащий информацию об объекте (данные), который взаимодействует с Базой Данных. Отвечает за моделирование объектов;

- Представление - это интерфейс системы, непосредственно взаимодействующий с пользователем;

- Контроллер получает запрос от пользователя. Он использует модель и представление для обработки и возврата результатов пользователю.

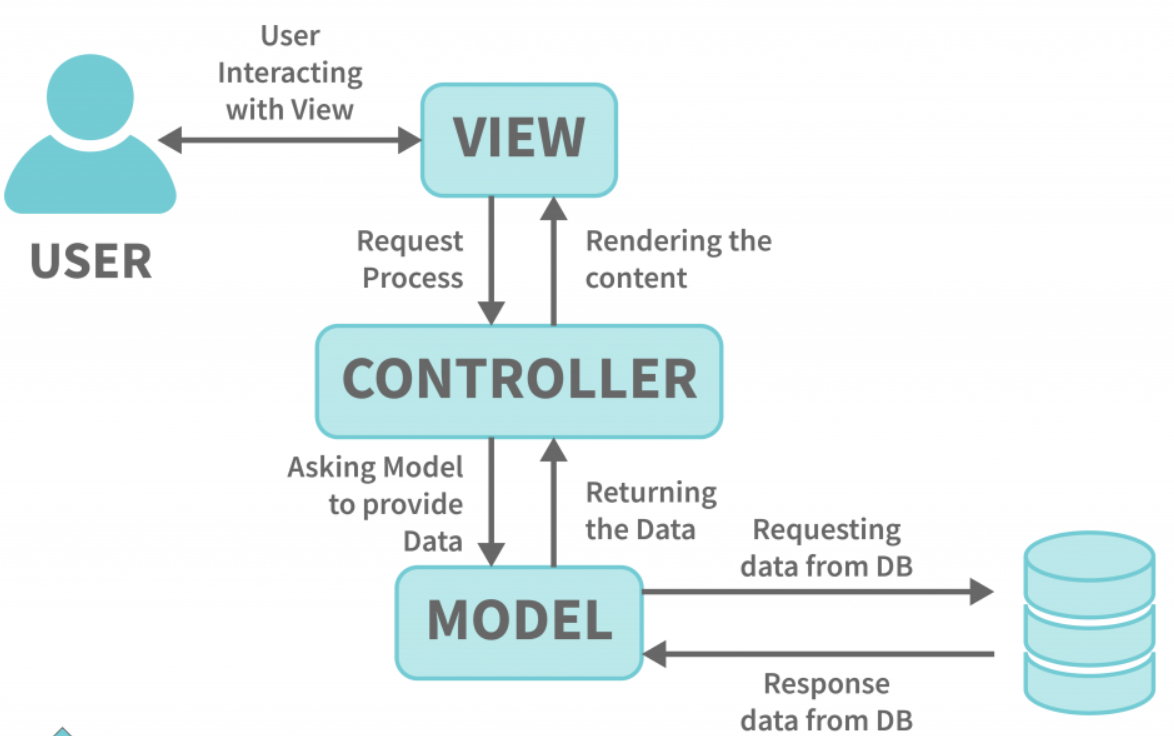


Рисунок 1 - Работа MVC

При этом, представление заключается в front end, а модель и контроллер заключаюся в back end.

Spring Boot – инструмент фреймворка Spring для написания приложений на Spring с минимальной конфигурацией. Данный инструмент также имеет встроенный контейнер сервлетов (Tomcat по умолчанию) что значительно упрощает запуск приложения. Веб приложения на Spring Boot запакованы в jar файл, что позволяет запускать их как обычные java приложения.

1. **Разработка диаграммы вариантов использования**

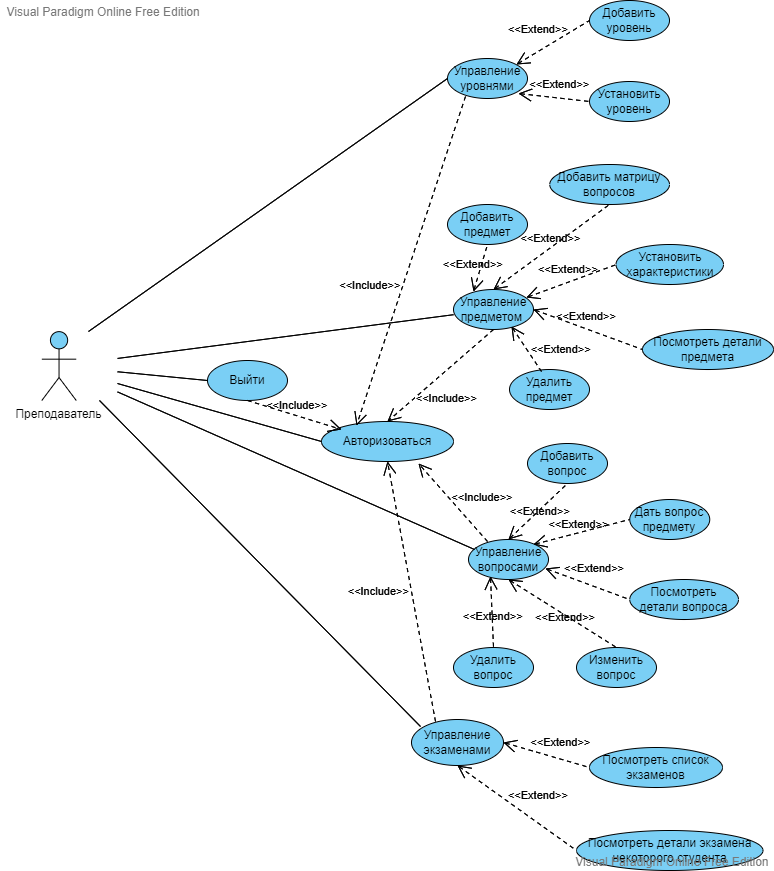
Разработку спецификаций программного обеспечения начинают с анализа требований к функциональности, указанных в техническом задании. В процессе анализа выявляют внешних пользователей разрабатываемого программного обеспечения и перечень отдельных аспектов его поведения в процессе взаимодействия с конкретными пользователями. Аспекты поведения программного обеспечения были названы «вариантами использования» или «прецедентами» (use cases).

Согласно техническому заданию, это веб-приложение используется многими пользователями (преподаватель и иностранные студенты). Выполняемые функции каждого пользователя представлены в таблице 1.

Таблица 1 - Выполняемые функции каждого пользователя

|  |  |
| --- | --- |
| Для преподавателя | Для иностранных студентов |
| * Авторизация; * Создание, редактирование и удаление вопросов; * Установка количества вопросов по каждому тестированию; * Просмотр всех вопросов в базе знаний; * Просмотр результатов тестирования пользователей системы. | * Ввод информации (Ф.И.О, номер группы, номер зачётки); * Прохождение теста за определенной время; * Просмотр финального результата тестирования. |

Из этой таблицы, получена диаграмма вариантов использования. Диаграмма вариантов использования представлена на рисунке 2.



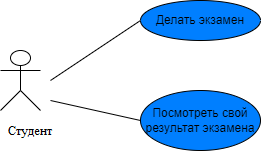


Рисунок 2 - Диаграмма вариантов использования

На основе анализа диаграммы вариантов использования можно начинать следующий шаг проектирования: определить требуемые объекты системы.

1. **Разработка диаграммы классов программы**

Одна из причин использования объектного подхода — близость к реальному миру. Такие сущности, как предметы, вопросы и экзамены, очень знакомы и близки, можно легко абстрагировать их в классы, чтобы из каждого класса можно создавать объекты с похожими свойствами и методами, соответствующие этой сущности в реальной жизни.

По вариантам использования системы можно определить требуемые объекты системы, потом определить отношения между объектами. Основываясь на этом, существуют 8 классов (и каждому классу соответствует таблица в базе данных). В соответствии с перечисленными требованиями на рисунке 3 представлена концептуальная диаграмма классов. В этом использована агрегация — отношение когда один объект является частью другого. Заметно, что в случае агрегации целое хоть и содержит свою составную часть, время их жизни не связано.

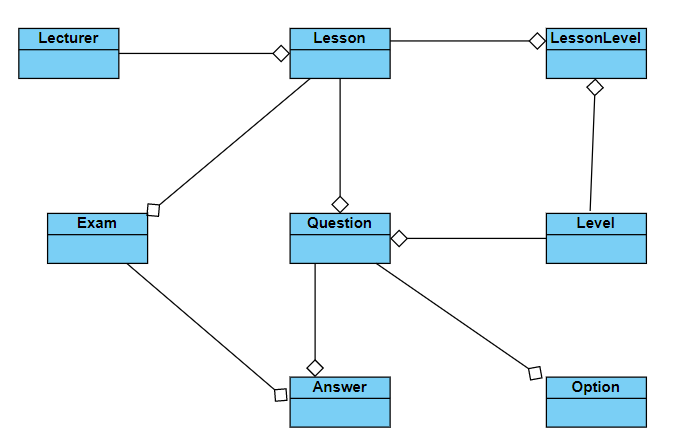
****

Рисунок3 **-** Концептуальная диаграмма классов

На следующем шаге определёны свойства и методов каждого объекта. Из вышеприведенного анализа отношения между объектами системы показаны на рисунке 4.

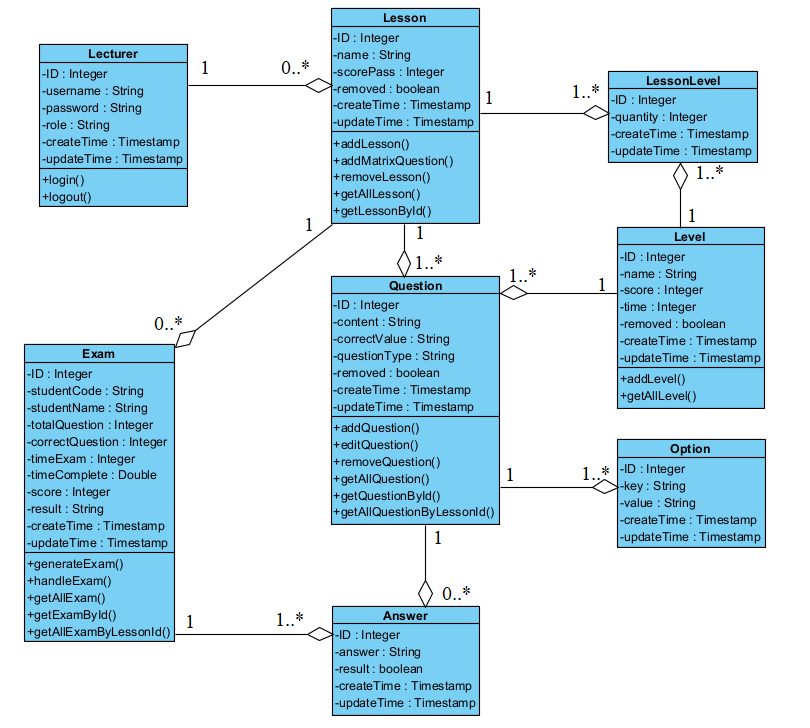


Рисунок 4 – Диаграмма классов

1. **Процесс обработки информации и анализ структуры данных**

С помощью процесса анализа диаграммы классов можно начинать построить базы данных. Техническое задание проанализовано для ясности, какие данные должны находиться в базе данных.

Необходимо связывать таблицы вместе, чтобы корректно импортировать и получать данные. Использована связь «Один ко многим» (с обязательной связью и с необязательной связью) в таблицах.

- Таблица «lecturer»: Надо связывание между таблицами «lecturer» и «lesson». Действительно, преподаватель может научить много предметов. Поэтому, связь между таблицами «lecturer» и «lesson»: 1 - N;

- Таблица «lesson»: В каждом предмете есть банк вопросов, уровни самого этого предмета и экзамены. Итак, связи между таблицами «lesson» - «question», «lesson» - «lessonlevel», «lesson» - «exam»: 1 – N;

- Таблица «lessonlevel»: связь между таблицами «lessonlevel» - «level»: 1 – N;

- Таблица «level»: Некоторые вопросы относятся к определённому уровню, поэтому связь между таблицами «level» - «question»: 1 – N;

- Таблица «question»: поскольку каждый вопрос задан с множественными выборами, связи между таблицами «question» - «option\_», «question» - «answer»: 1 – N;

- Таблица «answer»: Некоторые ответы относятся к определённому экзамену, поэтому связь между таблицами «answer» - «exam»: 1 – N.

Схема базы данных показана на рисунке 5.

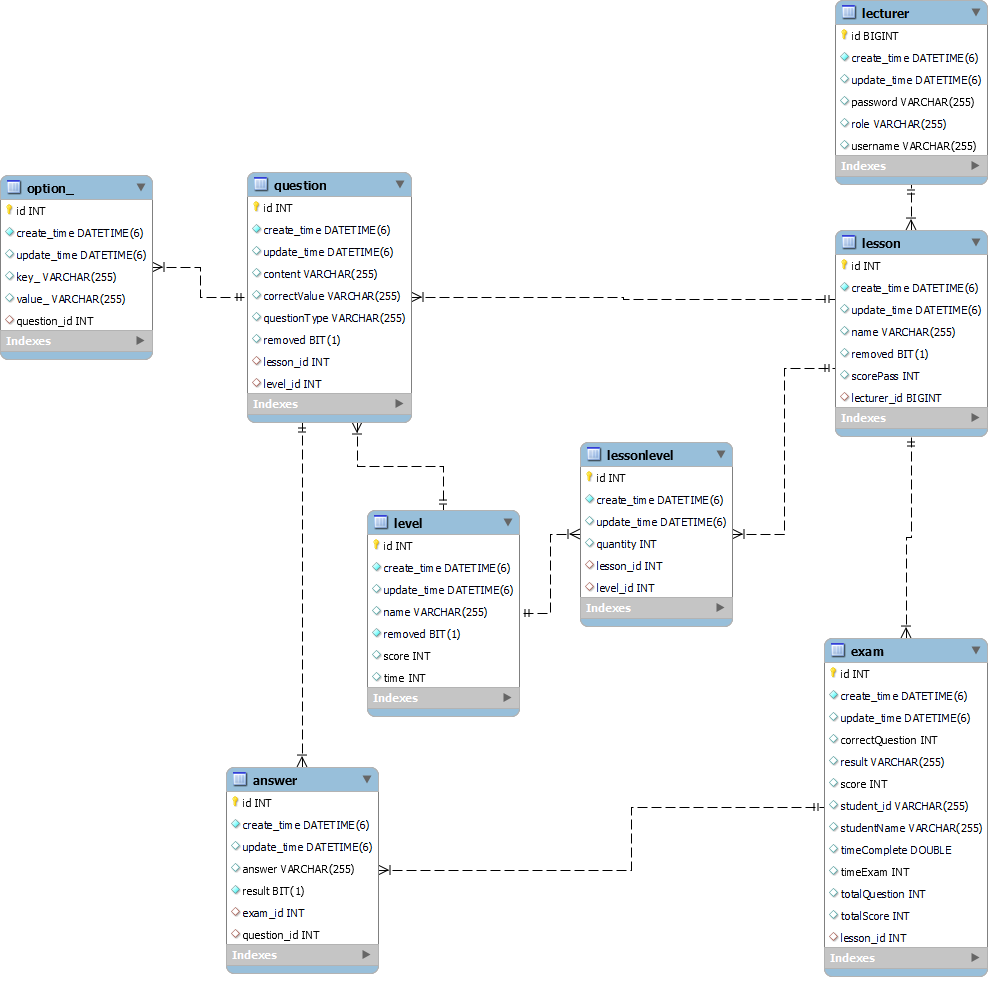


Рисунок 5 - Схема базы данных

После процесса связываний между таблицами, начинать построить часть Back End по следующим шагам:

- Создать пакет сущностей (package entity), затем создать вышеуказанные классы объектов;

- Создать интерфейсы репозитория (interface repository) для взаимодействия с базой данных (database);

- Cоздать контроллер, содержащий API (Application Programming Interface);

- В каждом API будет вызываться соответствующий сервис (service) для выполнения бизнес-логики (business logic) этого API.

1. **Разработка интерфейса пользователей**

Созданы интерфейсы с помощью HTML, CSS, JS. Используя такие библиотеки, как Bootstrap, jQuery помогает создавать более быстрые, полные и удобные интерфейсы для пользователей.

Front End общается и обменивается информациями с Back End через API. Обмен данными между двумя сторонами в формате JSON. Это распространенный формат данных в современных системах API. Когда пользователь взаимодействует с интерфейсом, Front End вызывает или отправляет необходимые данные в Back End через API, определенные в BE. Благодаря JSON, Front End и Back End могут понимать данные, которыми обмениваются друг с другом. Вызов API с использованием AJAX — это способ загрузки новых данных без перезагрузки всей страницы. Он взаимодействует только с элементами на странице и динамически обновляет данные для каждого элемента, с которым он взаимодействует.

После процесса анализа интерфейсных компонент, диаграммы состояний формы интерфейсов предстанвлены на рисунках 6 - 7.

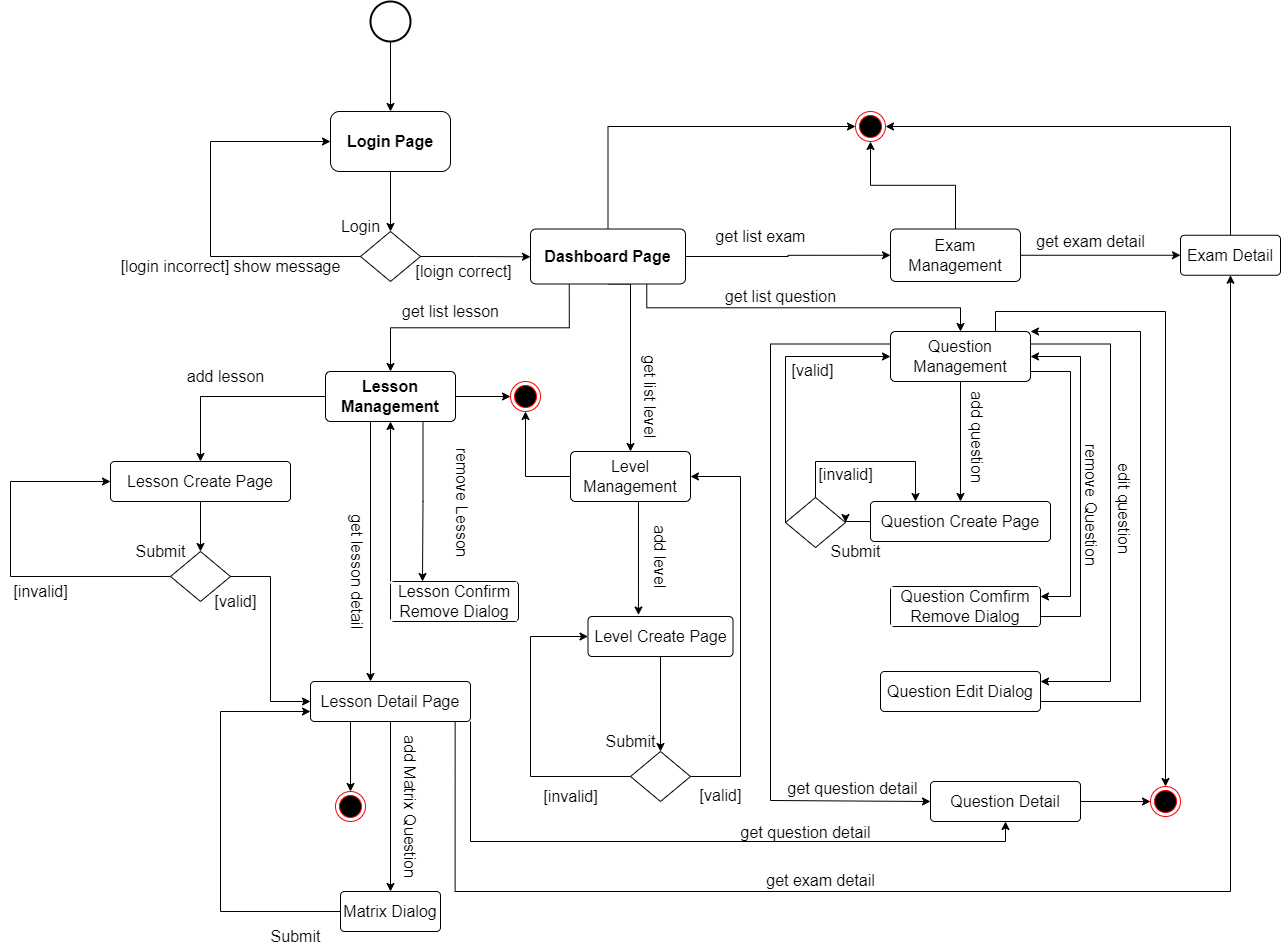


Рисунок 6 - Диаграмма состояний формы интерфейсов преподавателя

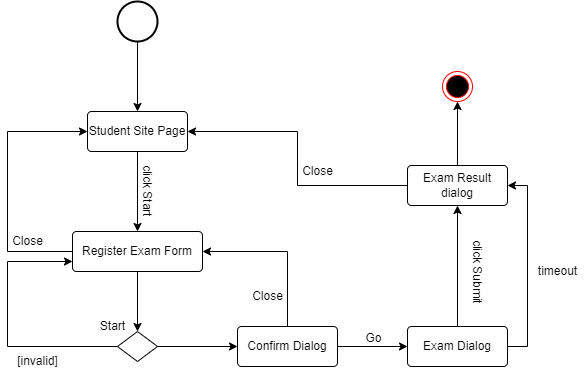


Рисунок 7 - Диаграмма состояний формы интерфейсов студента

1. **Описание форм интерфейса**

Для преподаветеля, все возможности работы которого на некоторых страницах и их формы интерфейсов представлены на рисунках 8-17.

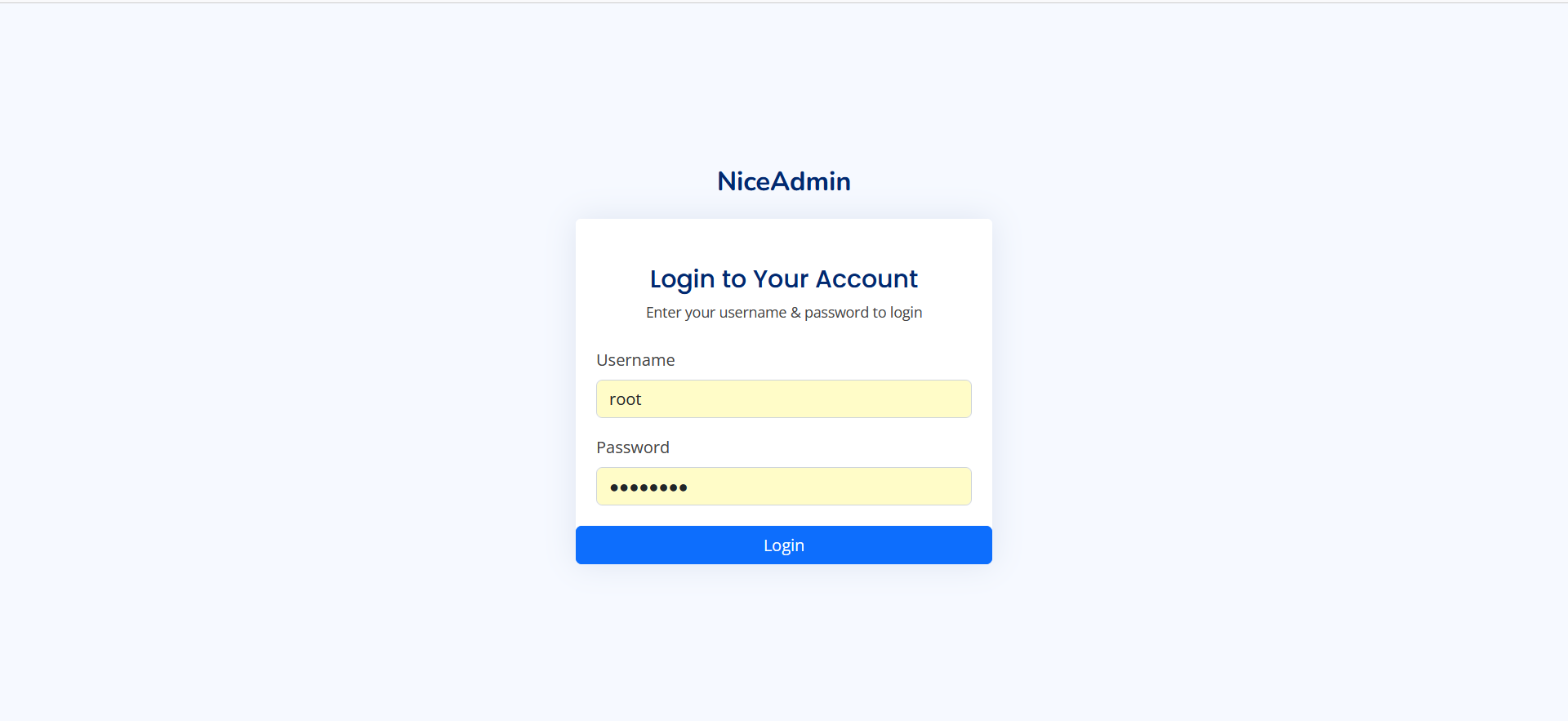


Рисунок 8 - Страница авторизации для преподавателя

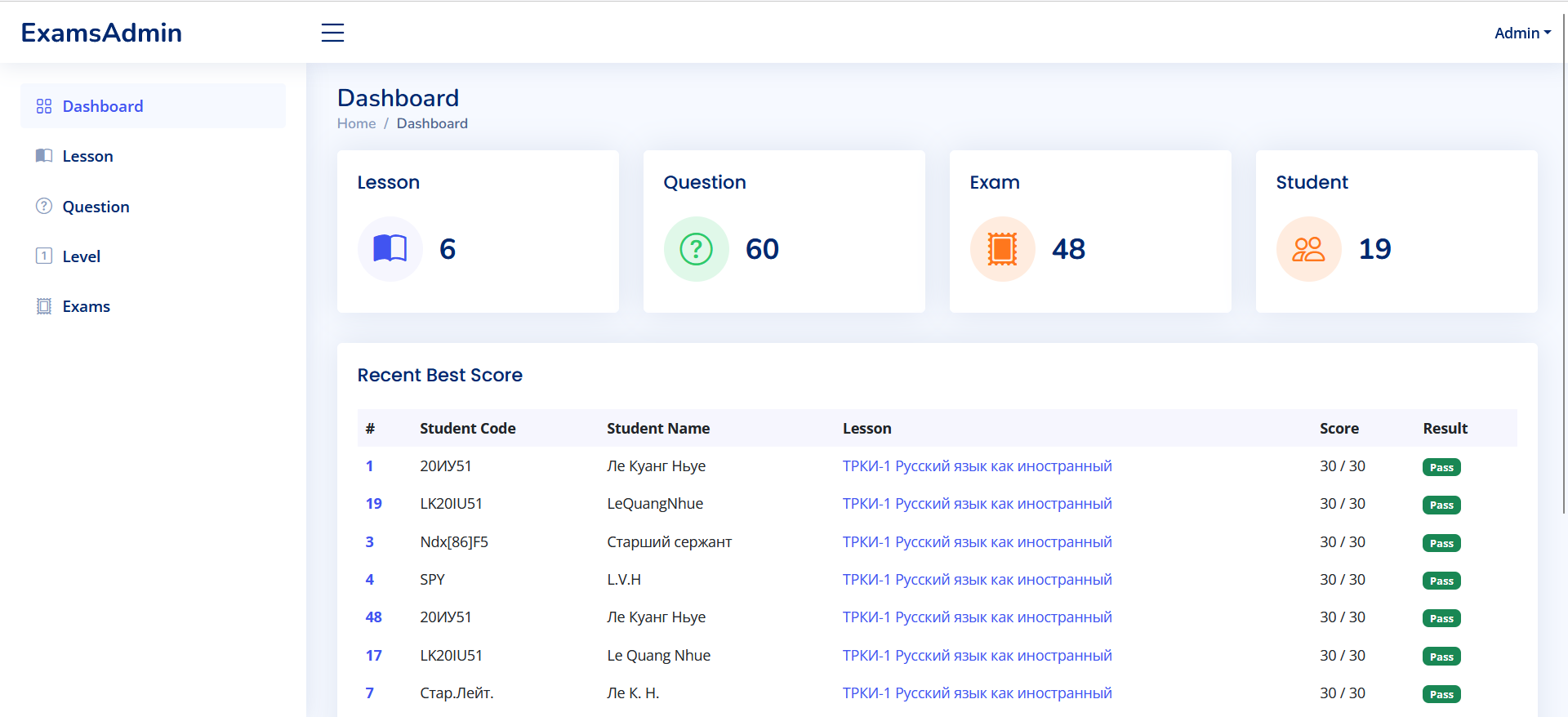


Рисунок 9 - Страница приборной доски преподавателя

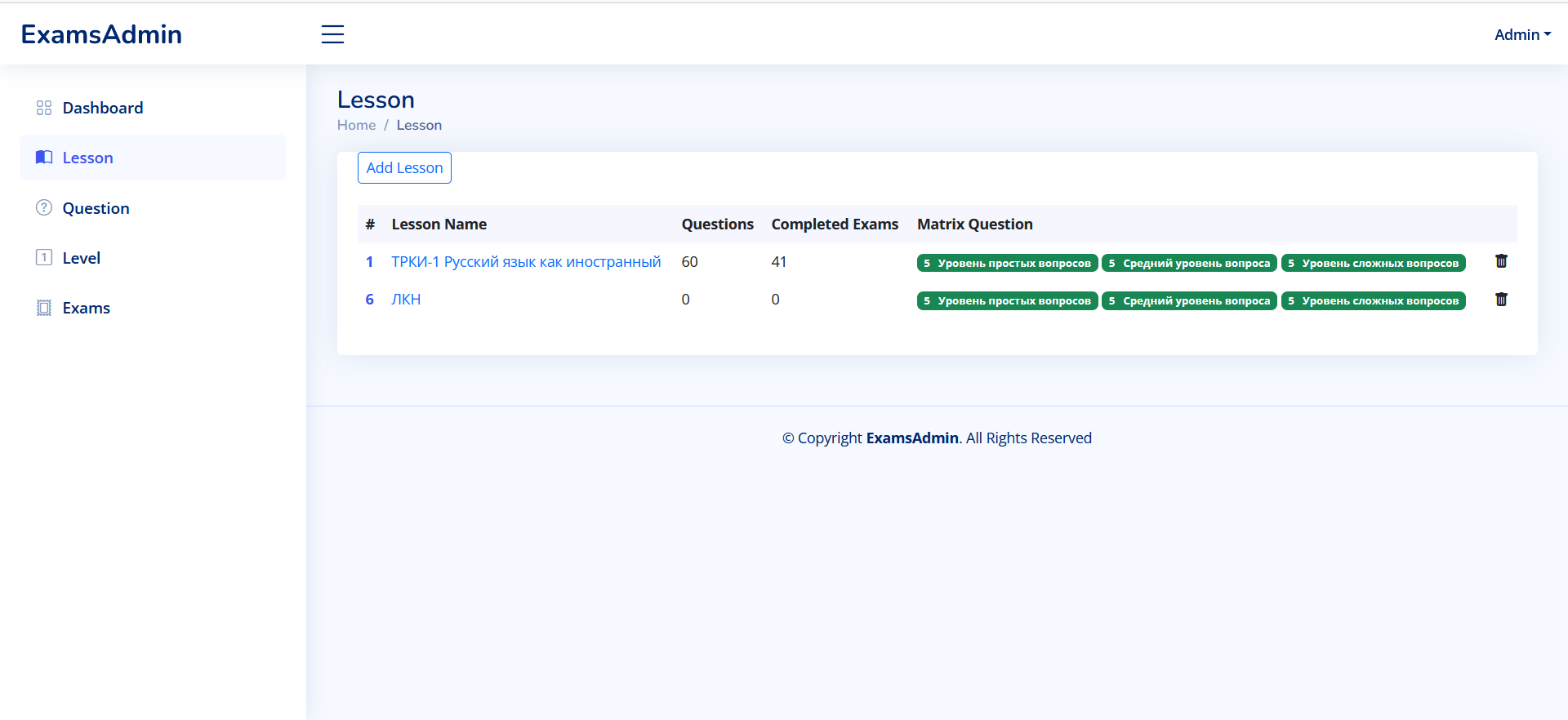


Рисунок 10 - Страница перечисления предметов, созданных преподавателем



Рисунок 11 - Страница добавления предмета с установкой некоторых полей

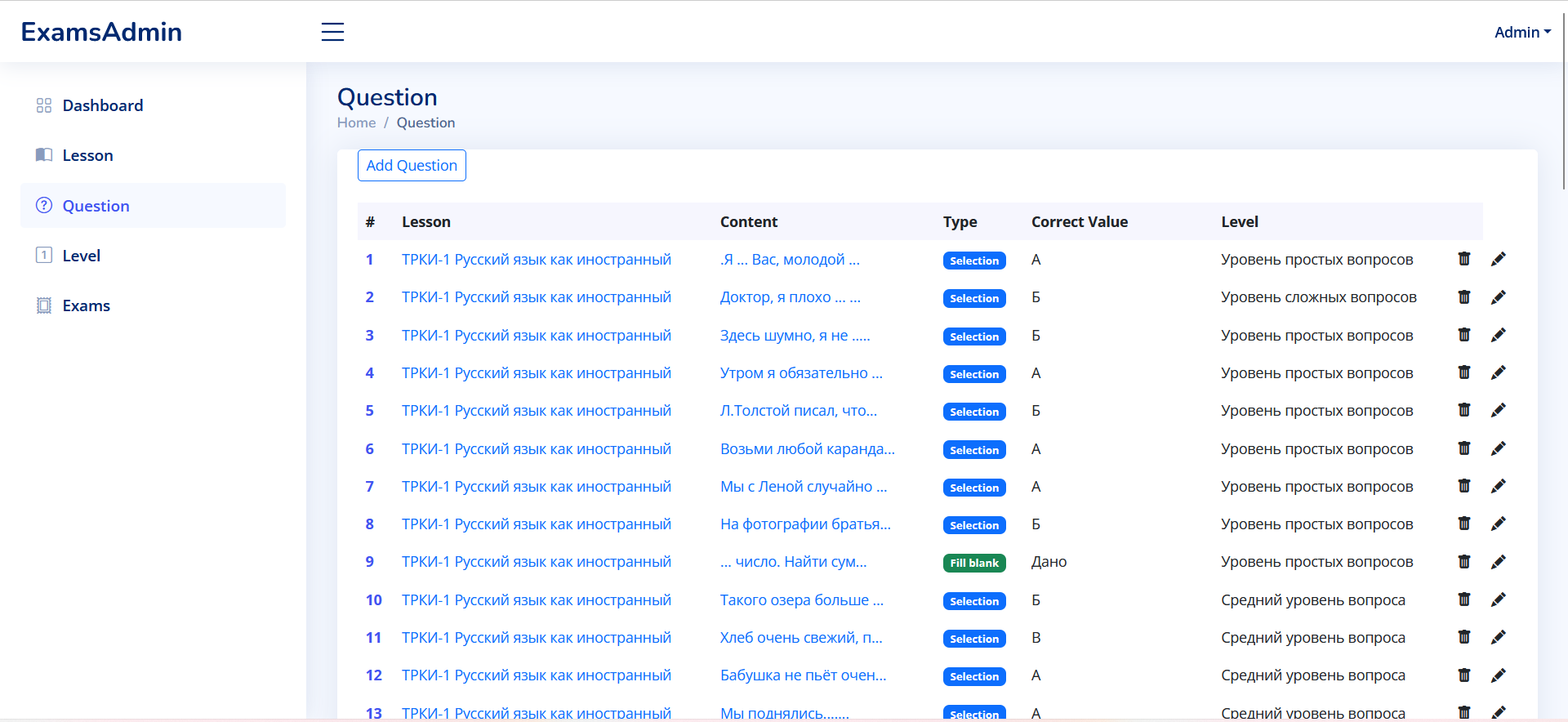


Рисунок 12 – Страница перечисления вопросов, созданных преподавателем

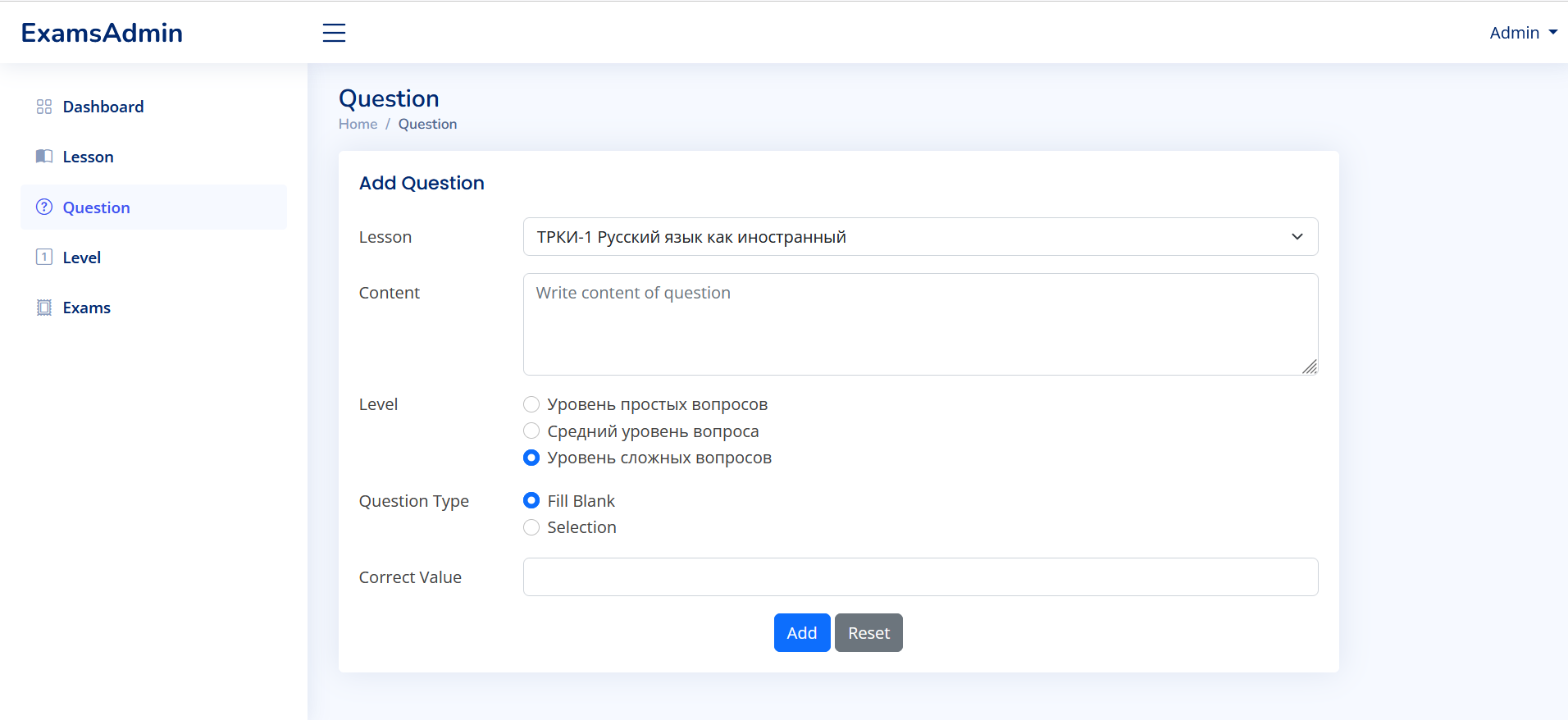


Рисунок 13 - Страница добавления вопроса с установкой некоторых полей



Рисунок 14 - Страница перечисления уровней, созданных преподавателем

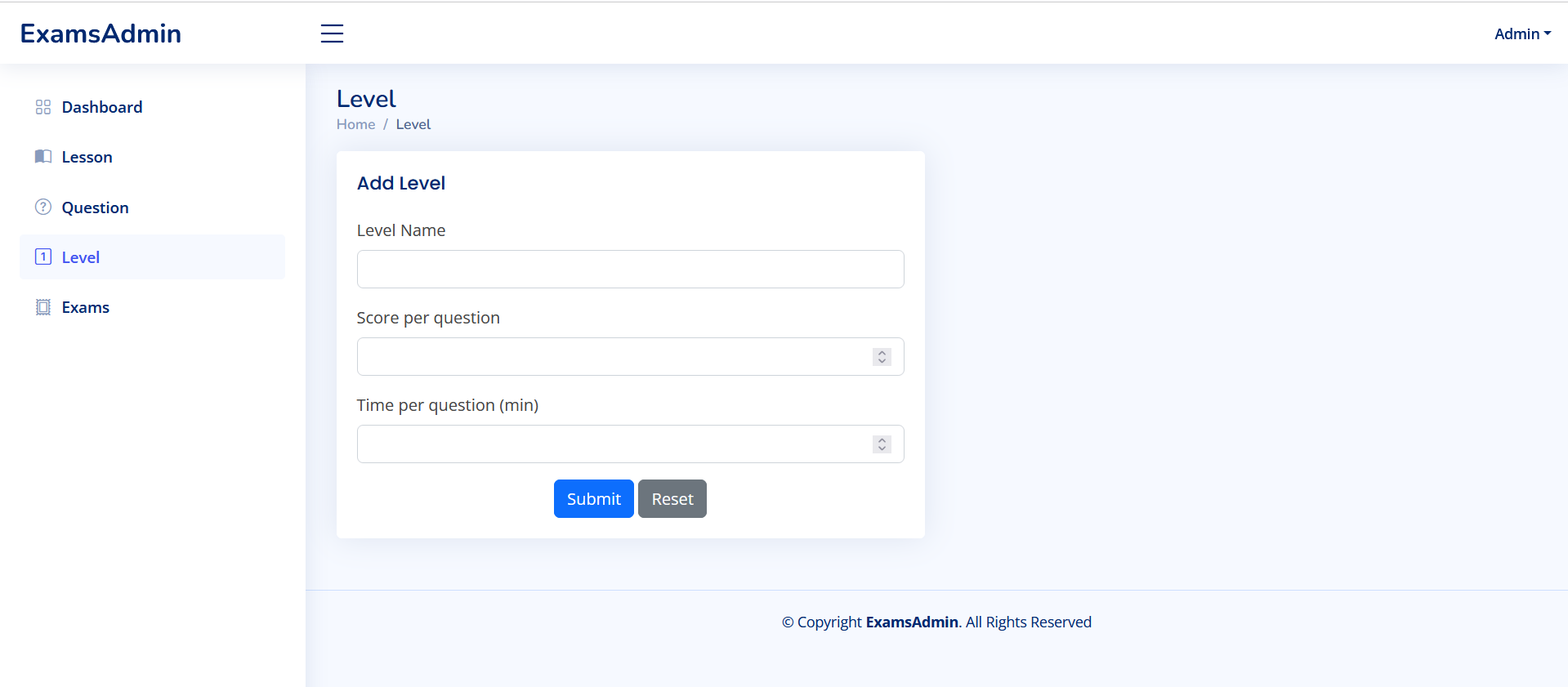


Рисунок 15 - Страница добавления уровня с установкой некоторых полей

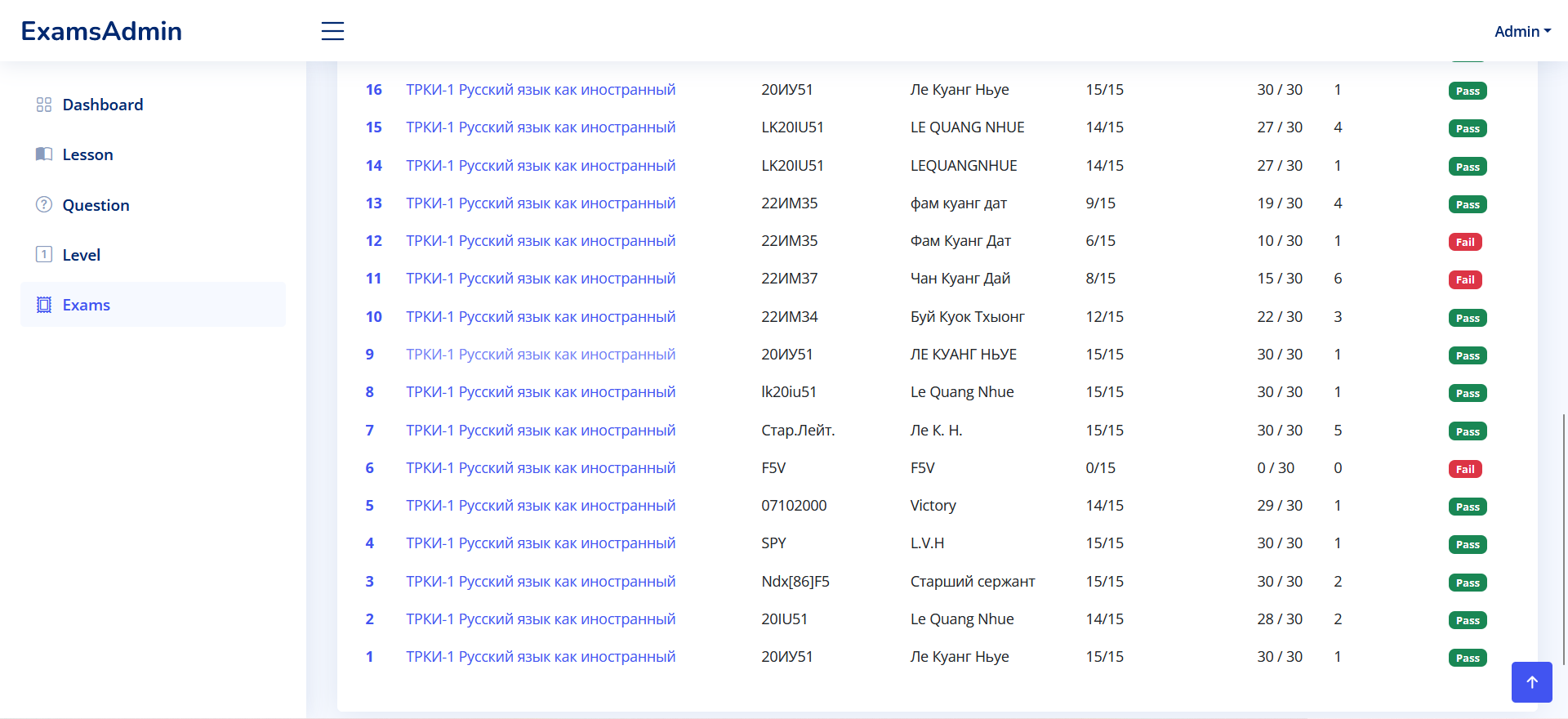


Рисунок 16 - Страница свода выполненных экзаменов, в котором хранены все данные какого-то экзамена

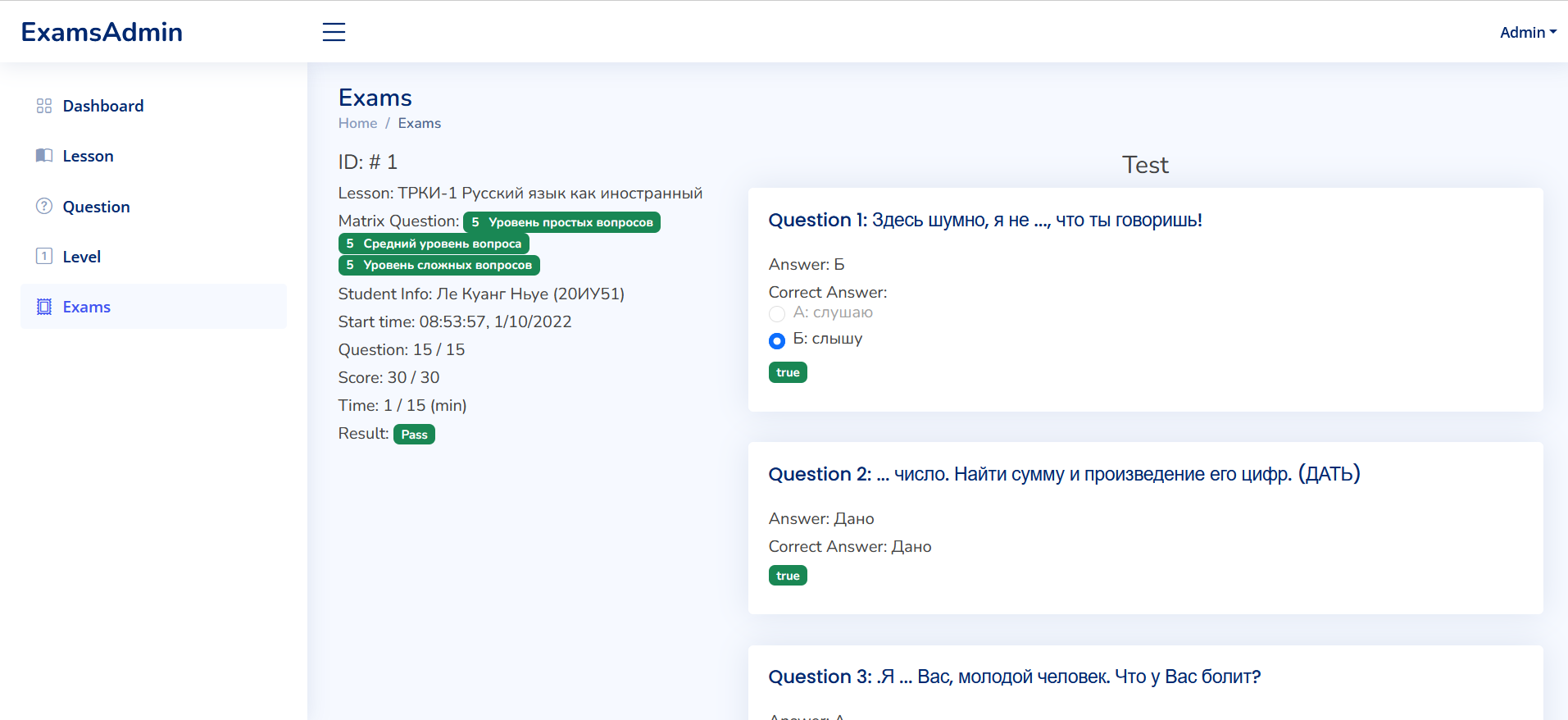


Рисунок 17 - Страница деталей одного экзамена

Для студентов, все возможности работы которых на некоторых страницах и их формы интерфейсов представлены на рисунках 18 - 21.

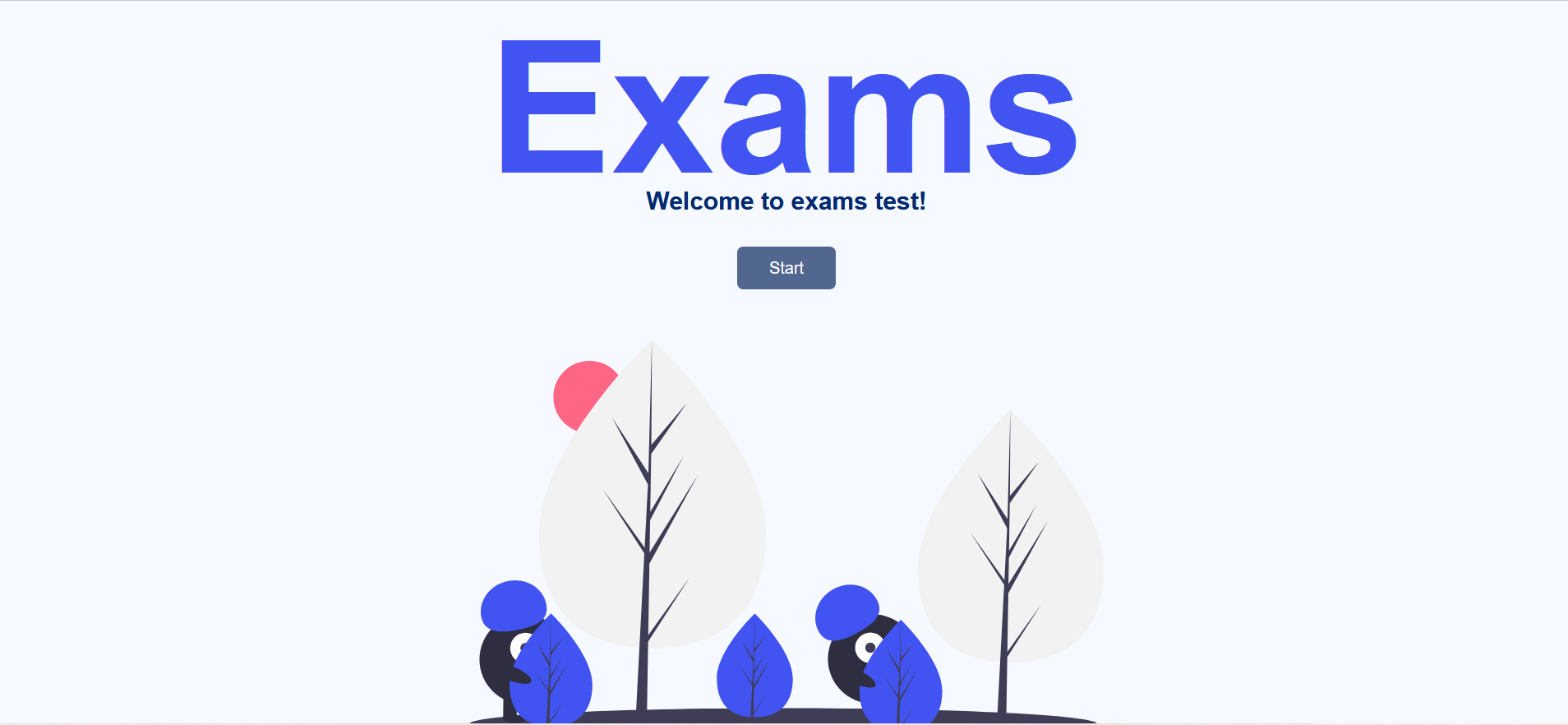


Рисунок 18 – Начальная страница для студента

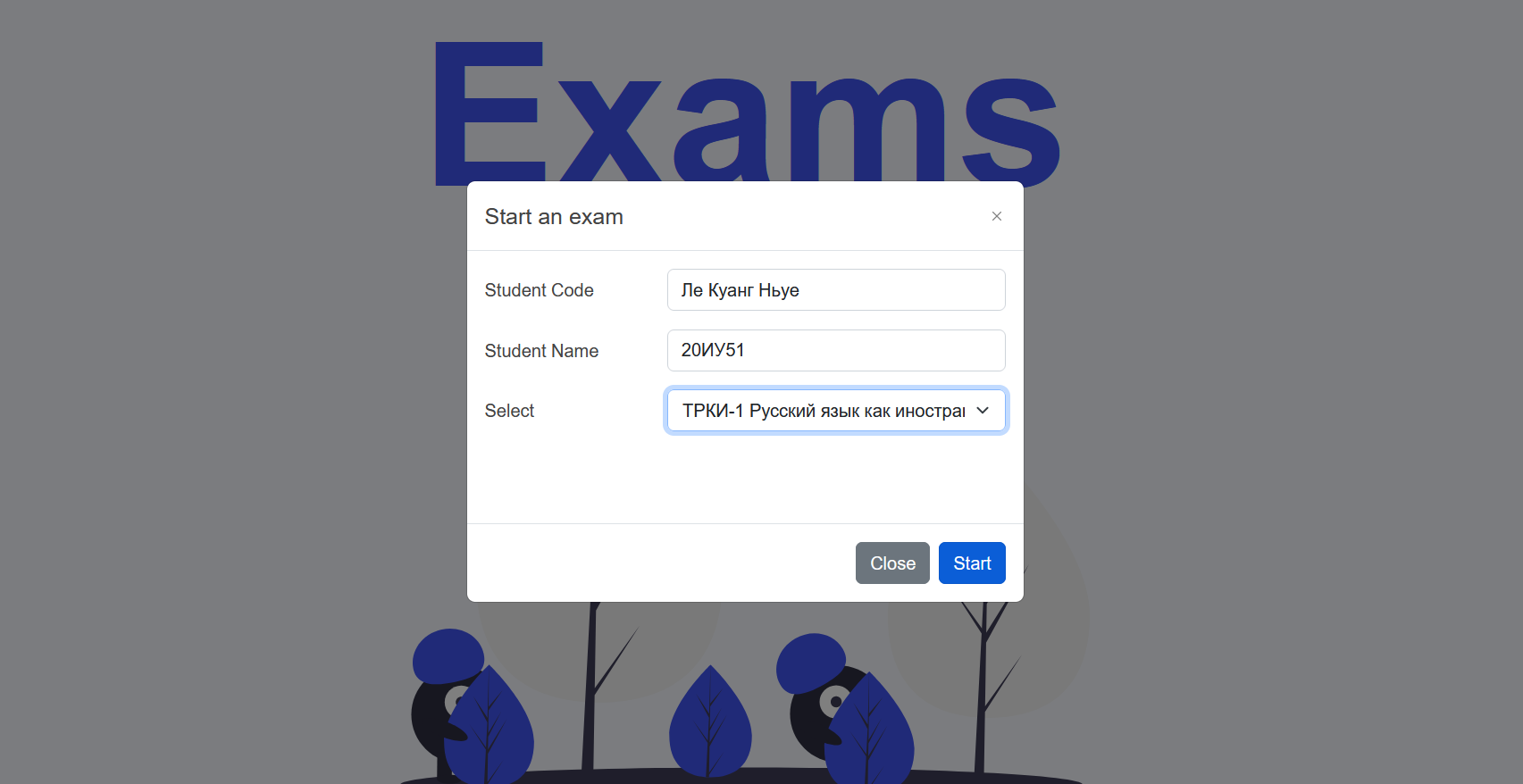


Рисунок 19 – Страница окна выполнения полей для студента

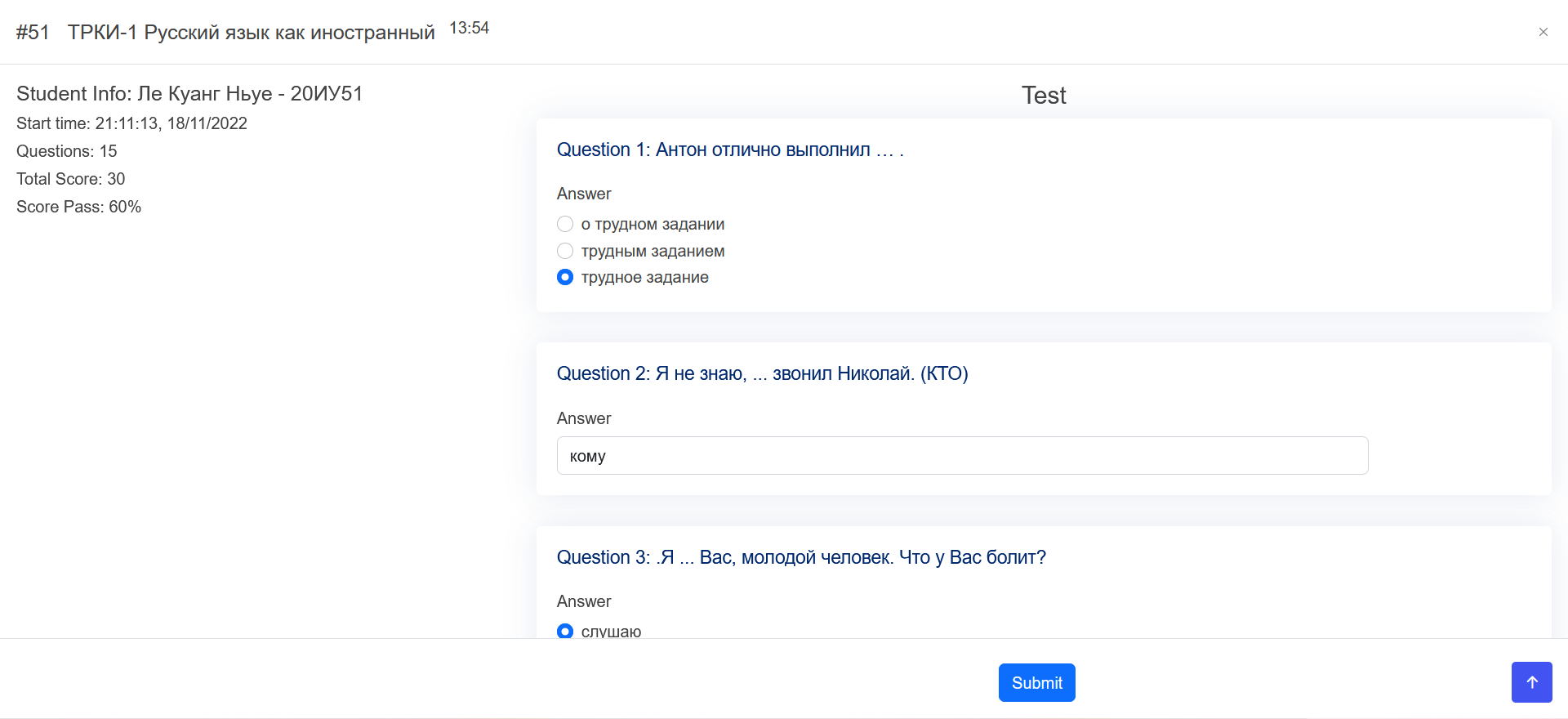


Рисунок 20 – Страница выполнения экзамена для студента

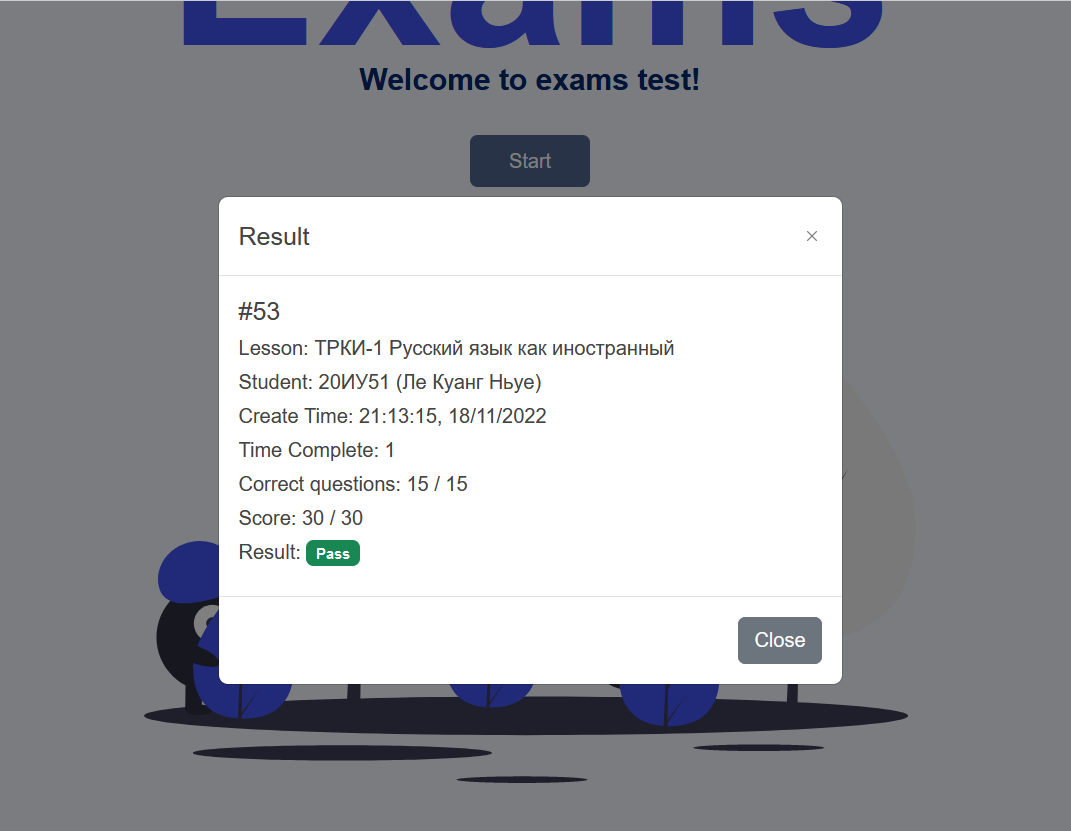


Рисунок 21 – Страница смотра финального результата экзамена для студента

# **Глава 3. Тестирование программного продукта**

Тестирование веб - приложения должно выполняться в несколько этапов. Синтаксические ошибки, возникающие при написании кода, тут же сразу исправляются программистами.

Различают два подхода к формированию тестов: структурный и функциональный. Каждый из указанных подходов имеет свои особенности и области применения. Структурное тестирование предполагает создание тестов на основе структуры системы и её реализации. А функциональное тестирование базируется на том, что все тесты основываются на проверке выполняемых им функций. Для разработаемого веб – приложения, пользователям интересно важные выполняемые функции. Поэтому необходимо выбираться функциональное тестирование. При этом программа представляется как «черный ящик» (Black-box), поведение которого можно определить только посредством изучения ее входных и соответствующих выходных данных. При функциональном тестировании различают следующие методы формирования тестовых наборов:

- эквивалентное разбиение;

- анализ граничных значений;

- анализ причинно-следственных связей;

- предположение об ошибке.

Проведенное тестирование описано по принципу «Черного ящика», рассмотрев некоторые вида функционального тестирования.

1. Анализ причинно-следственных связей

Анализ причинно-следственных связей позволяет системно выбирать высокорезультативные тесты. Метод использует алгебру логики и оперирует понятиями «причина» и «следствие».

Интересно используется это функциональное тестирование для этапа авторизации преподавателя.

Таблица 2 – Тест дизайна при авторизации

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Логин | Пустой | | Не пусто и не совпадает | | Совпадает | |
| Пароль | Пустой | Не пусто | Пустой | Не пусто | Пустой | Совпадает |
| Результат действия | Тупик | Тупик | Тупик | Тупик | Тупик | Успешно |

Таблица 3 - Тестирование методом анализа причинно-следственных связей для преподавателя при авторизации

|  |  |  |  |
| --- | --- | --- | --- |
| Описание теста | Ожидаемый результат | Полученный результат | Вывод |
| Поля логина и пароля пусты, затем нажать на кнопку «Login» | Авторизуйтесь не успешно и не перейдите на главную страницу | Авторизуйтесь не успешно и не перейдите на главную страницу | Успешно |
| Поля логина и пароля заполнены, но неправильные, затем нажать на кнопку «Login» | Авторизуйтесь не успешно и не перейдите на главную страницу | Авторизуйтесь не успешно и не перейдите на главную страницу | Успешно |
| Поля логина и пароля правильно заполнены, затем нажать на кнопку «Login» | Авторизуйтесь успешно и перейдите на главную страницу | Авторизуйтесь успешно и перейдите на главную страницу | Успешно |

Таблица 4 - Тестирование методом анализа причинно-следственных связей для преподавателя при других функциях преподавателя (после успешной авторизации)

|  |  |  |  |
| --- | --- | --- | --- |
| Описание теста | Ожидаемый результат | Полученный результат | Вывод |
| Нажать на кнопку | Показать или скрыть меню страницы | Показать или скрыть меню страницы | Успешно |
| Нажать на кнопку | Перейдите на страницу приборной доски преподавателя | Перейдите на страницу приборной доски преподавателя | Успешно |
| Нажать на кнопку | Перейдите на страницу предметов | Перейдите на страницу предметов | Успешно |
| Нажать на кнопку | Перейдите на страницу банка вопросов | Перейдите на страницу банка вопросов | Успешно |
| Нажать на кнопку | Перейдите на страницу уровней | Перейдите на страницу уровней | Успешно |
| Нажать на кнопку | Перейдите на страницу свода выполненных экзаменов | Перейдите на страницу свода выполненных экзаменов | Успешно |
| На странице уровней, нажать на «Add Level», заполнение правильно полей и нажать «Submit» | Новый уровень добавлен | Новый уровень добавлен | Успешно |

Продолжение таблицы 4

|  |  |  |  |
| --- | --- | --- | --- |
| На странице уровней, нажать на «Add Level», заполнение правильно полей и нажать «Reset» | Все выполненные поля сразу пусты | Все выполненные поля сразу пусты | Успешно |
| На странице уровней, нажать на «Add Level», нет заполнения полей и нажать «Submit» | Выходит сообщение, что поле нельзя оставлять пустым | Выходит сообщение, что поле нельзя оставлять пустым | Успешно |
| На странице предметов, нажать на «Add Lesson», заполнение правильно полей и нажать «Submit» | Новый предмет добавлен | Новый предмет добавлен | Успешно |
| На странице банка вопросов, нажать на «Add Question», заполнение правильно полей и нажать «Add» | Новый вопрос добавлен | Новый вопрос добавлен | Успешно |
| На странице предметов удалить один предмет | Этот предмет исчез | Этот предмет исчез | Успешно |
| На странице банка вопросов удалить один вопрос | Этот вопрос исчез | Этот вопрос исчез | Успешно |
| На странице банка вопросов изменить содежимое одного вопроса | Изменения вопроса обновлены | Изменения вопроса обновлены | Успешно |

Продолжение таблицы 4

|  |  |  |  |
| --- | --- | --- | --- |
| На странице свода выполненных экзаменов, выбирать и нажать на любой номер экзамена | Перейдите на страницу деталей этого выполненного экзамена | Перейдите на страницу деталей этого выполненного экзамена | Успешно |

Данный метод позволяет строить высокорезультативные тесты и обнаруживать неполноту и неоднозначность исходных спецификаций. Его недостатком является неадекватное исследование граничных значений.

1. Эквивалентное разбиение

Метод эквивалентного разбиения заключается в следующем. Область всех возможных наборов входных данных программы по каждому параметру разбивают на конечное число групп - классов эквивалентности.

Используется этот метод на шаге добавления матрицы вопросов одного предмета (см. рисунку х). Существуют 3 поля, значения которых должны быть целыми и положительными числами.

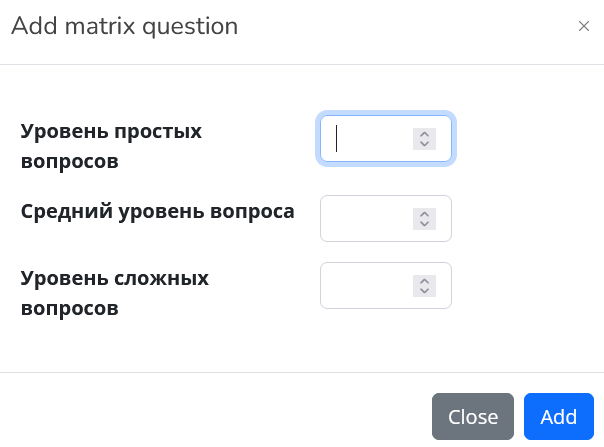


Рисунок 22 – Добавить матрицу вопросов одного предмета

Таблица 5 – Таблица эквивалентности

|  |  |  |
| --- | --- | --- |
| Входное условие | Правильные классы эквивалентности | Неправильные классы эквивалентности |
| Поля входа (Inp) должны быть числами | Поля с целыми числами | Поля со стороками и символами |
| Поля входа (Inp) должны быть целыми | Inp целое | Inp не целое |
| Поля входа (Inp) должны быть положительными | Inp > 0 | Inp  0 |

Таблица 6 – Тестирование по таблице эквивалентности

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назначение теста | Значение исходных данных | Ожидаемый результат | Реакция программы | Вывод |
| Поля входа (Inp) должны быть числами | Inp = {“Le”, “Quang”, “Nhue”} | Матрица вопросов не может быть успешно обновлена | Матрица вопросов не может быть успешно обновлена | Надо выполнить поля с числами |
| Поля входа (Inp) должны быть целыми | Inp = {5,5; 6,5; 7,5} | Матрица вопросов не может быть успешно обновлена | Матрица вопросов не может быть успешно обновлена | Надо выполнить поля с целыми числами |
| Поля входа (Inp) должны быть положительными | Inp =  {-1; -2; -3} | Матрица вопросов не может быть успешно обновлена | Матрица вопросов не может быть успешно обновлена | Надо выполнить поля с целыми положительными числами |

1. **Анализ граничных условий**

Применение метода анализа граничных значений требует определенной степени творчества и специализации в рассматриваемой проблеме. Используется этот метод на стороне студента. Установив количество процентов баллов успешного прохождения экзамена (для экзамена «ТРКИ-1 Русский язык как иностранный» - Score pass = 60%), два студента сделают этот экзамен (один сделает успешно, а другой неуспешно). Детали экзамена «ТРКИ-1 Русский язык как иностранный» представлены на рисунке 23.

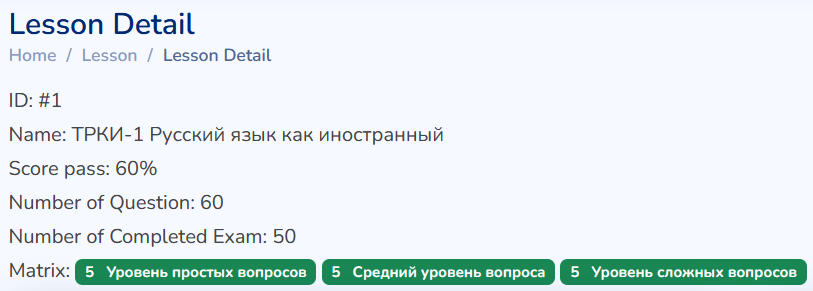




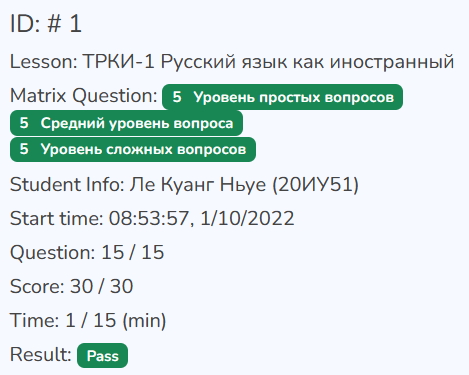
Рисунок 23 - Детали экзамена «ТРКИ-1 Русский язык как иностранный»

Таблица 7 – Тестирование методом анализа граничных условий

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Назначение теста | Значение исходных данных | Ожидаемый результат | Реакция программы | Вывод |
| Студент не сдает экзамен  (cм. рис. 24) | У этого студента Score pass < 60% | Сообщеия «Fail» | Сообщеия «Fail» | Успешно |

Продолжение таблицы 7

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент успешно сдает экзамен  (cм. рис. 25) | У этого студента Score pass  60% | Сообщеия «Pass» | Сообщеия «Pass» | Успешно |



Рисунки 24 и 25 – Сообщения «Pass» и «Fail»

Анализ граничных значений, если он применен правильно, является одним из наиболее полезных методов проектирования тестов.

**ЗАКЛЮЧЕНИЕ**

В результате выполнения курсовой работы были изучены среда разработки Eclipse и язык программирования Java, и как итог – разработано веб – приложение «Программа ТЕСТ-ТЕСТЫ», с помощью которого можно выучить и проверить знания русских лексик и грамматик. Даётся вывод, что это приложение стало технологически продвинутым, потому что части Front End и Back End хорошо разработаны, структура данных четко ясно определена. В дополнение к этому, система основана на языке программирования JAVA (для серверной части) и jQuery (для интерфейсной части) и объединена с помощью баз данных MySQL.

Так же в процессе разработки были составлены различные диаграммы и схемы для детального понимания работы программы, соответствующие с использованием объектного подхода. Используя их во время разработки, программист правильно написал код программы и как итог корректной работы веб - приложения. После получения готового веб - приложения было проведено тестирование по принципу «Черного ящика» для проверки правильной и безошибочной работы веб - приложения.

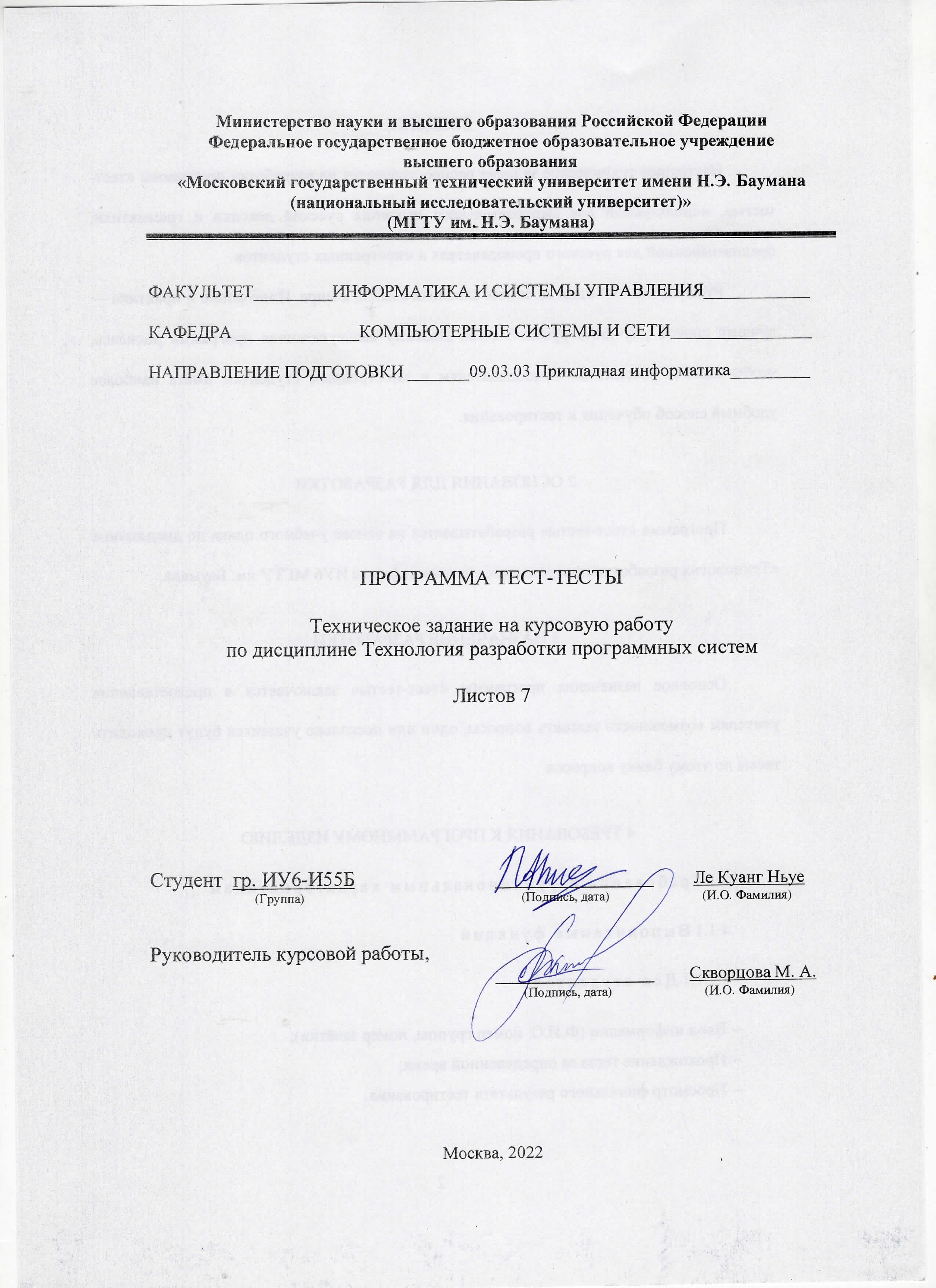
Итоговое веб-приложение соответствует предыдущему ТЗ и выполняет все требования. В дальнейшем развитии проекта планируется добавить техлоногию машинного обучения или прикладного анализа данных, чтобы дал пользователю - преподавателю более ясные статистики анализа результатов экзаменов студента.

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Иванова Г.С. Технология программирования: учебник / Г.С. Иванова. – 3-е изд., стер. – М.: КНОРУС, 2016. – 334 с. – (Бакалавриат). – ISBN 978-5-04734-7.
2. Иванова Г. С., Ничушкина Т.Н., Пугачев Е.К., Самарев Р.С., Фетисов М.В., Бычков Б.И. Методические указания по выполнению курсовой работы по дисциплине «Технология разработки программных систем». Электронное учебное издание. - М.: МГТУ имени Н.Э. Баумана, 2021, 42 с.
3. Официальная документация Spring. Электронный ресурс. Режим доступа: https://spring.io/ (дата обращения 15.11.2022)
4. Никсон Робин, Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и НТML5: пер. с англ./ 6-е изд. – СПБ.: Питер, 2022. – 832 с.: ил. – (Серия «Бестселлеры O’Reilly»).
5. Хеклер Марк, Spring Boot по-быстрому. : пер. с англ./ – СПБ.: Питер, 2022. – 352 с.: ил. – (Серия «Бестселлеры O’Reilly»).

# **ПРИЛОЖЕНИЯ А**

**Техническое задание**Листов 7



# 1 Введение

Настоящее техническое задание распространяется на разработку программы «тест-тесты», используемой для смостоятельного изучения русской лексики и грамматики, предназначенной для русского преподавателя и иностранных студентов.

Русский язык — один из самых сложных языков в мире. Повторение и практика — лучший способ улучшить русский язык. Поэтому вышеуказанная программа родилась, чтобы помочь российским преподавателям и иностранным студентам иметь наиболее удобный способ обучения и тестирования.

# 2 Основания для разработки

Программа «тест-тесты» разрабатывается на основе учебного плана по дисциплине «Технология разработки программных систем» кафедры ИУ6 МГТУ им. Баумана.

# 3 Назначение разработки

Основное назначение программы «тест-тесты» заключается в предоставление учителям возможность задавать вопросы, один или несколько учащихся будут проходить тесты по этому банку вопросов.

# 4 Требования к программному изделию

## 4.1 Требования к функциональным характеристикам

## 4.1.1 Выполняемые функции

## 4.1.1.1 Для студентов:

* Ввод информации (Ф.И.О, номер группы, номер зачётки);
* Прохождение теста за определенной время;
* Просмотр финального результата тестирования.

## 4.1.1.2 Для администратора системы (преподавателя):

* Авторизация;
* Создание, редактирование и удаление вопросов открытого и закрытого типа;
* Установка количества вопросов по каждому тестированию;
* Просмотр всех вопросов в базе знаний;
* Просмотр результатов тестирования пользователей системы.

## Исходные данные:

* Ф.И.О студента: ввод с клавиатуры в текстовое поле;
* Номер группы студента: ввод с клавиатуры в текстовое поле;
* Номер зачётки студента: ввод с клавиатуры в текстовое поле;
* Выбор ответа студента: ввод с помощью клика мыши или клавиатуры;
* Аутентификация (логин и пароль): ввод с клавиатуры в текстовое поле;
* Множество вопросов и их параметры: ввод с клавиатуры в текстовое поле.

## 4.1.3 Результаты:

* Сообщение об успешной регистрации для преподавателя;
* Сообщение об успешном создании вопроса для преподавателя;
* Сообщение об успешном заполнении полей для студентов;
* Cообщение о начале теста для студентов;
* Cообщение о конце теста для студентов.

## 4.2 Требования к надежности

4.2.1 Предусмотреть контроль вводимой информации.

4.2.2 Предусмотреть блокировку некорректных действий пользователя.

4.2.3 Обеспечить целостность информации в базе данных.

## 4.3 Условия эксплуатации

4.3.1 Условия эксплуатации в соответствии с СанПиН 2.2.2/2.4.1340-03.

4.3.2 Обслуживание

4.3.3 Обслуживающий персонал

## 4.4 Требования к составу и параметрам технических средств

4.4.1 Программное обеспечение должно функционировать на IBM-совместимых персональных компьютерах.

4.4.2 Минимальная конфигурация технических средств:

4.4.2.1 Тип процессора x86 или x86-64 совместимый

## 4.4.2.2 Объем ОЗУ 2 Гб.

## 4.5 Требования к информационной и программно совместимости

4.5.1 Программное обеспечение должно работать под управлением операционных систем семейств Windows 10/ Windows 11 или, что предпочтительнее, с семейством 64-х разрядных ОС Linux.

4.5.2 Рекомендуется следующее программное обеспечение: при использовании сервера с ОС Linux х64: операционная система – Debian 8, CentOS 7, Red Hat Enterprise Linux 7, Ubuntu 14.04; при использовании сервера с ОС Windows x64: установленное приложение Java VM (JDK).

4.5.3 JDK не ниже v.11

4.5.4 MySql не ниже v.8

4.5.5 Eclipse не ниже v.4.24.0

## 4.6 Специальные требования

1. Создать дистрибутивную версию программного обеспечения.

# Требования к программной документации

5.1 Разрабатываемые программные модули должны быть самодокументированы, т.е. тексты программ должны содержать все необходимые комментарии.

5.3 В состав сопровождающей документации должны входить:

5.3.1 Расчетно-пояснительная записка на 25-30 листах формата А4 (без приложений 5.3.2, 5.3.3 и 5.3.4).

5.3.2 Техническое задание (Приложение А).

5.3.3 Руководство пользователя (Приложение Б).

5.4 Графическая часть должна быть включена в расчетно-пояснительную записку в качестве иллюстраций:

5.4.1 Диаграмма вариантов использования.

5.4.2 Концептуальная диаграмма классов.

5.4.3 Диаграмма классов.

5.4.4 Диаграммы (схемы) компонентов структур данных.

5.4.5 Граф состояний интерфейса.

5.4.6 Формы интерфейса.

5.4.7 Таблицы тестов.

# 6 Стадии и этапы разработки

| **Этап** | **Содержание этапа** | **Сроки и**  **объем** | **Представляемые результаты** | |
| --- | --- | --- | --- | --- |
| **Спецификации и программный продукт** | **Документы** |
|  | Выбор темы, составление задания, решение организационных вопросов | 1..2 недели  (10 %) | - | **Заполненный бланк задания на курсовую работу – вывешивается на сайт кафедры для получения утверждающей подписи заведующего кафедрой** |
|  | Анализ предметной области, разработка ТЗ. Исследование методов решения, выбор основных проектных решений | 3..4 недели | Результаты декомпозиции предметной области.  Эскизный проект: интерфейс, схемы, возможно, часть программы (выбранные готовые решения). | Фрагмент расчетно-пояснительной записки с обоснованием выбора средств и подходов к разработке |
|  | **Сдача ТЗ** | **4 неделя**  **(25 %)** | **-** | **Техническое задание – утверждается руководителем** |
|  | Проектирование и реализация основных компонентов – ядра программы | 5..7 недели | Технический проект основной части: структура программы, алгоритмы программ, описания структур данных, диаграмма классов – в зависимости от выбранной технологии разработки.  Программный продукт, реализующий основные функции (демонстрируется руководителю) | Фрагмент расчетно-пояснительной записки с обоснованием разработанных спецификаций  Тексты части программного продукта, реализующего основные функции. |
|  | **Сдача прототипа программного продукта** | **7 неделя**  **(50 %)** | **Прототип программного продукта – демонстрируется руководителю** |  |
|  | Разработка компонентов, обеспечивающих функциональную полноту | 8..10 | Рабочий проект программы.  Готовая программа | Черновик расчетно-пояснительной записки.  Тексты программного продукта. |
|  | **Сдача программного продукта** | **11 неделя**  **(75 %)** | **Готовая программа – оценивается руководителем в баллах** | **-** |
|  | Тестирование программы и подготовка документации | 12..14 | Тесты и результаты тестирования. | РПЗ и Руководство пользователя. |
| **9.** | **Оформление и сдача документации** | **14 неделя**  **(90 %)** | **–** | **Расчетно-пояснительная записка и Руководство пользователя – проверяются и подписываются руководителем** |
| 10. | Защита курсовой работы | 15..16 недели  (100%) | – | Доклад (3-5 минут). Защита курсовой работы.  Подписанная документация – вывешивается на сайт кафедры |

# 7 Порядок контроля и приемки

## 7.1 Порядок контроля

Контроль выполнения осуществляется руководителем еженедельно.

## 7.2 Порядок защиты

Защита осуществляется комиссии преподавателей кафедры.

## 7.3 Срок защиты

Срок защиты: 15-16 недели.

# 8 Примечание

В процессе выполнения работы возможно уточнение отдельных требований технического задания по взаимному согласованию руководителя и исполнителя.

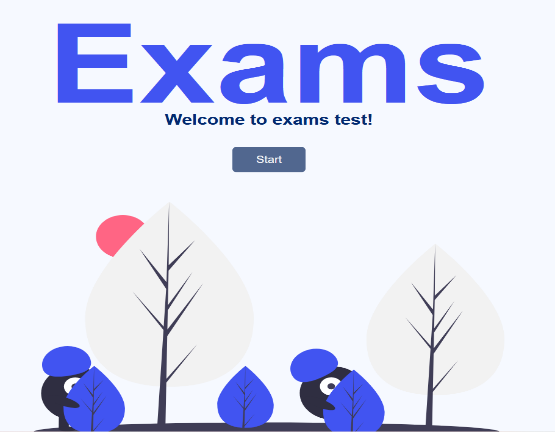
# **ПРИЛОЖЕНИЯ Б**

**Руководство пользователей**Листов 10

1. **Руководство для студентов**

Веб – приложение «Программа ТЕСТ-ТЕСТЫ» даёт студентам возможность прохождения экзамена и посмотр финальный результат этого экзамена.

Нажимая на кнопку «Start» , студенту надо выполнить все поля информаций и выбирать экзамен (см. рисунку 1).



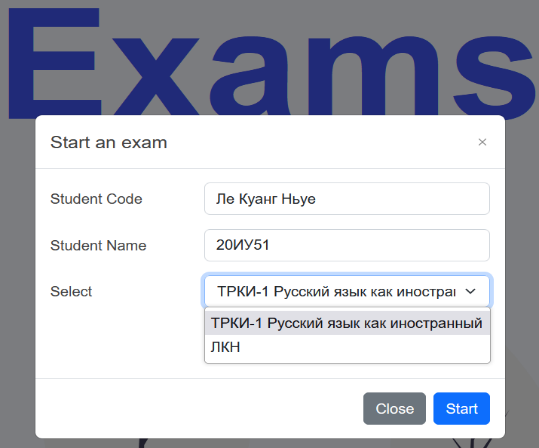


Рисунок 1 – Выполнение полей информаций

Выполнив все поля, студент нажимает на кнопку «Start». Будет появлена одна форма (см. на рисунку 2). Если студент действительно хочет сразу делать экзамен, то выбирает «Go». В противном случае, он нажимает на «Close» и вернуется в страницу выполнения полей информаций.

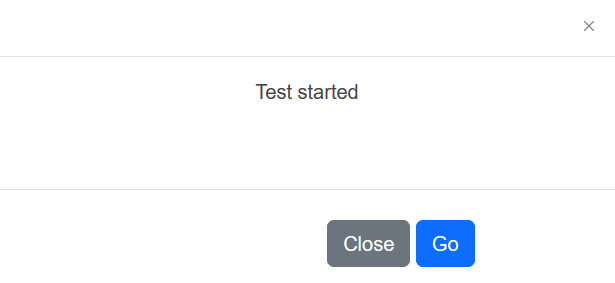


Рисунок 2 – Форма проверяет желание выполнения экзамена студента

После нажатия на кнопку «Go», студент сразу участвует в выбранном экзамене. На начале слево страницы есть все информации о экзамене и студенте, важным и необходимым является наличие таймера обратного отсчета. (см. на рисунку 3) . Список вопросов этого экзамене тоже появляется. Существуют два типа вопросов: вопрос с множественными выборами и вопрос с бланком (см. на рисунку 4 и 5).

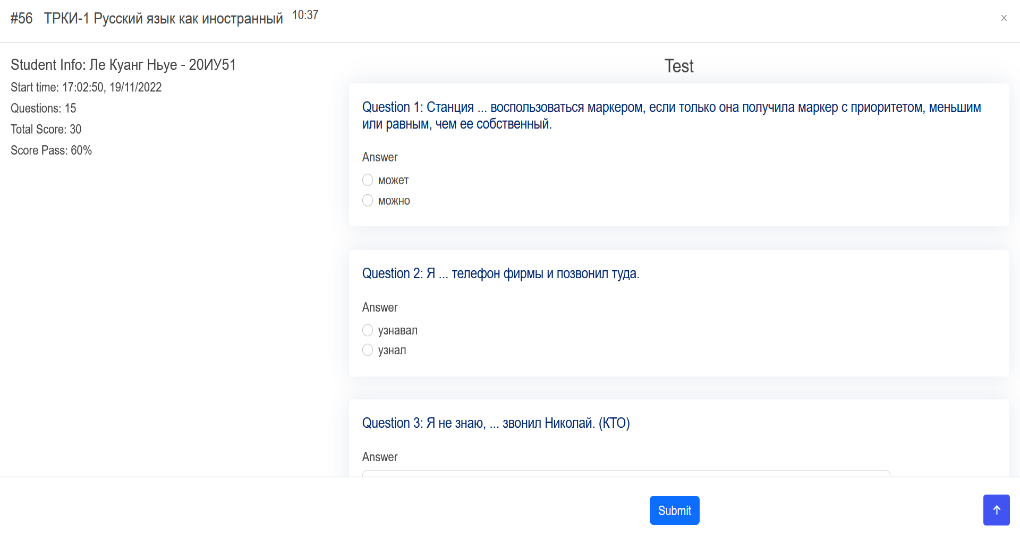


Рисунок 3 – Страница процесса выполнения экзамена для студента

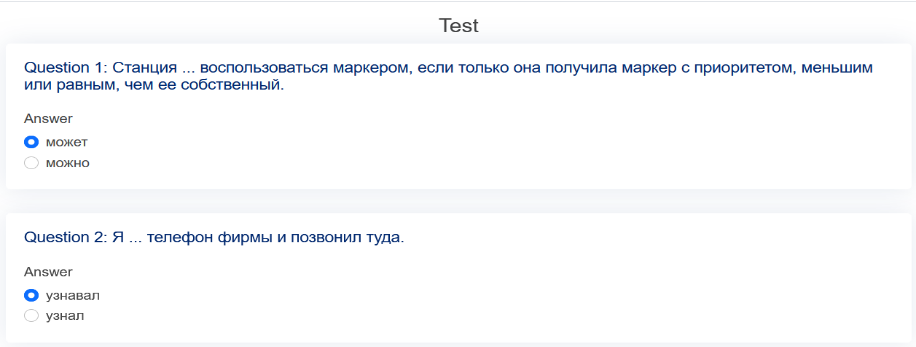


Рисунок 4 – Вопросы с множественными выборами

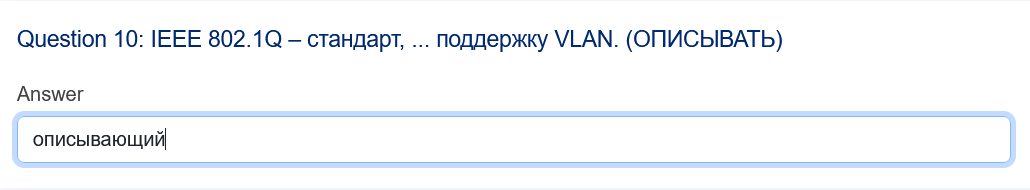


Рисунок 5 – Вопрос с бланком

Сделав все вопросы в экзамене, студент нажимает на кнопку «Submit» и сразу получает финальный результат этого экзамене (см. на рисунку 6). В случае, если время истечет (таймер обратного отсчета указывает 0:00), система автоматически оценит выполненные вопросы и предоставит окончательный результат студенту. Следует отметить, что если студент самовольно закроет браузер во время теста, он сразу же провалится на экзамене.

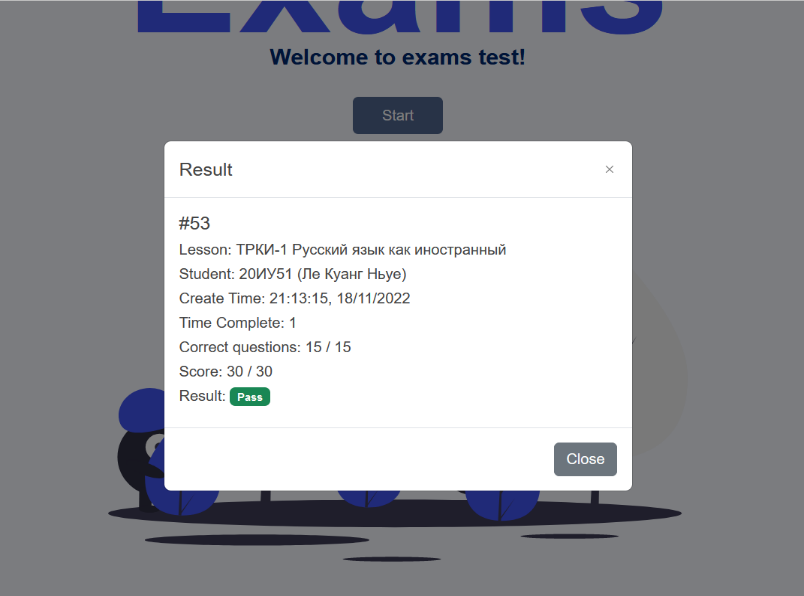
****

Рисунок 6 – Страница смотра финального результата экзамена для студента

1. **Руководство для преподавателя**

Для преподавателя, это единственный пользователь с наивысшим уровнем полномочий в системе. Для возможности работы с системой, пользователю необходимо авторизоваться в системе (см. на рисунку 7).

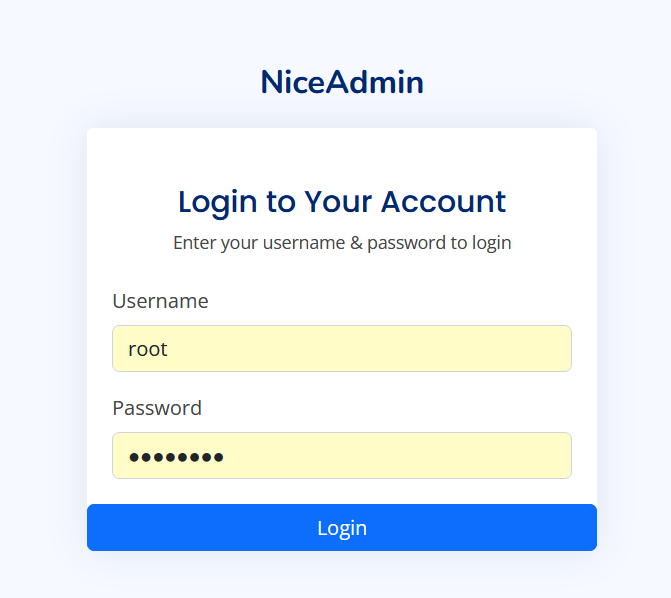


Рисунок 7 – Страница авторизации

После успешного входа в систему вы будете перенаправлены на главную страницу, показанную на рисунке 8.



Рисунок 8 – Главная страница после успешного входа в систему

Если преподаватель хочет построить один экзамен, то надо выполнить 3 следующих шага: добавить уровень вопроса, добавить предмет, и добавить вопросы в банк вопросов. Система поможет преподавателю связать вышеперечисленные параметры и составить произвольные экзаменационные вопросы для каждого студента. Кроме того, система также поможет преподавателю оценивать экзамены и сохранять все данные о экзамене каждого студента.

При добавлении уровней вопросов, преподаватель уснановит следующие параметры: имя уровня, количество баллов для одного вопроса этого уровня, время для одного вопроса этого уровня (см. рисунки 9 – 10).



Рисунок 9 – Подстранца уровней вопросов

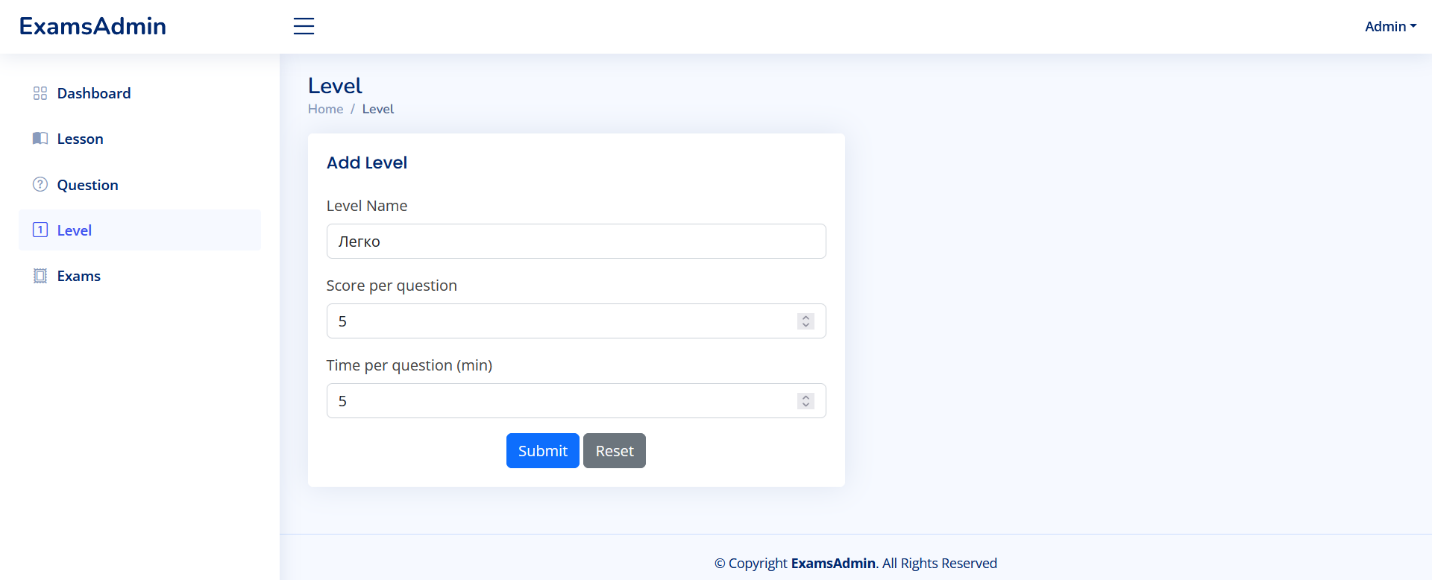


Рисунок 10 – Добавление и установка параметров какого-то уровня

После создания уровней вопросов, преподаватель начинает создать предмет. Преподаватель уснановит следующие параметры: имя предмета, количество процентов баллов успешного прохождения экзамена, и матрицу вопросов для экзамена этого предмета (см. рисунки 11 – 12).

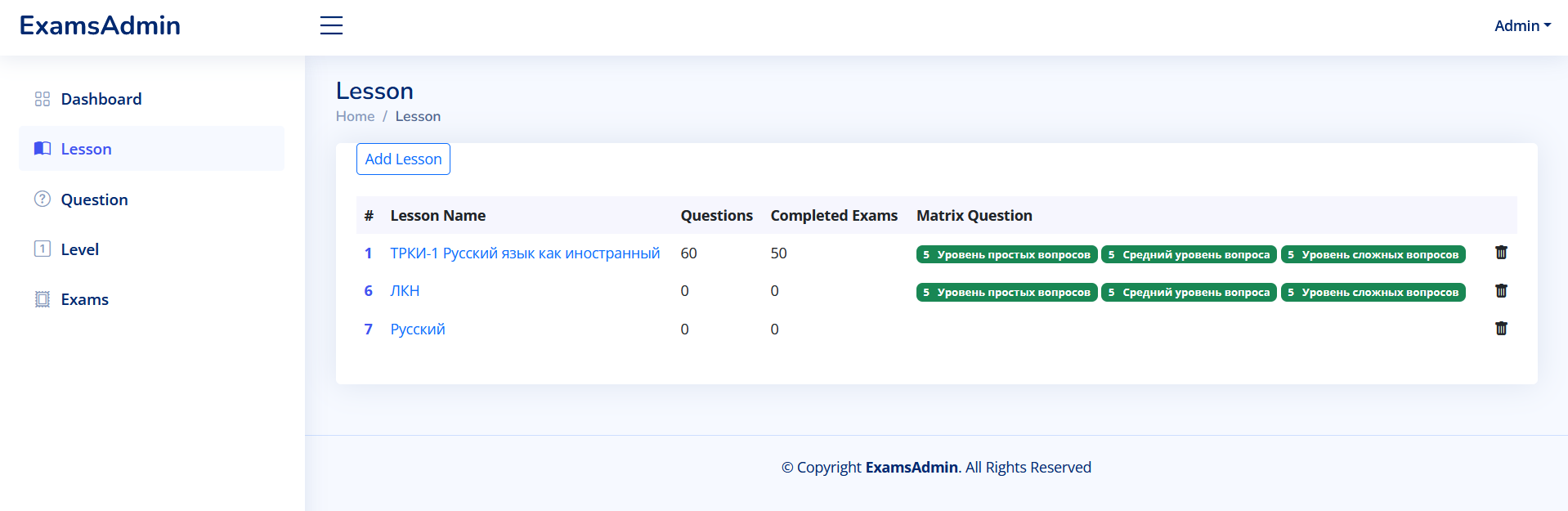
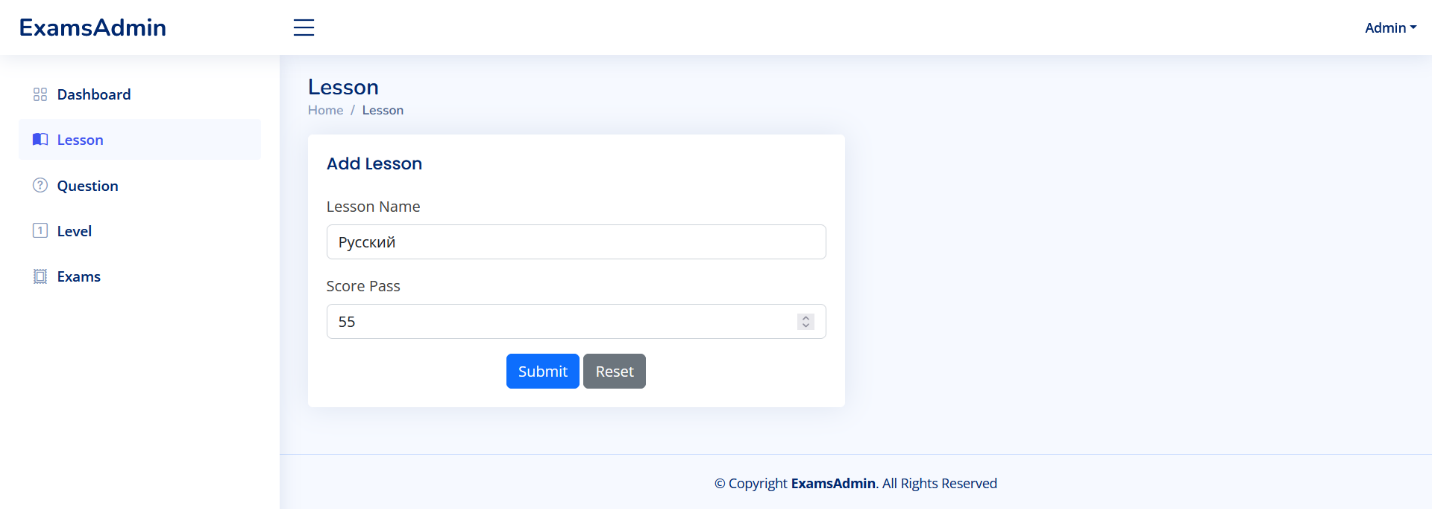


Рисунок 11 – Подстранца уровней вопросов



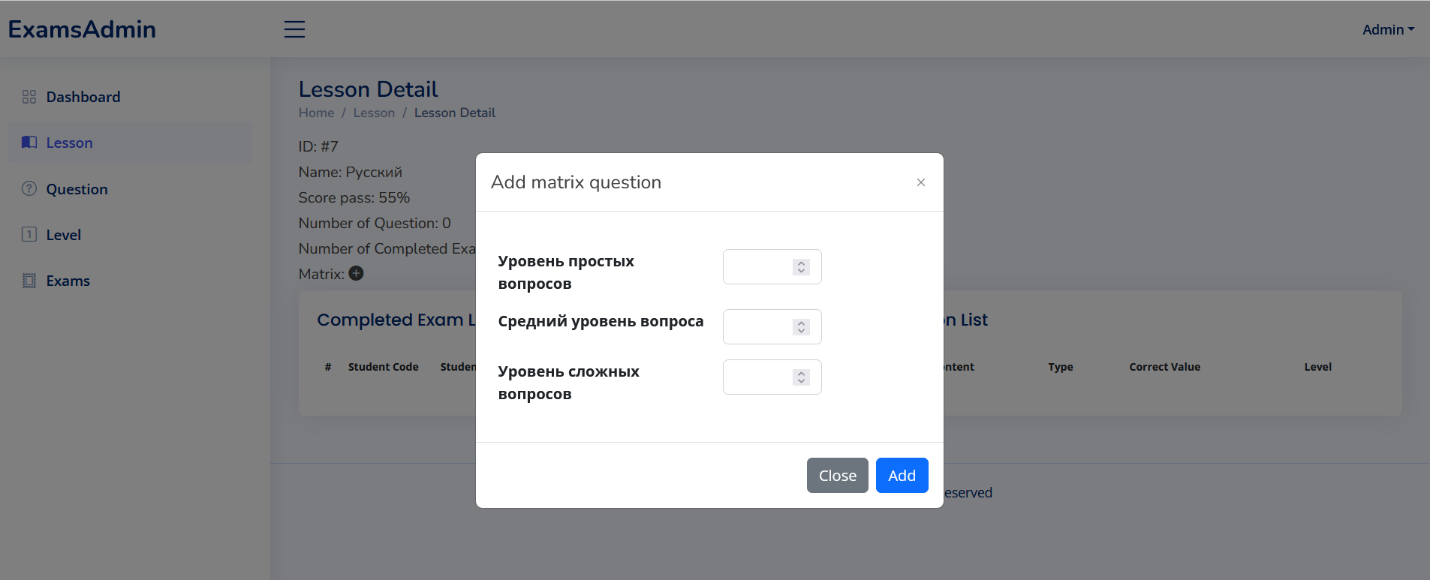


Рисунок 12 – Добавление и установка параметров какого-то предмета

Третьй шаг – это добавление вопроса в банк вопросов. При этом преподаватель уснановит следующие параметры: имя предмета, содержание вопроса, уровень вопроса, тип вопрос (если это вопрос с множественными выборами, то надо добавить выборы и соответствующий правильный выбор; а если это вопрос с бланком, то надо добавить правильные слова) (см. рисунки 13 – 14).

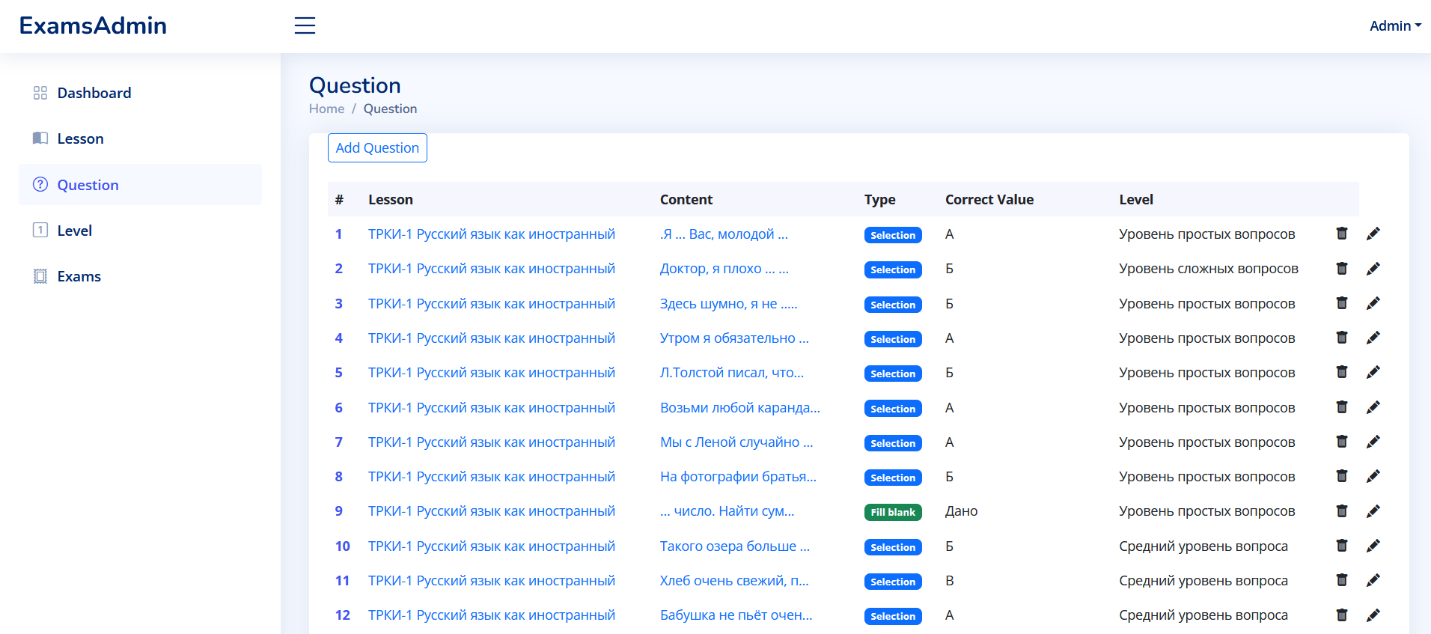


Рисунок 13 – Подстранца банка вопросов

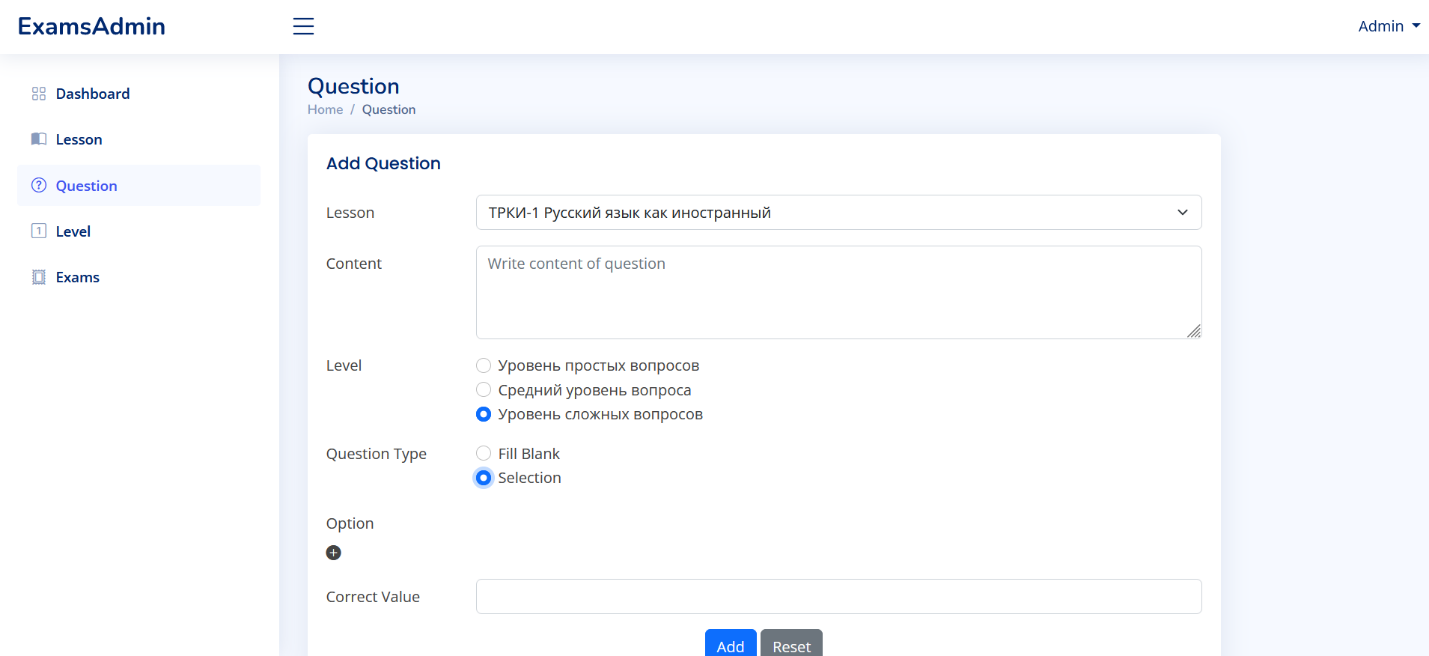


Рисунок 14 – Добавление и установка параметров какого-то вопроса

Поскольку это пользователь с наивысшим уровнем доступа, очевидно, что преподаватель имеет право видеть подробности экзаменов каждого студента (см. рисунки 15 – 16).

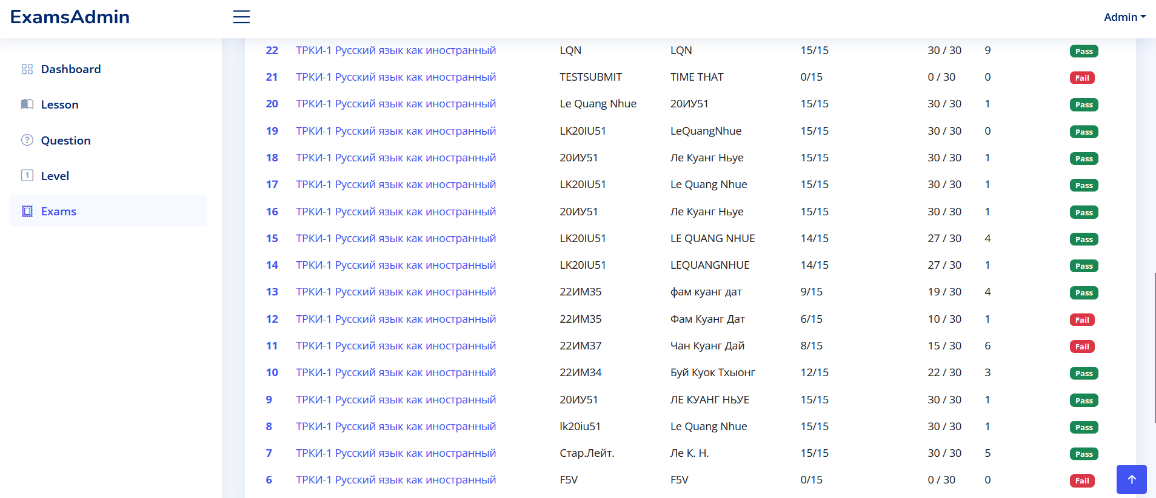


Рисунок 15 – Подстранца банка выполненных экзаменов

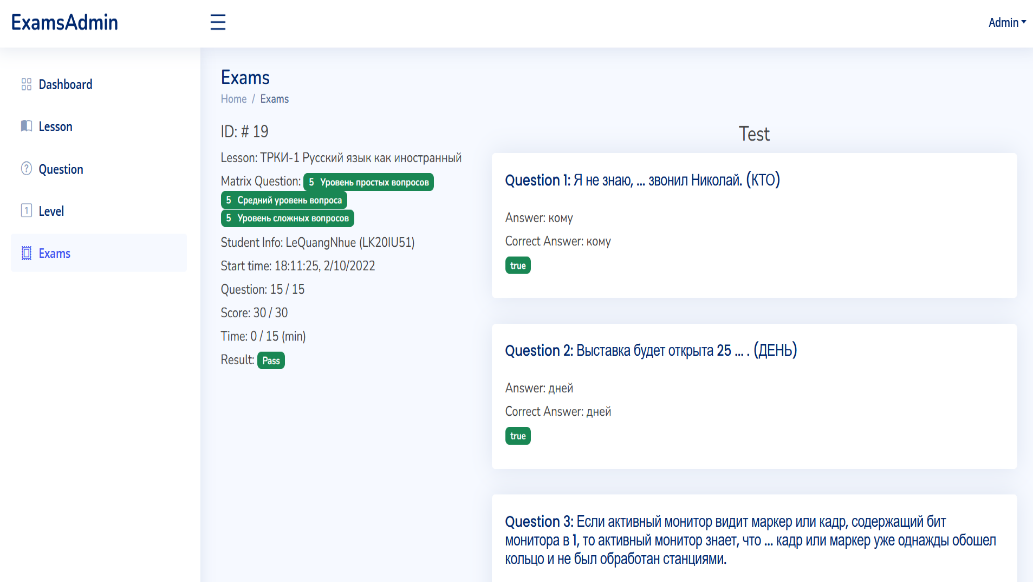


Рисунок 16 – Посмотреть детали одного экзамена какого-то студента

Помимо вышеперечисленных прав, преподаватель также имеет следующие права: удалять предмет; изменить содержимое вопроса или удалить вопрос (см. рисунки 17-19).

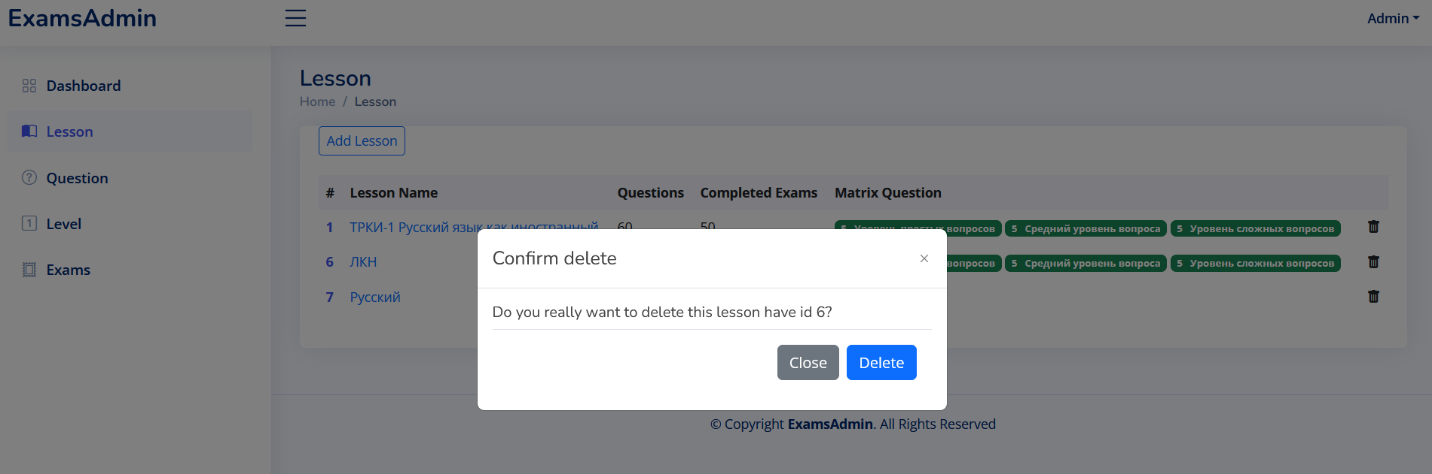


Рисунок 17 – Удаление предмета

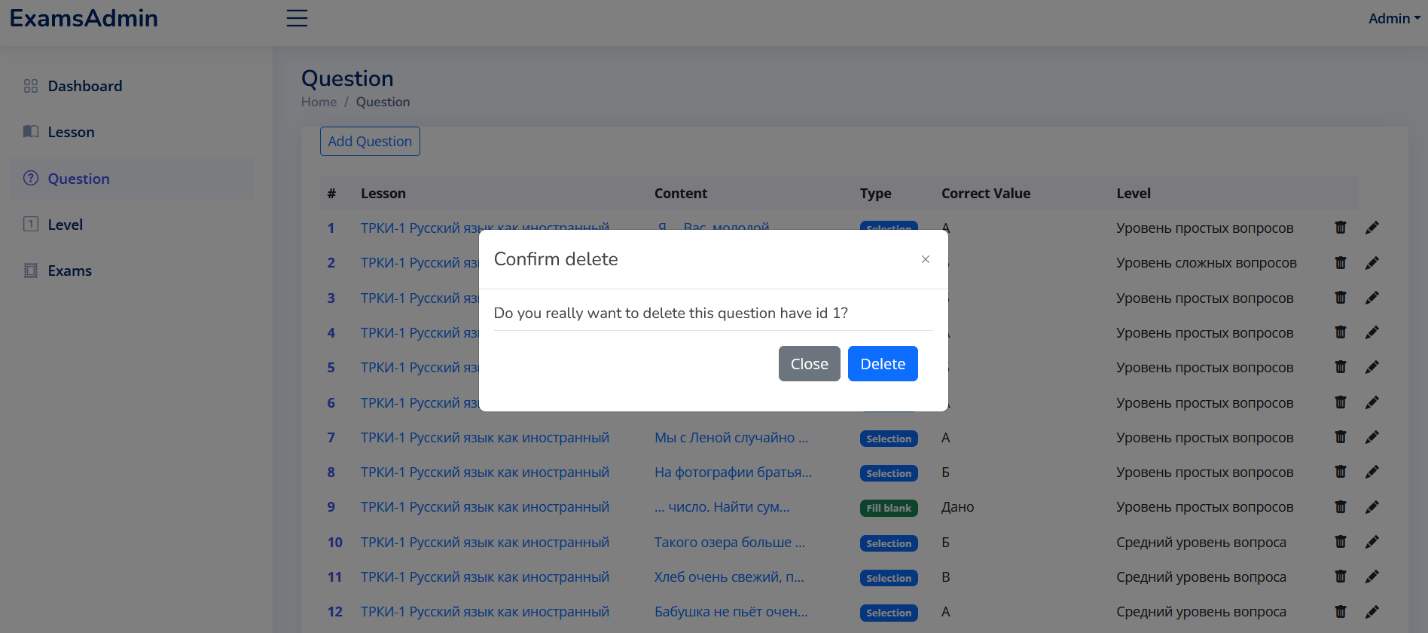


Рисунок 18 – Удаление вопроса

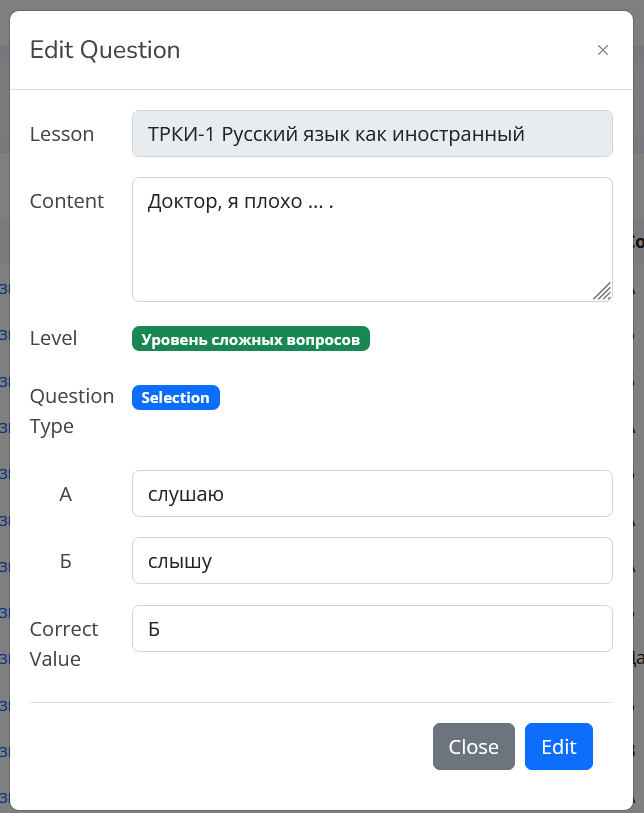


Рисунок 19 – Изменение содержимого вопроса

# **ПРИЛОЖЕНИЯ В**

**Листинг программы**Листов 2

Cсылка на местоположение полного комплекта исходных модулей программы: [**https://github.com/lequangnhueNdx86/EXAM.git**](https://github.com/lequangnhueNdx86/EXAM.git)

Некоторые тексты кодов (в файле ExamService.java):

**public** ExamGenerateResponse generateExam(ExamRequest examRequest) {

Exam exam = mapper.map(examRequest, Exam.**class**);

Lesson lesson = lessonRepository.findByIdAndRemovedFalse(examRequest.getLesson()).get();

// generate exam

// totalQuestion

Integer totalQuestion = getTotalQuestion(lesson);

// totalTime

Integer timeExam = getTotalTimeExam(lesson);

// total score

Integer totalScore = getTotalScore(lesson);

// random question & set answer false

List<Answer> listAnswer = randomQuestionAndSetAnswerFalse(lesson, exam);

exam.setTotalQuestion(totalQuestion);

exam.setCorrectQuestion(0);

exam.setTimeExam(timeExam);

exam.setTimeComplete(0.0);

exam.setTotalScore(totalScore);

exam.setScore(0);

exam.setResult(Constants.***FAIL***);

exam.setLesson(lesson);

exam.setListAnswer(listAnswer);

exam = examRepository.saveAndFlush(exam);

**return** mapExamToExamGenerateResponse(exam);

}

**public** ListExamGetResponse getAllExam() {

List<Exam> listExam = examRepository.findByOrderByCreateTimeDesc();

List<ExamGetResponse> list = **new** ArrayList<>();

listExam.forEach(exam -> {

**if** (exam.getLesson().getRemoved() == **false**) {

list.add(mapExamToExamGetResponse(exam, mapLessonToLessonGetResponse(exam.getLesson())));

}

});

**return** **new** ListExamGetResponse(list, Integer.*valueOf*(list.size()), **new** LessonGetResponse());

}

**public** ListExamGetResponse getAllExamByLesson(Integer lessonId) {

// get list exam by lesson id from db

List<Exam> listExam = examRepository.findAllByLessonId(lessonId);

// get lesson from db

Lesson lesson = lessonRepository.findByIdAndRemovedFalse(lessonId).get();

// map lesson -> lesson get response

LessonGetResponse lessonGetResponse = mapLessonToLessonGetResponse(lesson);

List<ExamGetResponse> listExamGetResponses = **new** ArrayList<>();

// each exam, map exam -> exam get response

listExam.forEach(exam -> {

listExamGetResponses.add(mapExamToExamGetResponse(exam, lessonGetResponse));

});

**return** **new** ListExamGetResponse(listExamGetResponses, listExamGetResponses.size(), lessonGetResponse);

}

**public** ExamDetailGetResponse getExamById(Integer examId) {

Exam exam = examRepository.findById(examId).get();

// map exam -> exam detail by name & type

ExamDetailGetResponse examDetailGetResponse = mapper.map(exam, ExamDetailGetResponse.**class**);

examDetailGetResponse.setLesson(mapLessonToLessonGetResponse(exam.getLesson()));

List<Answer> listAnswer = exam.getListAnswer();

List<AnswerGetResponse> answerGetResponses = **new** ArrayList<>();

listAnswer.forEach(answer -> {

answerGetResponses.add(mapAnswerToAnswerGetResponse(answer));

});

examDetailGetResponse.setListAnswer(answerGetResponses);

**return** examDetailGetResponse;

}