**Поволжский Государственный Университет Телекоммуникаций и Информатики**Кафедра «Программная инженерия»

|  |  |
| --- | --- |
| Сдана на проверку  «\_\_» \_\_\_\_2023 г. | Допустить к защите  «\_\_» \_\_\_\_\_ 2023 г.  Защищена с оценкой  «\_\_\_» \_\_\_\_\_\_\_\_ 2023 г. |

**КУРСОВАЯ РАБОТА**По дисциплине: «Прикладное программирование»  
На тему: «Сайт для обучения языку»  
  
Пояснительная записка

|  |  |
| --- | --- |
| Студент группы ИВТ-11\_\_\_\_\_\_\_ Хохлов М. С.  (роспись) (ФИО)  Руководитель \_\_\_\_\_\_\_\_ к.т.н., доц. Ахметшина Э. Г.  (роспись) (ФИО) | 0 200588 0  (№ зачетной книжки) |

Самара 2023

# Рецензия

# **Содержание**

[Описание предметной области. Актуальность 4](#_Toc3)

[Описание программы. Общие сведения 5](#_Toc4)

[Функциональное назначение 5](#_Toc5)

[Логическая модель базы данных 7](#_Toc6)

[Физическая модель базы данных 8](#_Toc7)

[Диаграмма классов 9](#_Toc8)

[Диаграмма компонентов 10](#_Toc9)

[Диаграмма вариантов использования 11](#_Toc10)

[Диаграмма последовательности 12](#_Toc11)

[Демонстрация работы приложения 13](#_Toc12)

[Список используемых источников 24](#_Toc13)

[Приложение А – Листинг программного кода 25](#_Toc14)

# Описание предметной области. Актуальность

В условиях современного мира, где общение и образование переходят в виртуальное пространство, проект разработки сайта для поиска и проведения занятий по языкам приобретает особую актуальность. Эффективное владение иностранными языками становится ключевым элементом успешной социальной и профессиональной адаптации.

**Актуальность предложенной темы:**

1. потребность в языковом обучении: растущий интерес к языковому обучению обусловлен глобальными тенденциями. Наш сайт предоставляет платформу для эффективного поиска и проведения занятий, обеспечивая обмен знаниями и опытом в области языкового образования.
2. онлайн-образование: современные технологии открывают новые возможности для обучения в сети. Проект адаптируется под актуальные требования онлайн-образования, обеспечивая доступность и удобство проведения занятий в любом месте и в любое время.
3. сообщество преподавателей и студентов: создание виртуального пространства для взаимодействия преподавателей и студентов удовлетворяет потребность в гибкости и выборе. Проект способствует формированию образовательного сообщества, где каждый может найти оптимальные условия для проведения занятий.

Резюмируя, актуальность проекта заключается в его способности соответствовать современным тенденциям в образовании и обеспечивать эффективные инструменты для поиска и проведения занятий по языкам в онлайн-формате. Платформа предоставляет ресурсы для обмена знаниями и опытом, поддерживая учебный процесс в динамичной образовательной среде.

# Описание программы. Общие сведения

Приложение – веб-ресурс. Установка не требуется.

Язык программирования: JavaScript, Java.

Среда разработки: Intellij IDEA, VS Code.

Фреймворк для Backend: Spring Boot

Библиотека для Frontend: React.js

Объем проекта: 193,3 МБ МБ (вместе с исходным кодом).

Исходный код (Java и JavaScript): 1493 строк.

# Функциональное назначение

Веб-ресурс представляет собой веб-сайт, разработанный для обеспечения базовых функций управления данными по обучению языку. Проект ориентирован на простоту и эффективность использования, фокусируясь на реализации CRUD операций для четырех ключевых сущностей: Преподаватель, Тип Занятия, Занятие и Язык.

**Функциональные Возможности:**

* Преподаватель:
  + создание: добавление новых преподавателей с указанием их стажа и контактной информации
  + чтение: просмотр списка всех преподавателей с подробной информацией о каждом
  + обновление: внесение изменений в данные о преподавателях
  + удаление: удаление информации о преподавателях из базы данных.
* тип Занятия:
  + создание: определение новых типов занятий
  + чтение: просмотр списка всех типов занятий с их характеристиками
  + обновление: внесение изменений в существующие типы занятий
  + удаление: удаление типов занятий из базы данных
* Занятие:
  + создание: регистрация новых занятий с указанием даты, времени и связанных сущностей (преподаватель, тип занятия и язык)
  + чтение: просмотр списка всех занятий с деталями каждого занятия
  + обновление: возможность внесения изменений в информацию о занятиях
  + удаление: удаление занятий из базы данных
* Язык:
  + создание: добавление новых языков, доступных для изучения
  + чтение: просмотр списка всех языков на платформе
  + обновление: внесение изменений в данные о языках
  + удаление: удаление языков из базы данных

Функциональное наполнение сайта ориентировано на обеспечение максимального комфорта пользователей при поиске и выборе языковых занятий. Предоставляемые инструменты направлены на создание эффективной образовательной среды, где каждый участник может найти оптимальные условия для успешного обучения.

# Логическая модель базы данных

Сущность «Язык» (**language)**:

* id (bigint, primary key): уникальный идентификатор языка.
* name (varchar(50)): наименование языка.

Сущность «Тип занятия» (**lesson\_type)**:

* id (bigint, primary key): уникальный идентификатор типа урока.
* name (varchar(50)): наименование типа урока.

Сущность «Преподаватель» (**teacher)**:

* id (bigint, primary key): уникальный идентификатор преподавателя.
* experience (integer): опыт преподавателя.
* firstname (varchar(50)): имя преподавателя.
* lastname (varchar(50)): фамилия преподавателя.
* contacts (varchar(250)): контактные данные преподавателя.

Сущность «Занятие» (**lesson)**:

* duration (integer): продолжительность урока.
* timestamp (date): дата и время проведения урока.
* id (bigint, primary key): уникальный идентификатор урока.
* language\_id (bigint, foreign key): связывает урок с языком.
* lesson\_type\_id (bigint, foreign key): связывает урок с типом урока.
* teacher\_id (bigint, foreign key): связывает урок с преподавателем.
* lesson\_difficulty (varchar(20)): сложность урока, ограниченная значениями 'EASY', 'MIDDLE', 'HARD'.
* name (varchar(150)): наименование урока.
* description (varchar(255)): описание урока.

# Физическая модель базы данных

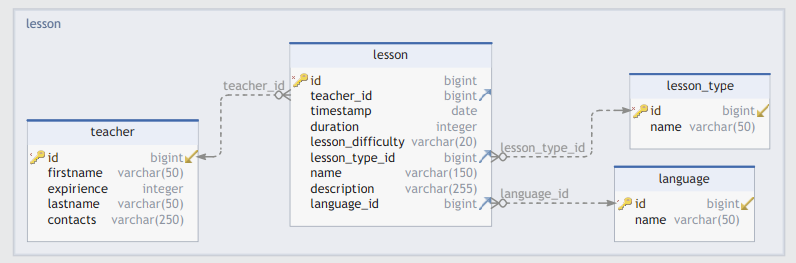


Рис. 1 – Физическая модель базы данных

# Диаграмма классов

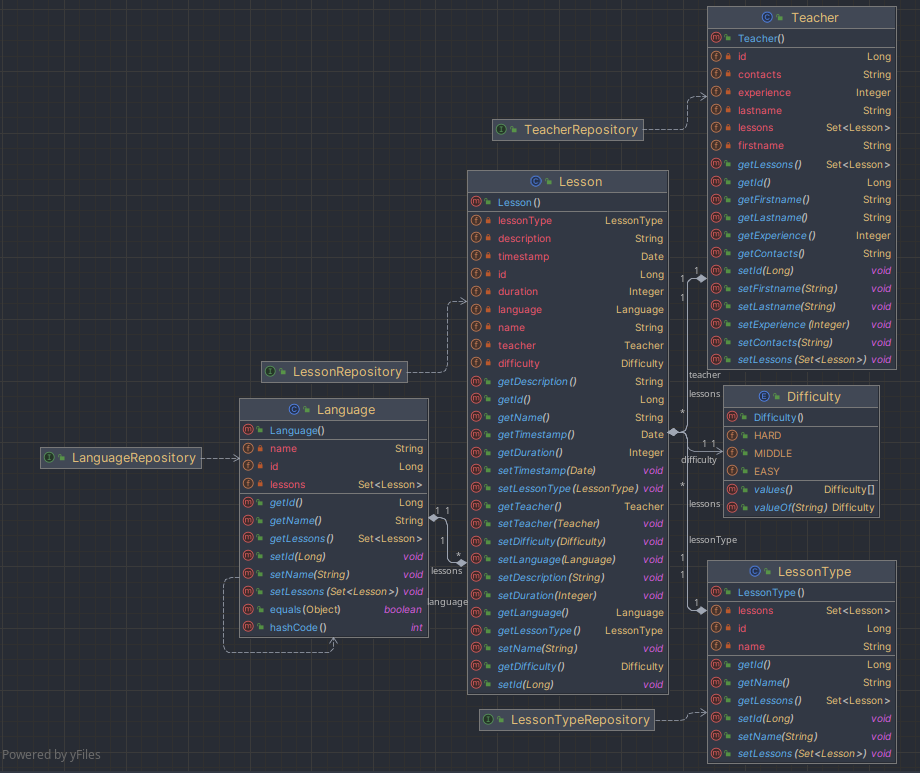


Рис. 2 – Диаграмма классов

# Диаграмма компонентов

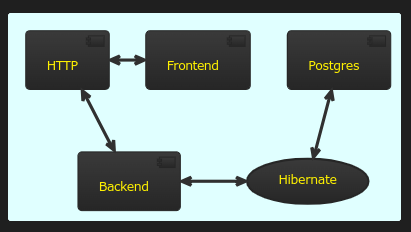


Рис. 3 – Диаграмма компонентов

**Компоненты:**

* Backend представляет серверный бэкенд системы.
* Frontend представляет клиентский интерфейс.
* Postgres обозначает базу данных PostgreSQL.
* Hibernate указывает на инструмент для взаимодействия с базой данных.

**Связи**:

* Связь между Frontend и HTTP отображает взаимодействие клиентского интерфейса с протоколом HTTP.
* Связь между HTTP и Backend показывает взаимодействие бэкенда с клиентским интерфейсом посредством HTTP.
* Связь между Backend и Hibernate отражает использование инструмента Hibernate для работы с базой данных.
* Связь между Hibernate и Postgres указывает на взаимодействие с базой данных PostgreSQL.

# Диаграмма вариантов использования

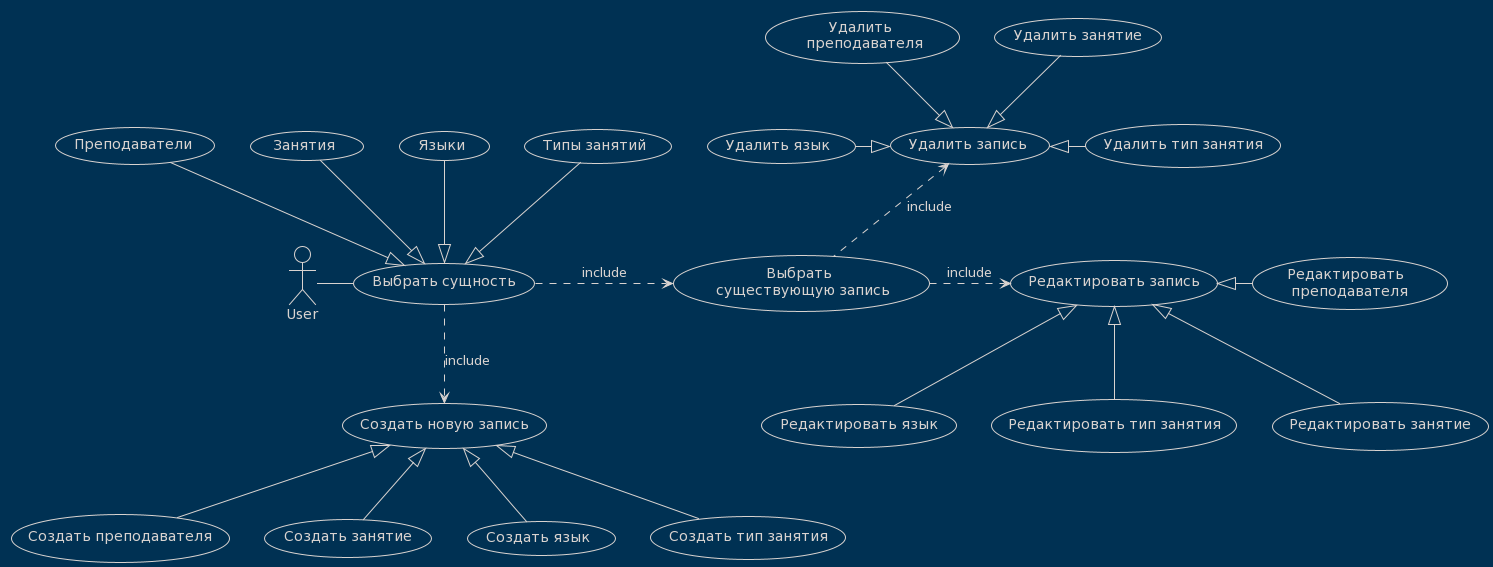


Рис. 4 – Диаграмма вариантов использования

# Диаграмма последовательности

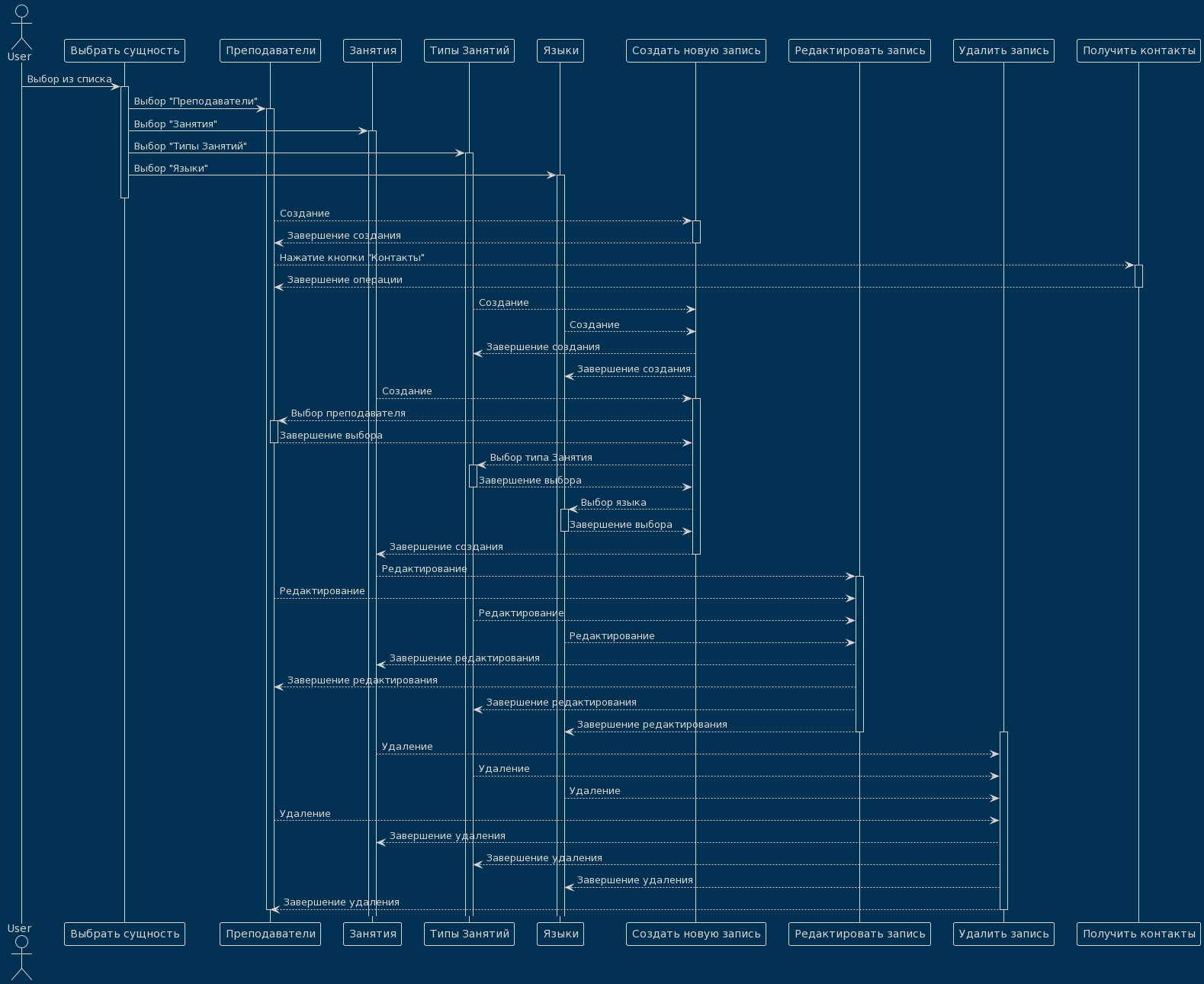


Рис. 5 – Диаграмма последовательности

# Демонстрация работы приложения

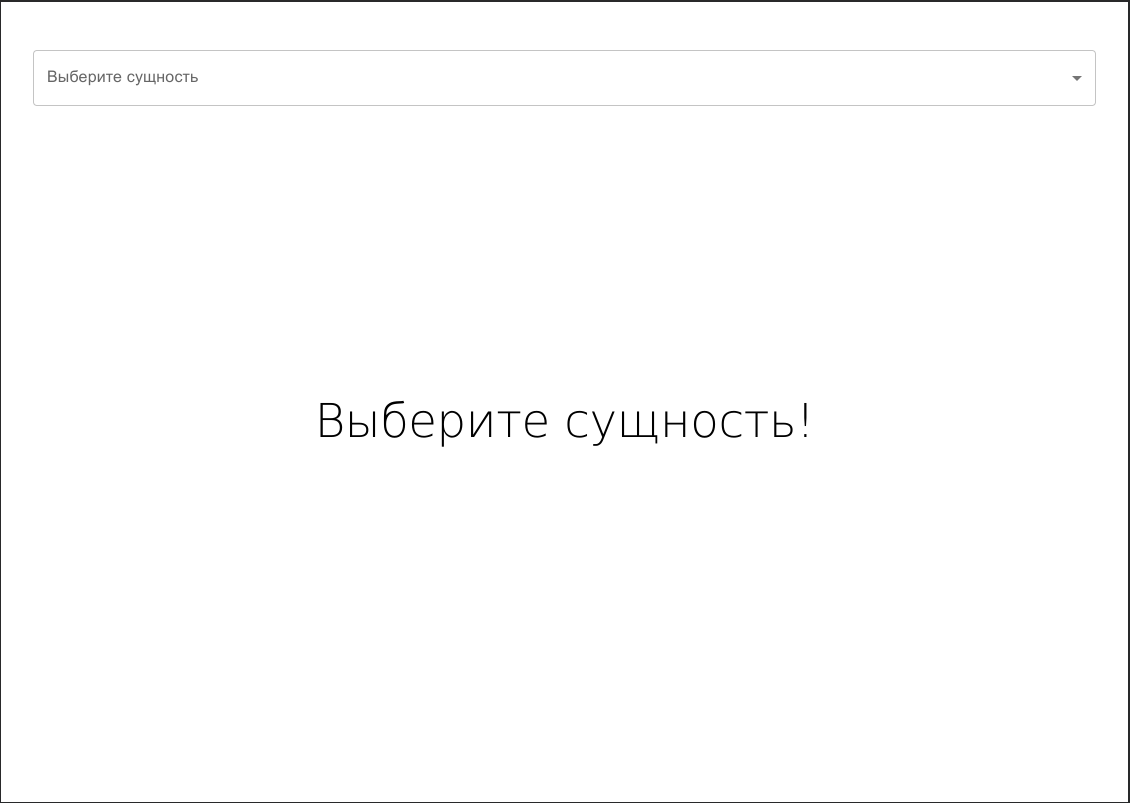


Рис. 6.1 – Главная страница приложения

При открытии сайта пользователя приветствует главная и единственная страница приложения. Первым шагом является выбор сущности.

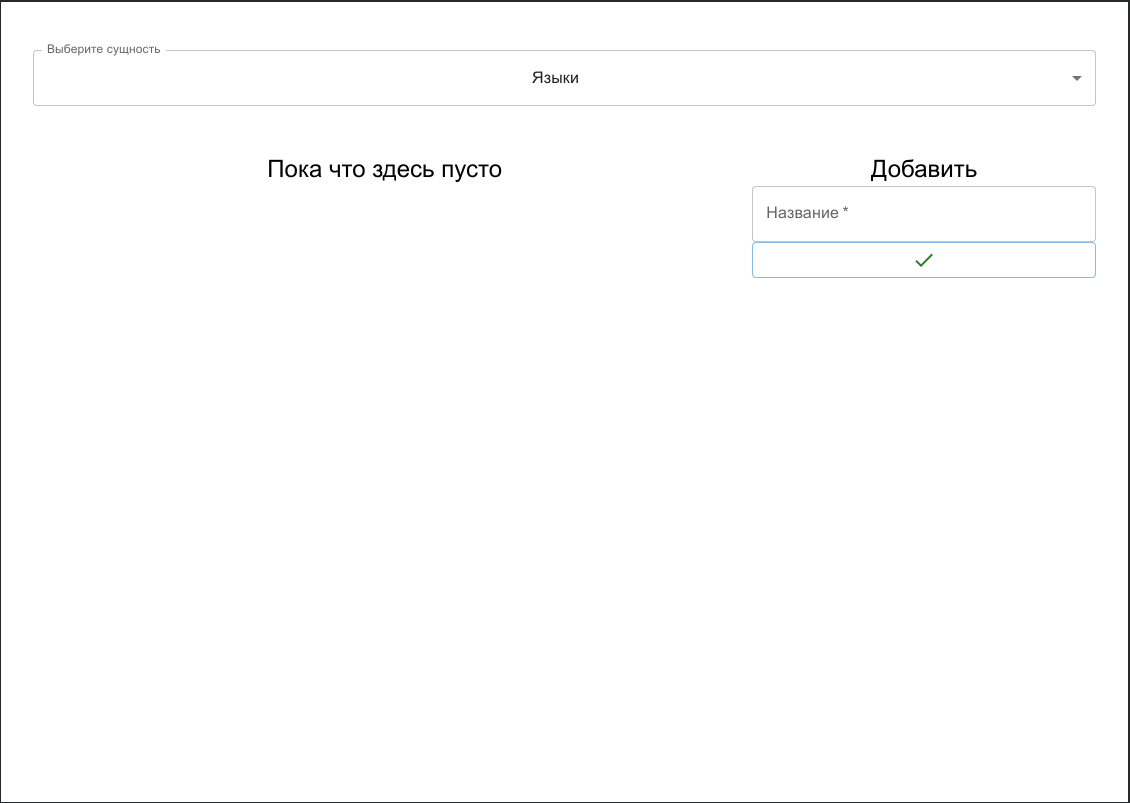


Рис. 6.2 – Выбор сущности «языки»

После выбора сущности посреди страницы появляется форма для создания новых данных, а слева от нее область, в которой отображаются созданные данные.

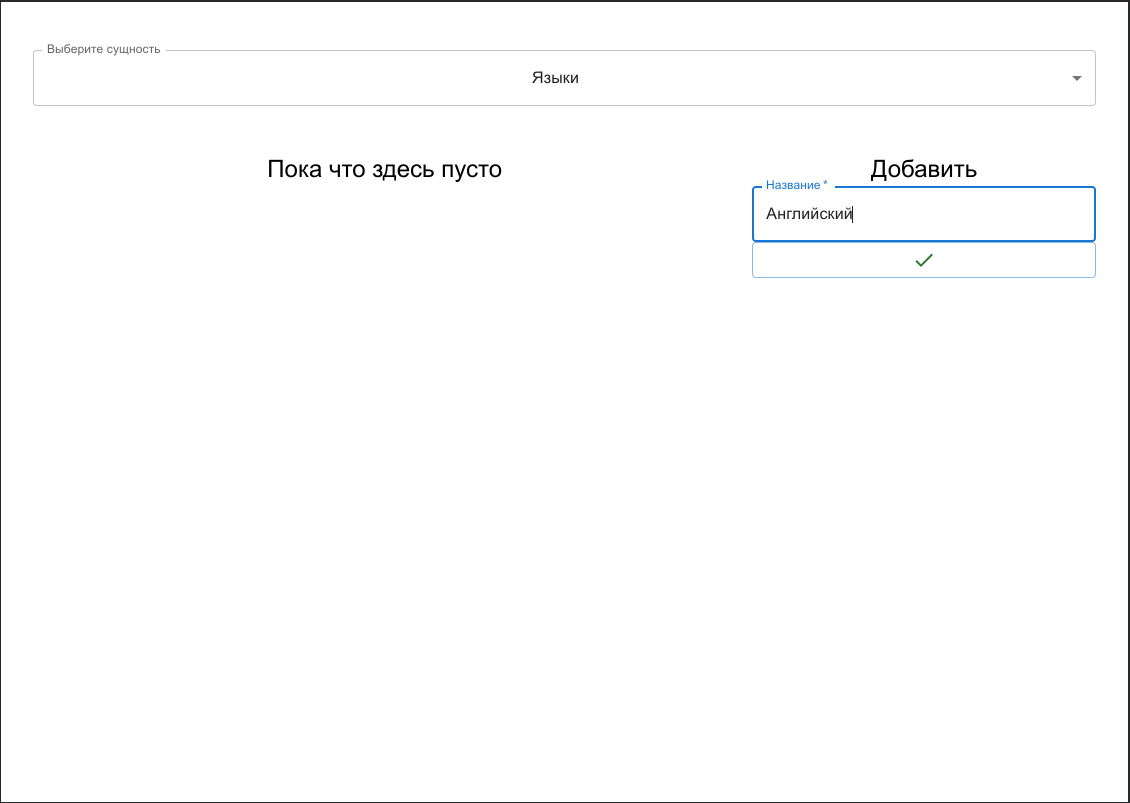


Рис. 6.3 – Ввод новых данных в форму

Для создания вводим данные в единственное после «Название».

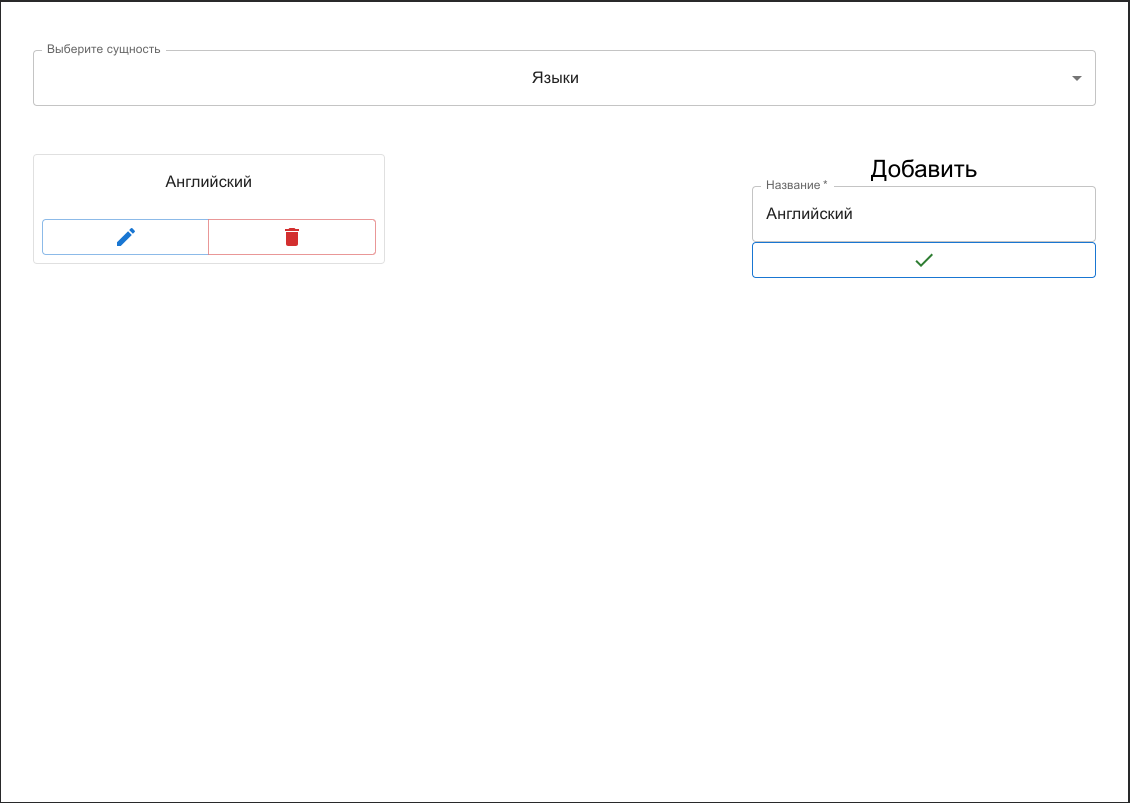


Рис. 6.4 – Последствия создания языка

После отправки формы на сервер мы можем убедиться, что у нас добавилась карточка языка. На ней располагаются 2 кнопки для удаления и редактирования данных.

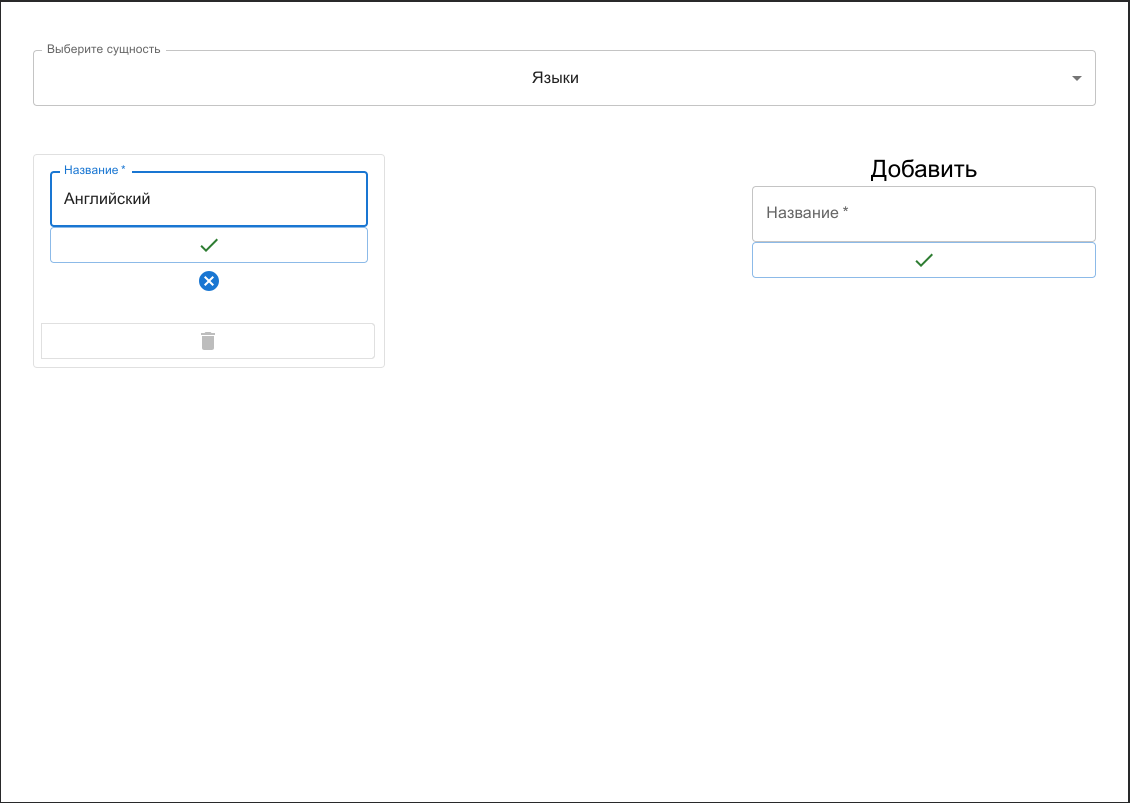


Рис. 6.5 – Редактирование названия языка

Нажатием на карандаш внутри карточки языка мы вызываем форму редактирования. В текстовое поле с названием мы введем «Французский».

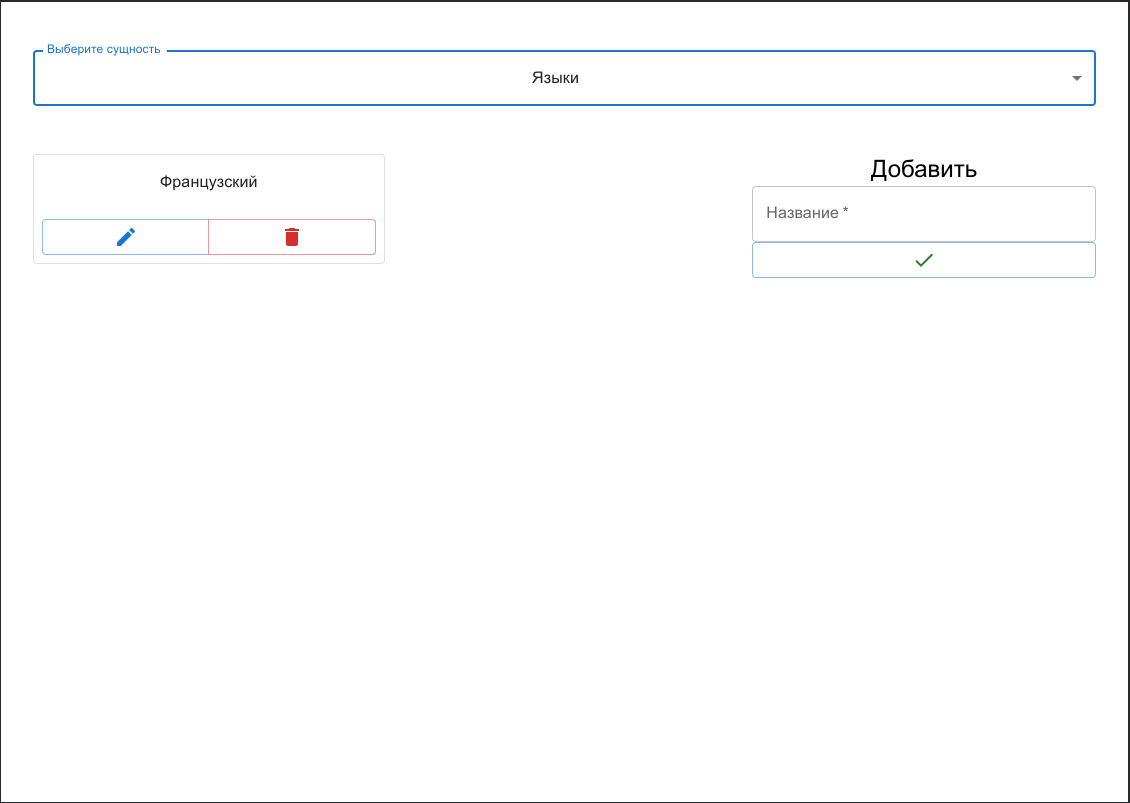


Рис. 6.6 – Последствия редактирования

Нажатием на зеленую кнопку с галочкой мы сохранили результат редактирования.

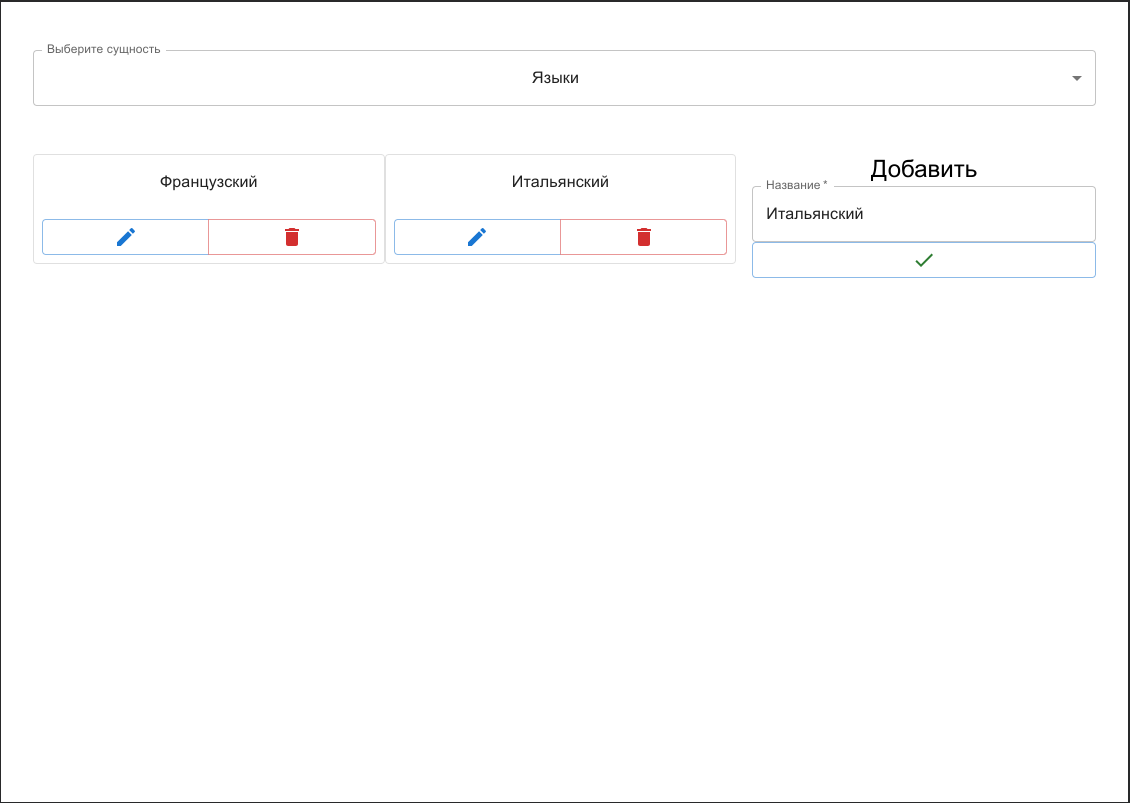


Рис. 6.7 – Результат добавления итальянского языка

Добавляем итальянский язык следуя тем шагам, что мы уже прошли выше.

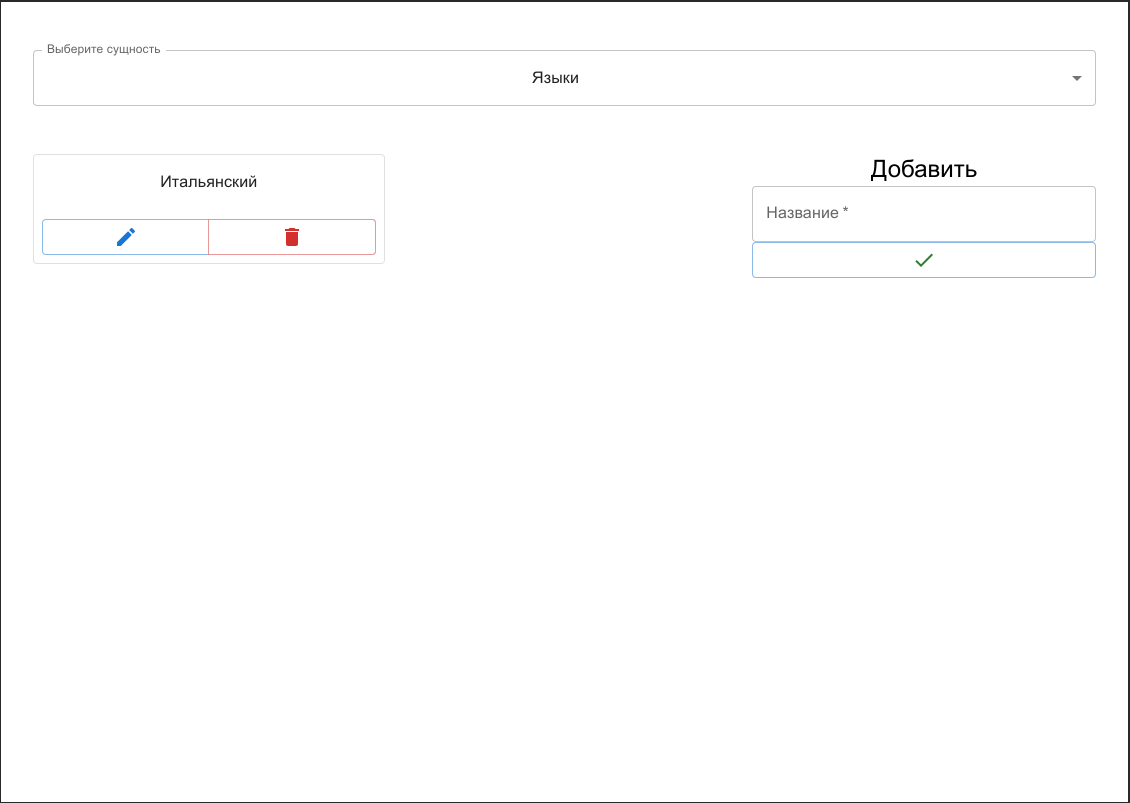


Рис. 6.8 – Удаление французского языка

Нажатием на кнопку с красным мусорным баком мы удалили данные о француском языке в системе.

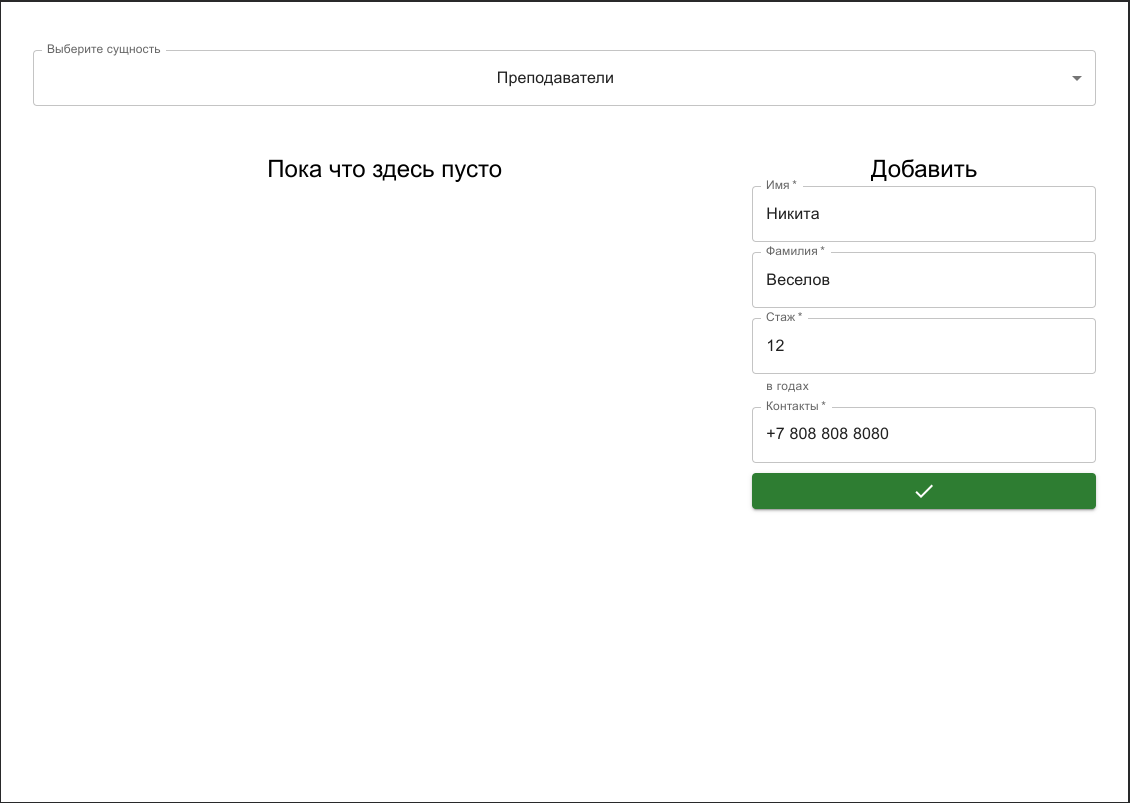


Рис. 6.9 – Создание преподавателя

Секция данных сущностей и форма имеют схожую структуру. На рис. 6.9 изображена заполненная форма с данными преподавателя.

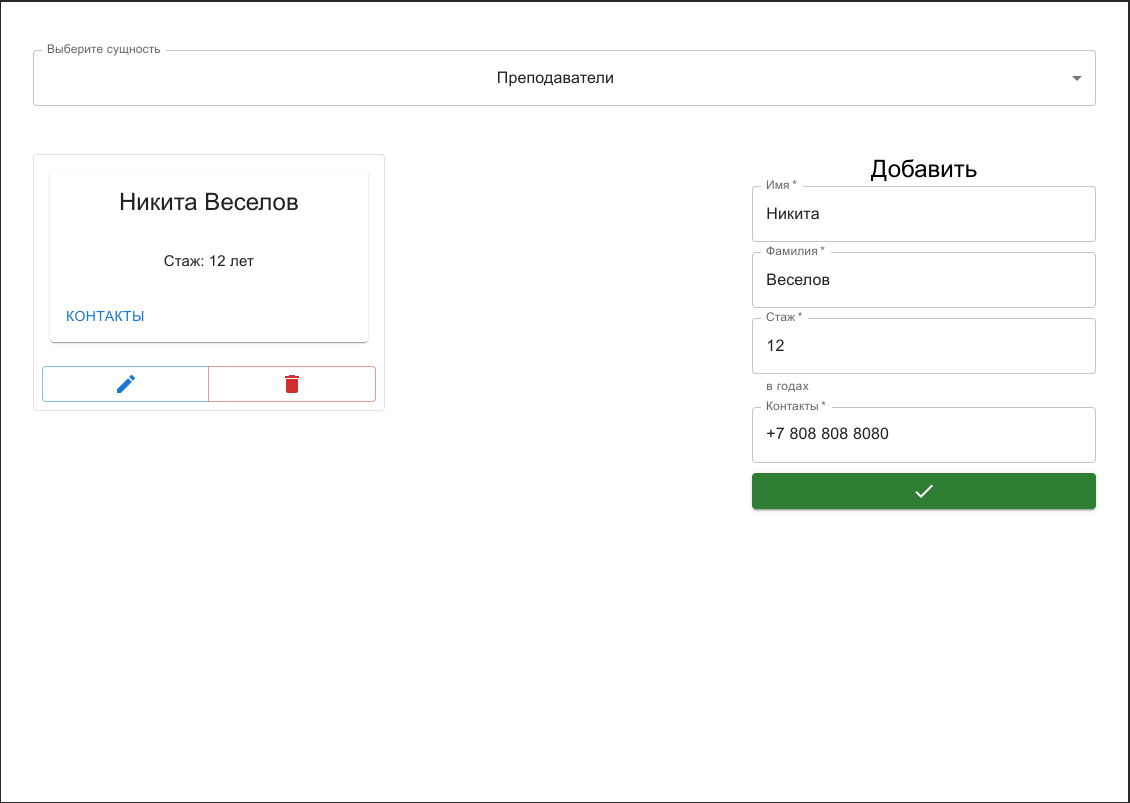


Рис. 6.10 – Результат создания преподавателя

После нажатия кнопки создания, на секции слева появляется карточка с преподавателем.

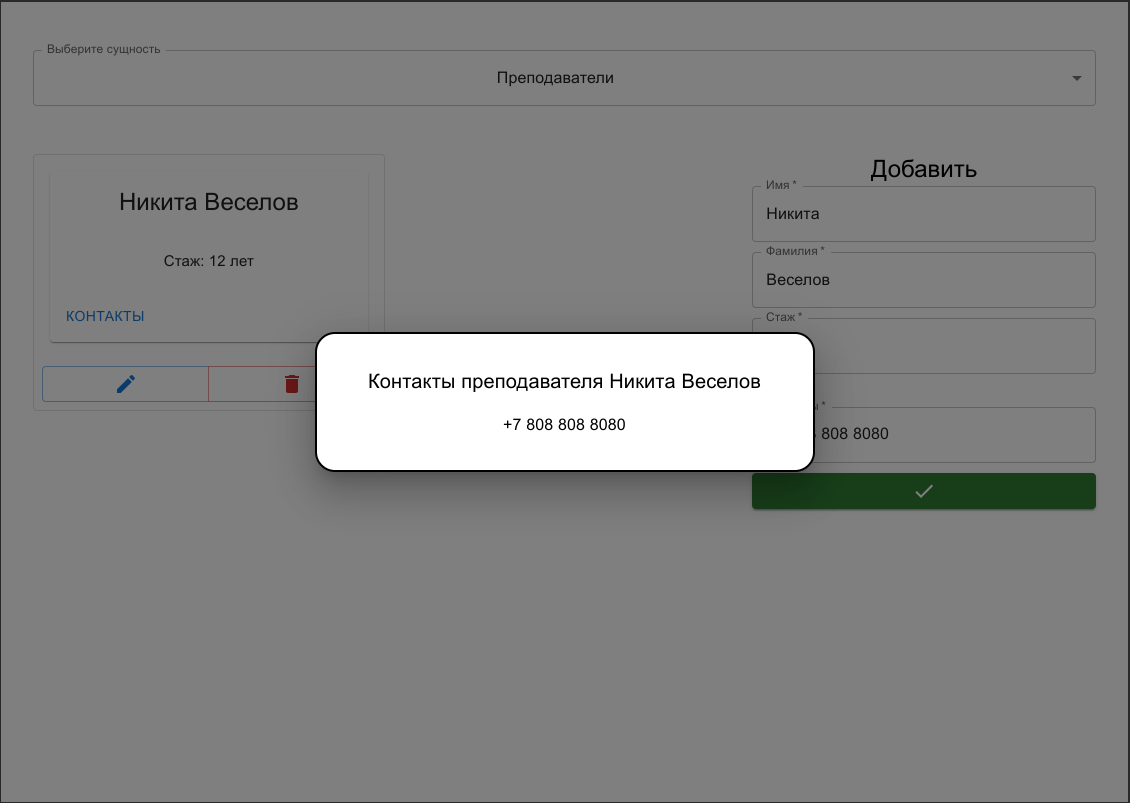


Рис. 6.11 – Просмотр контактов преподавателя

Нажатием на кнопку «Контакты», которая расположена в нижнем левом углу карточки, мы вызываем модальное окно, в котором располагаются нужные нам данные о номере телефона преподавателя.

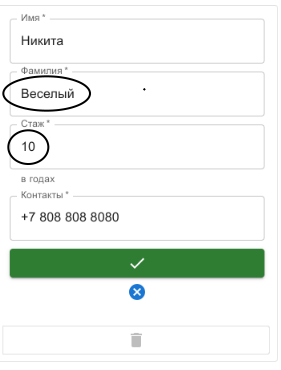


Рис. 6.12 – Редактирование данных преподавателя

Изменяем значения фамилии и стажа преподавателя.

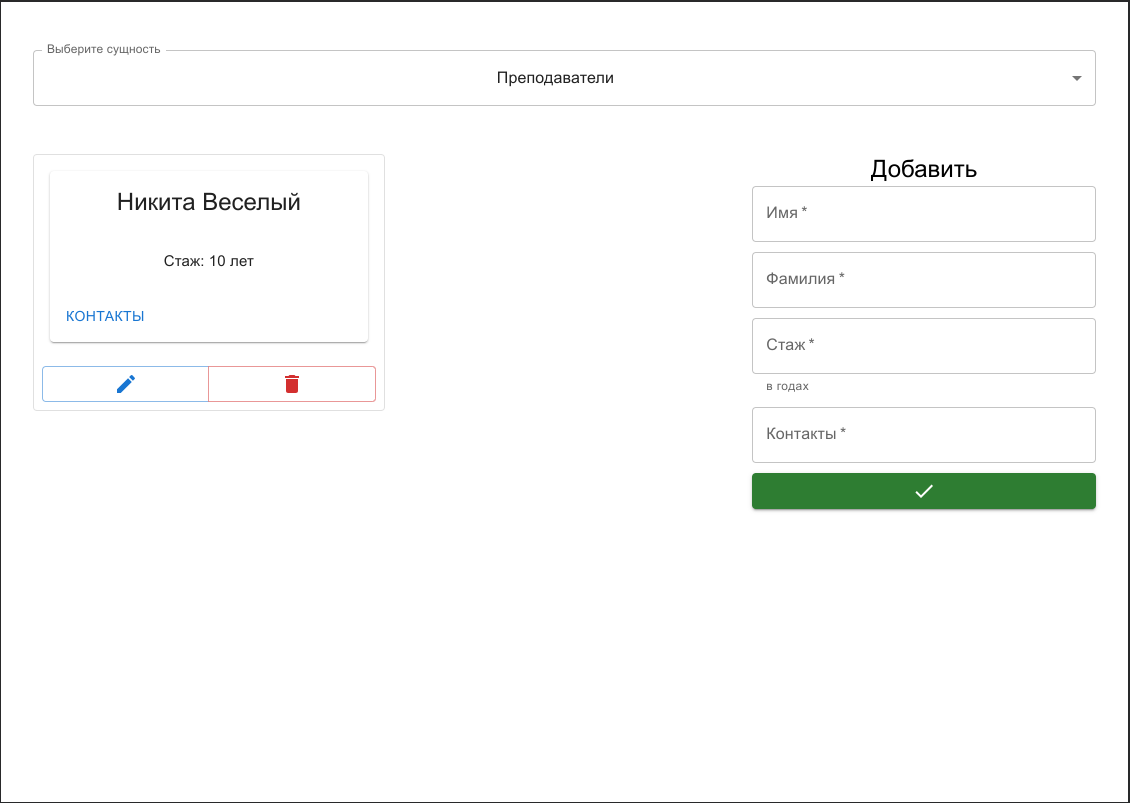


Рис. 6.13 – Обновленные данные преподавателя

После обновления у преподавателя изменился стаж работы и фамилия.

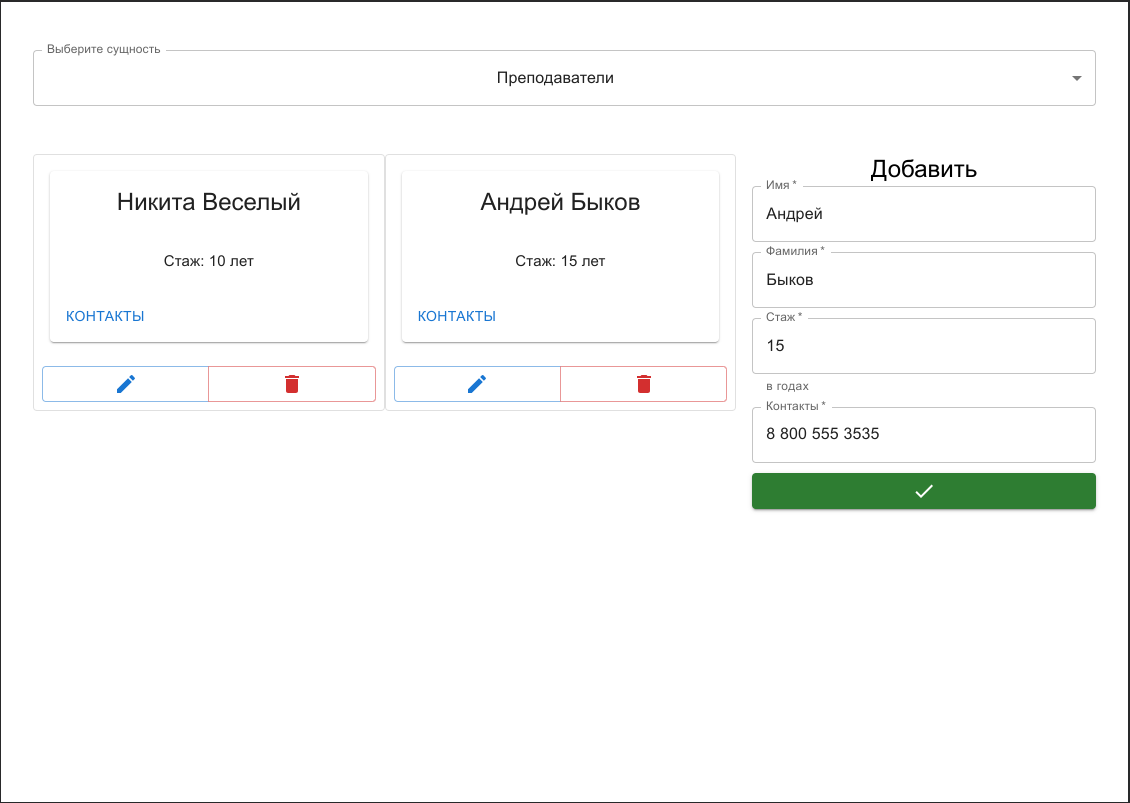


Рис. 6.14 – Создание нового преподавателя

Создали второго преподавателя. Теперь у нас 2 преподавателя в системе.

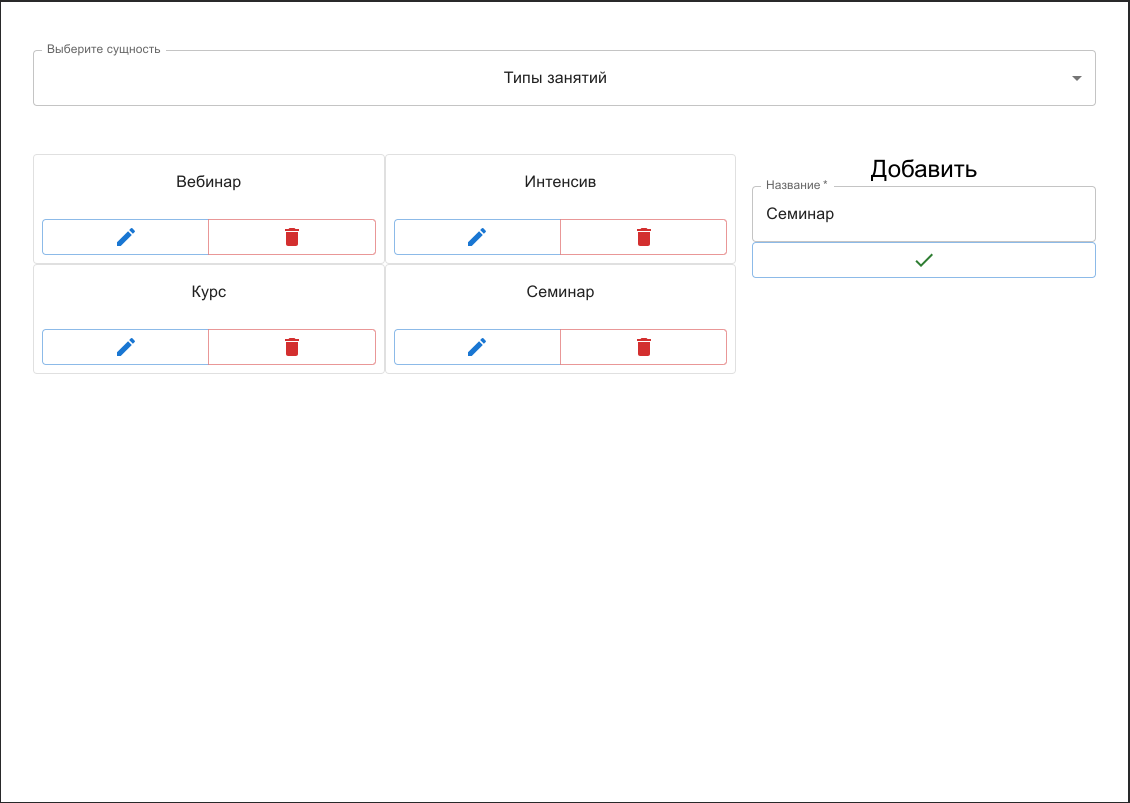


Рис. 6.15 – Создание типа занятия

Создание типа занятие схоже с созданием других сущностей, поэтому мы опускаем этапы создания каждого типа. На рис. 6.15 показаны результаты создания 4-х типов занятия.

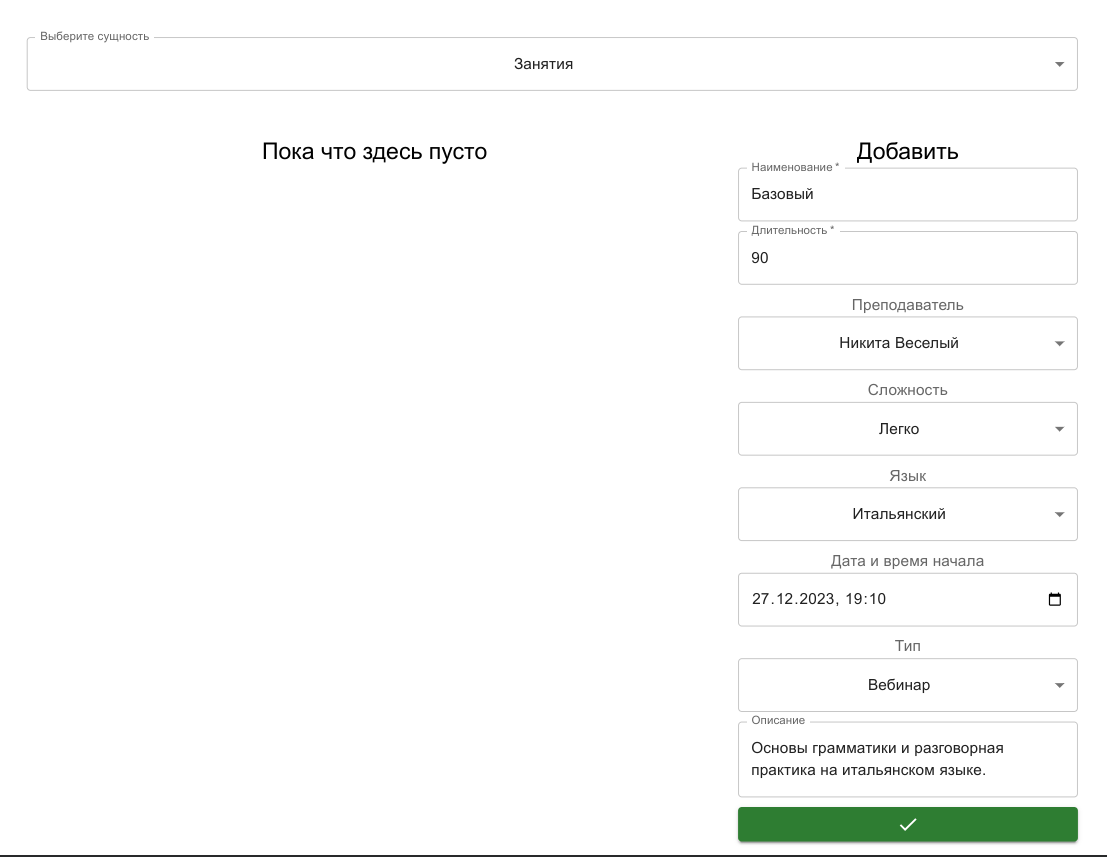


Рис. 6.16 – Заполнение формы

На рис. 6.16 изображен процесс заполнения формы создания занятия.

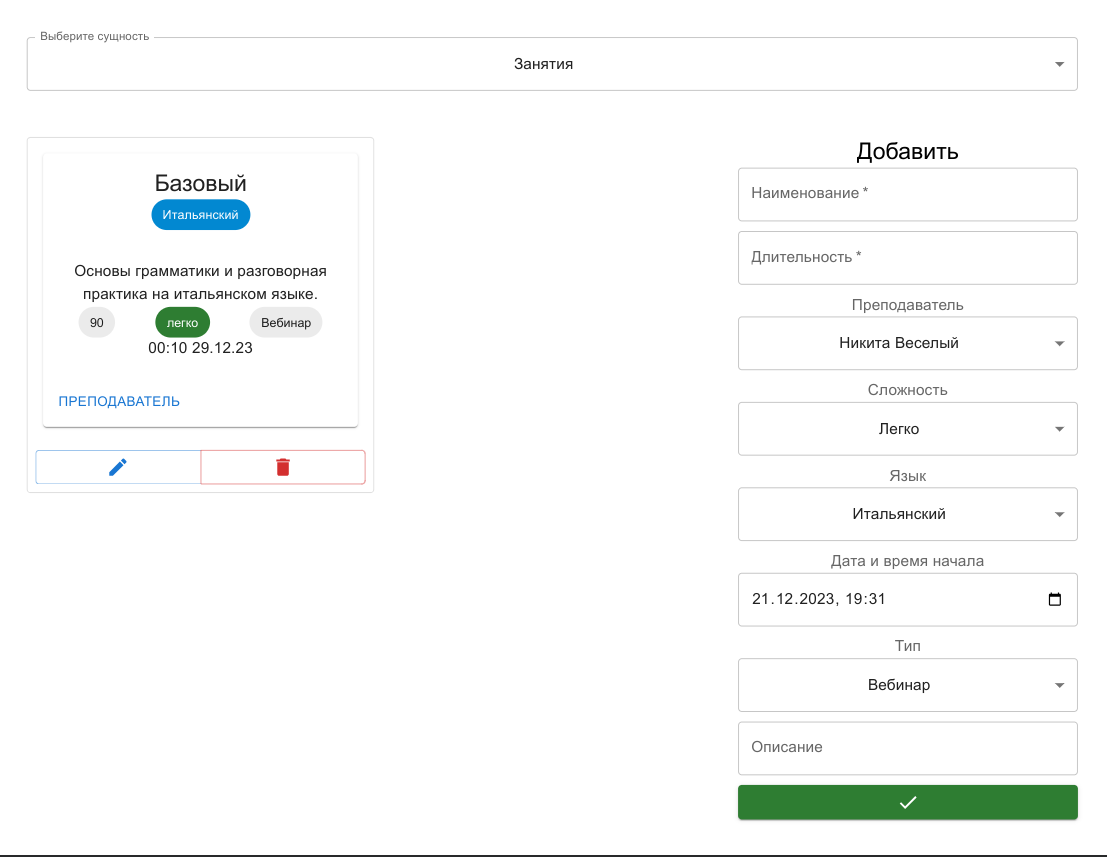


Рис. 6.17 – Результат создания занятия

После отправки формы, мы создали занятие. На рис. 6.17 изображена страница с созданной карточкой занятия

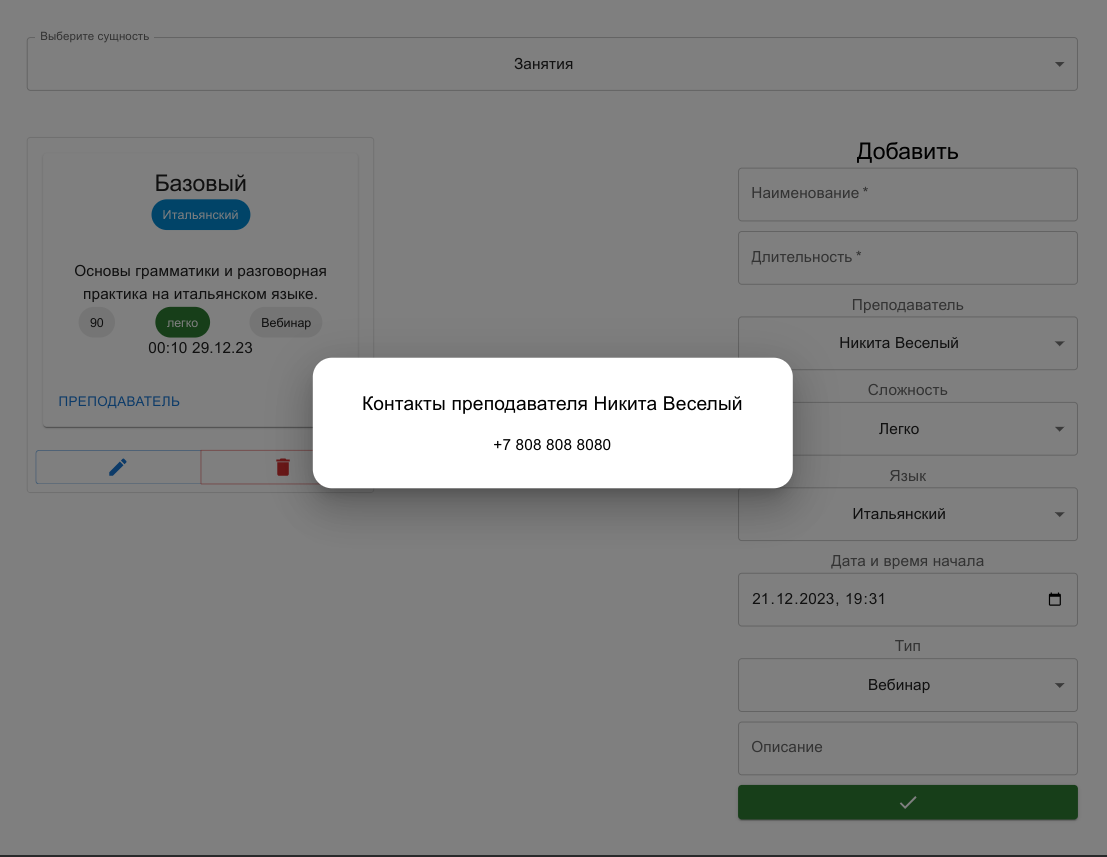


Рис. 6.18 – Обзор преподавателя занятия

На рис. 6.18 изображены данные преподавателя в результате нажатия кнопки «ПРЕПОДАВАТЕЛЬ» на карточке занятия.

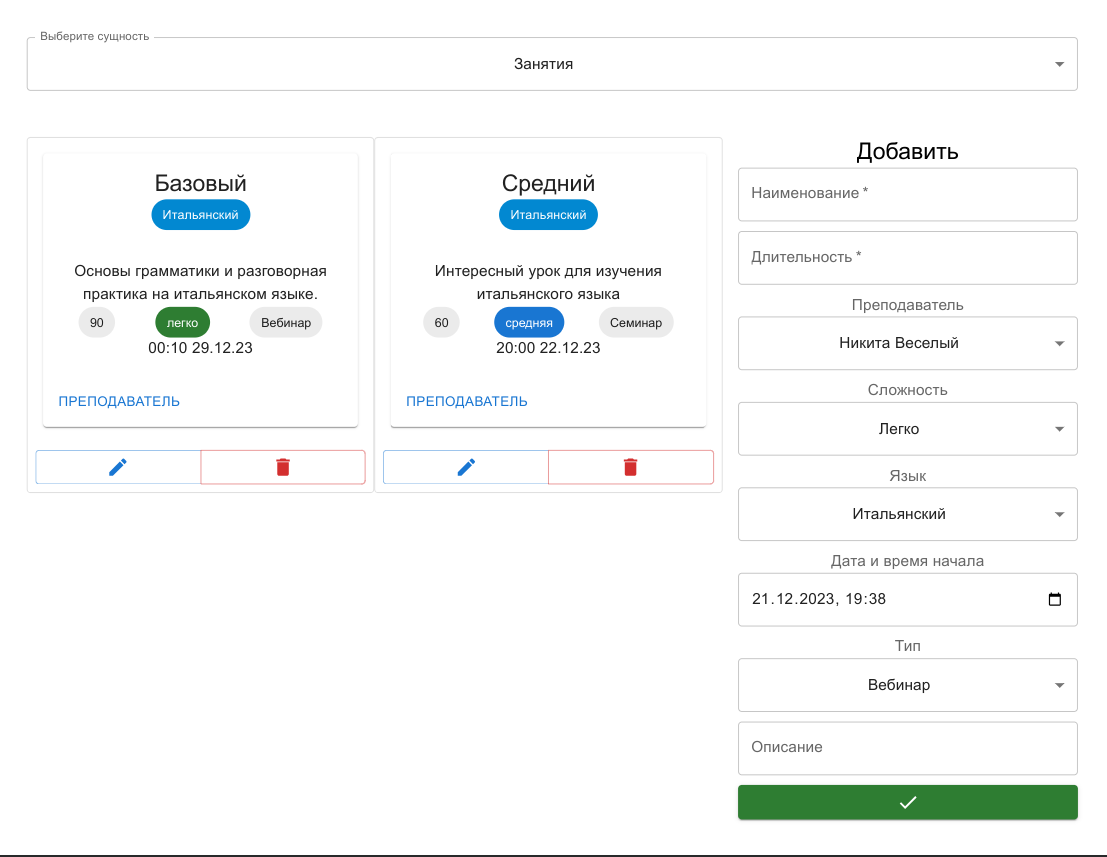


Рис. 6.19 – Создание второго занятия

На рис. 6.19 изображены занятия после создания нового под названием «Средний».

# Список используемых источников

1. Бен Фрейн "CSS3 в примерах", Питер, 2020, 288 стр. (ориг. название: "CSS in Depth", Manning Publications)
2. Эрик Фримен, Элизабет Робсон "JavaScript. Подробное руководство", Вильямс, 2022, 992 стр. (ориг. название: "JavaScript: The Definitive Guide", O'Reilly Media)
3. Крейг Уоллс "Spring Boot в действии", Питер, 2020, 616 стр. (ориг. название: "Spring Boot in Action", Manning Publications)
4. Крейг Уоллс "Spring в действии", Питер, 2020, 768 стр. (ориг. название: "Spring in Action", Manning Publications)
5. Герберт Шилдт "Java: Руководство для начинающих", ДМК Пресс, 2021, 704 стр. (ориг. название: "Java: A Beginner's Guide", McGraw-Hill)
6. Эрик Фримен, Элизабет Робсон "HTML и CSS. Дизайн и верстка сайтов", Вильямс, 2021, 720 стр. (ориг. название: "Head First HTML and CSS", O'Reilly Media)
7. Джон Дакетт "HTML5: Руководство разработчика", Питер, 2021, 944 стр. (ориг. название: "HTML5 Developer's Guide", McGraw-Hill)
8. Дэвид Флэнаган "JavaScript: Подробное руководство", Вильямс, 2022, 1224 стр. (ориг. название: "JavaScript: The Definitive Guide", O'Reilly Media)
9. Крис Шейфер, Джон Уайсс "Spring Framework. Практика разработки Enterprise-приложений", Питер, 2022, 784 стр. (ориг. название: "Pro Spring 5", Apress)
10. Дэвид Геккер, Джон Оспорн "React: Быстрый старт", Питер, 2021, 320 стр. (ориг. название: "Learning React: Modern Patterns for Developing React Apps", O'Reilly Media)

# Приложение А – Листинг программного кода

package ru.michael.backend.entity;  
  
import jakarta.persistence.\*;  
import lombok.Getter;  
import lombok.Setter;  
import org.hibernate.proxy.HibernateProxy;  
  
import java.util.LinkedHashSet;  
import java.util.Objects;  
import java.util.Set;  
  
@Getter  
@Setter  
@Entity@Table(name = "language")  
public class Language {  
 @Id  
 @GeneratedValue(strategy = GenerationType.AUTO)  
 @Column(name = "id", nullable = false)  
 private Long id;  
  
 @Column(name = "name", length = 50)  
 private String name;  
  
 @OneToMany(mappedBy = "language", cascade = CascadeType.ALL, orphanRemoval = true)  
 private Set<Lesson> lessons;  
  
 @Override  
 public final boolean equals(Object o) {  
 if (this == o) return true;  
 if (o == null) return false;  
 Class<?> oEffectiveClass = o instanceof HibernateProxy ? ((HibernateProxy) o).getHibernateLazyInitializer().getPersistentClass() : o.getClass();  
 Class<?> thisEffectiveClass = this instanceof HibernateProxy ? ((HibernateProxy) this).getHibernateLazyInitializer().getPersistentClass() : this.getClass();  
 if (thisEffectiveClass != oEffectiveClass) return false;  
 Language language = (Language) o;  
 return getId() != null && Objects.equals(getId(), language.getId());  
 }  
  
 @Override  
 public final int hashCode() {  
 return this instanceof HibernateProxy ? ((HibernateProxy) this).getHibernateLazyInitializer().getPersistentClass().hashCode() : getClass().hashCode();  
 }  
}

package ru.michael.backend.entity;  
  
import jakarta.persistence.\*;  
import lombok.Getter;  
import lombok.Setter;  
  
import java.util.Date;  
  
@Getter  
@Setter  
@Entity  
@Table(name = "lesson")  
public class Lesson {  
 @Id  
 @GeneratedValue(strategy = GenerationType.AUTO)  
 @Column(name = "id", nullable = false)  
 private Long id;  
  
 @Column(name = "name", length = 150)  
 private String name;  
  
 @Column(name = "description")  
 private String description;  
  
 @Temporal(TemporalType.TIMESTAMP)  
 @Column(name = "timestamp")  
 private Date timestamp;  
  
 @Column(name = "duration")  
 private Integer duration;  
  
  
 @ManyToOne  
 @JoinColumn(name = "teacher\_id")  
 private Teacher teacher;  
  
 @ManyToOne  
 @JoinColumn(name = "language\_id")  
 private Language language;  
  
 @ManyToOne  
 @JoinColumn(name = "lesson\_type\_id")  
 private LessonType lessonType;  
  
  
 @Enumerated(EnumType.STRING)  
 @Column(name = "lesson\_difficulty", length = 20)  
 private Difficulty difficulty;  
  
}

package ru.michael.backend.entity;  
  
import jakarta.persistence.\*;  
import lombok.Getter;  
import lombok.Setter;  
  
import java.util.LinkedHashSet;  
import java.util.Set;  
  
@Getter  
@Setter  
@Entity  
@Table(name = "lesson\_type")  
public class LessonType {  
 @Id  
 @GeneratedValue(strategy = GenerationType.AUTO)  
 @Column(name = "id", nullable = false)  
 private Long id;  
  
 @Column(name = "name", length = 50)  
 private String name;  
  
 @OneToMany(mappedBy = "lessonType", cascade = CascadeType.ALL, orphanRemoval = true)  
 private Set<Lesson> lessons;  
  
}

package ru.michael.backend.entity;  
  
import jakarta.persistence.\*;  
import lombok.Getter;  
import lombok.Setter;  
  
import java.util.Set;  
  
@Getter  
@Setter  
@Entity  
@Table(name = "teacher")  
public class Teacher {  
 @Id  
 @GeneratedValue(strategy = GenerationType.AUTO)  
 @Column(name = "id", nullable = false)  
 private Long id;  
  
 @Column(name = "firstname", length = 50)  
 private String firstname;  
  
 @Column(name = "lastname", length = 50)  
 private String lastname;  
  
 @Column(name = "expirience")  
 private Integer experience;  
  
 @Column(name = "contacts", length = 250)  
 private String contacts;  
  
 @OneToMany(targetEntity = Lesson.class, mappedBy = "teacher",  
 cascade = CascadeType.ALL)  
 private Set<Lesson> lessons;  
  
  
}

package ru.michael.backend.entity;  
  
public enum Difficulty {  
 EASY,  
 HARD,  
 MIDDLE;  
}