МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
**«Южно-Уральский государственный университет
(национальный исследовательский университет)»**
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

ДОПУСТИТЬ К ЗАЩИТЕ

Заведующий кафедрой, д.ф.-м.н.,
профессор

_____ Л.Б. Соколинский

«___»_____ 2024 г.

Разработка мобильного приложения для организации дополнительного образования «Продленка»

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 09.03.04.2024.308-647.ВКР

Научный руководитель,
доцент кафедры СП, к.ф.-м.н.
_____ Т.Ю. Маковецкая

Автор работы,
студент группы КЭ-433
_____ И.С. Дремлюга

Ученый секретарь
(нормоконтролер)
_____ И.Д. Володченко
«___»_____ 2024 г.

Челябинск, 2024 г.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

УТВЕРЖДАЮ
Зав. кафедрой СП
_____ Л.Б. Соколинский
29.01.2024 г.

ЗАДАНИЕ
на выполнение выпускной квалификационной работы бакалавра
студенту группы КЭ-433
Дремлюга Ивану Сергеевичу,
обучающемуся по направлению
09.03.04 «Программная инженерия»

1. **Тема работы** (утверждена приказом ректора от 22.04.2024 г. №764-13/12)
Разработка мобильного приложения для организации дополнительного образования «Продленка».
2. **Срок сдачи студентом законченной работы:** 03.06.2024 г.
3. **Исходные данные к работе**

3.1. GitHub. [Электронный ресурс] URL: https://github.com/ (дата обращения: 29.01.2024 г.).
3.2. Нидерст Роббинс Дженнифер. HTML5, CSS3 и JavaScript исчерпывающее руководство (+ DVD-ROM). 5-е издание. // Эксмо, Москва, 2015. – 192 с.
3.3. Руководство по MySQL. [Электронный ресурс] URL: https://metanit.com/sql/mysql/ (дата обращения: 29.01.2024 г.).
3.4. Вывод MySQL в интернет. [Электронный ресурс] URL: https://www.phpmyadmin.net/docs/ (дата обращения: 29.01.2024 г.).
3.5. Open Server. [Электронный ресурс] URL: http://open-server.ru/ (дата обращения: 29.01.2024 г.).
3.6. Шмитт К. CSS рецепты программирования. 3-е издание. // Санкт Петербург, 2011. – 672 с.

4. **Перечень подлежащих разработке вопросов**

4.1. Выполнить обзор литературы.
4.2. Изучить аналогичные проекты.

4.3. Составить требования к проектируемой системе.
4.4. Реализовать систему.
4.5. Провести тестирование реализованной системы.
4.6. Сделать выводы о проделанной работе.
5. Дата выдачи задания: 29.01.2024 г.

Научный руководитель,
доцент кафедры СП, к.ф.-м.н. Т.Ю. Маковецкая

Задание принял к исполнению И.С. Дремлюга

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность

В наше быстро меняющееся время, в условиях стремительного развития технологий и повседневных потребностей общества, детские развивающие клубы становятся важным элементом образовательного процесса. Создание удобного и эффективного мобильного приложения для таких клубов становится неотъемлемой частью их управления и предоставления услуг. Приложения обеспечивают быстрый доступ к информации и улучшают общий опыт взаимодействия с пользователями.

Постановка задачи

Целью данной выпускной квалификационной работы является разработка мобильного приложения для организации дополнительного образования «Продленка».

Для достижения цели работы ставятся следующие задачи.

1. Выполнить обзор литературы.
2. Изучить аналогичные проекты.
3. Составить требования к проектируемой системе.
4. Реализовать систему.
5. Провести тестирование реализованной системы.
6. Сделать выводы о проделанной работе.

Структура и содержание работы

Работа состоит из введения, четырех глав, заключения и списка литературы. Объем работы составляет 80 страниц, объем списка литературы – 20 источников, также использовано 27 рисунков, 7 таблиц и 33 листинга.

Первая глава посвящена предметной области проекта. В ней представлено описание разрабатываемой системы, включая ее основные функции и цели. В этой главе проводится анализ аналогичных проектов, что позволяет выделить уникальные особенности разрабатываемого приложения. Анализ включает сравнение функциональных возможностей, удобства использования и других параметров, чтобы подчеркнуть преимущества и необходимые

функции для повышения удобства использования приложения.

Вторая глава описывает требования к проектируемой системе. Здесь представлены функциональные и нефункциональные требования, которые система должна удовлетворять. Включена диаграмма вариантов использования, демонстрирующая основные сценарии взаимодействия пользователей с системой. Также в этой главе рассматривается проектирование системы, включая ее архитектуру, проектирование интерфейса и базы данных. Архитектурная часть описывает компоненты системы и их взаимодействие, проектирование интерфейса охватывает пользовательский интерфейс, а проектирование базы данных включает схемы данных и их связи.

Третья глава охватывает реализацию системы. В ней описаны использованные инструменты разработки и среды. Глава подробно рассматривает процесс создания базы данных, включая создание таблиц, их связи и начальное заполнение данными. Реализация интерфейса охватывает разработку пользовательского интерфейса. Реализация серверной части включает разработку логики, обработку запросов, взаимодействие с базой данных и обеспечение безопасности системы. Также описывается процесс размещения системы на сервере, включая настройку домена, подключение сертификата безопасности.

Четвертая глава рассматривает тестирование системы. В ней представлены методы тестирования, используемые для проверки функциональности системы. Описаны сценарии использования, которые проверяют, соответствует ли система предъявленным требованиям. Глава включает функциональное тестирование, проверяющее работу основных функций системы, а также тестирование, оценивающее удобство интерфейса.

Работа включает приложения А и Б. В приложении А приведены скриншоты разработанного приложения, а в приложении Б листинг программного кода.

1. ПРЕДМЕТНАЯ ОБЛАСТЬ ПРОЕКТА

1.1. Описание разрабатываемой системы

В рамках выпускной квалификационной работы разрабатывается мобильное приложение для организации дополнительного образования «Продленка». Организация «Продленка» предоставляет разнообразные услуги дополнительного образования специально для детей и нацелено на обеспечение качественного образования и развития детей в разных направлениях.

Данный клуб предоставляет разнообразные образовательные услуги, включая продленное пребывание для детей, репетиторство, занятия каллиграфией, обучение английскому языку, комплексные развивающие занятия для дошкольников, танцы и физкультуру. Организация также заботится о выполнении домашних заданий, обеспечивает присмотр за детьми, предоставляет полноценный обед и предоставляет возможность аренды помещения для детских мероприятий, а также мероприятий у взрослых.

Разрабатываемое приложение должно помочь облегчить взаимодействие пользователей с организацией, набирать и отслеживать клиентскую базу. Приложение должно быть доступно любому пользователю на любом устройстве и легко обслуживаться, таким образом было принято решении о создании приложения в формате мобильного веб-приложения. В настоящее время мобильные веб-приложения набирают огромную популярность из-за их доступности, легкости в обслуживании со стороны организации, пользователю же нет необходимости проводить дополнительные манипуляции для обновления приложения, а также веб-приложение доступно с любого устройства, что делает его идеальным вариантом для организации.

1.2. Анализ аналогичных проектов

Анализ аналогичных проектов дает понять, каким должно быть приложение для удобства его использования. К тому же, подчеркивает важность создания удобной навигации и минималистичного дизайна интерфейса, а также добавление необходимых функций в конечное приложение.

В таблице 1 приведен анализ аналогичных проектов.

Таблица 1 – Анализ аналогичных проектов

Приложение	Достоинства
GYM STATION	Приложение для клиентов клуба «GYM STATION». С его помощью можно: получать информацию о клубе; включая фотографии; контактную информацию; режим работы; расписание всех занятий; отправлять заявку на запись в клуб дистанционно, а также получать уведомления о приближающемся мероприятии.
Фитнес-клуб «Форма»	Приложение для клиентов клуба «Форма». С помощью него можно следить за клубом, расписанием тренировок, получать информацию об изменениях в расписании, записываться на тренировки.
Stepik:Онлайн курсы	Это платформа с онлайн курсами. На платформе можно найти курсы по различным предметам и другим дисциплинам, присутствует удобный фильтр для поиска нужного курса.

При анализе аналогичных проектов полных аналогов найдено не было. Однако, обнаружены ключевые функциональные возможности, типичные для большинства аналогов, такие как управление расписаниями и регистрация на курсы. Тем не менее, каждое изученное приложение имеет свои уникальные особенности, выделяющие его на общем фоне. Но любое приложение имеет свои недостатки. Выявление недостатков является важным аспектом в разработке, оно помогает улучшить разрабатываемое приложение. У данных приложений наблюдается ограниченная совместимость, что затрудняет их использование на разных устройствах и операционных системах. А также некоторые имеют перегруженный интерфейс, что сбивать с толку новых пользователей, снижая общее удобство использования.

Таким образом, проводя сравнительную характеристику, было принято решение о разработке веб-приложения, как наиболее универсального варианта.

Вывод по первой главе

После изучения существующих решений и литературы был выявлен набор средств для реализации поставленной задачи, наиболее полно удовлетворяющий требованиям к подобного рода системам.

2. ПРОЕКТИРОВАНИЕ

2.1. Определение требований

Функциональные требования к проектируемой системе. К ним относятся ключевые операции и сервисы, которые система должна выполнить для обеспечения своей целевой функциональности. В данном проекте к функциональным требованиям можно отнести следующие.

1. Система должна обеспечивать возможность регистрации и входа для пользователей с использованием уникального логина и пароля.
2. Пользователи должны иметь возможность просматривать расписание курсов, включая дату, время и доступные места.
3. Пользователи должны иметь доступ к своему профилю, где отображаются их личные данные, информация о записях на курсы и другие персональные данные.
4. Пользователи должны иметь возможность изменять свои личные данные, такие как имя, email, номер телефона, логин и пароль.
5. Система должна предоставлять информацию о доступных курсах, включая описание.
6. Пользователи должны иметь возможность записываться на выбранные курсы.
7. Система должна предоставлять информацию о преподавателях.

Эти пункты отражают возможности, которые система предоставляет пользователям для достижения удобства использования, а также для облегчения управления их учебным процессом.

К нефункциональным требованиям относятся требования, определяющие условия и ограничения, в рамках которых система должна функционировать, влияя на ее эффективность, надежность и удобство использования. В данном проекте к нефункциональным требованиям можно отнести следующие.

1. Система должна быть спроектирована таким образом, чтобы легко

масштабироваться при увеличении количества пользователей и объема данных.

2. Система должна быть простой в использовании и понятной для пользователей с разным уровнем технической подготовки.

3. Система должна обеспечивать защиту персональных данных пользователей.

4. Система должна обеспечивать быструю и стабильную работу, минимизируя время отклика на запросы пользователей.

5. Система должна обеспечивать высокую доступность и минимальные простои, чтобы пользователи могли воспользоваться сервисом в любое время.

Эти нефункциональные требования обеспечивают работу системы, ее надежность и удобство для пользователей, а также соответствие современным стандартам безопасности и производительности.

2.2. Варианты использования

В таблице 2 приведена демонстрация одного из основных сценариев взаимодействия пользователя с системой. Она включает описание действий пользователя и задействованных актеров при записи на курс.

Таблица 2 – Основной вариант использования

Прецедент: Вход
ID: 1
Краткое описание: Вход в аккаунт
Главные актеры Пользователь
Второстепенные актеры: Администратор
Предусловия: Система должна быть установлена и работать корректно
Предусловия: 1. Система должна быть установлена и работать корректно. 2. Пользователь должен иметь мобильное устройство с поддержкой приложения.

Окончание таблицы 2

Основной поток: 1. Пользователь устанавливает мобильное приложение на свое устройство. 2. Пользователь открывает приложение и авторизуется в системе. 3. Пользователь получает список доступных ему курсов. 4. Пользователь может записаться на курс. Система регистрирует действие пользователя и записывает его на курс.
Постусловия: Система записывает действие пользователя.
Альтернативный поток (неуспешный вход): 1. Приложение требует ввода логина и пароля. 2. Пользователь вводит неправильный логин и/или пароль. Система отклоняет попытку входа и приложение сообщает об ошибке.

На основе имеющихся требований, была разработана диаграмма вариантов использования системы для авторизированного пользователя, администратора и неавторизированного пользователя, она представлена на рисунке 1. Диаграмма показывает основные варианты использования системы, предоставляя пользователям и администраторам весь необходимый функционал для управления учебным процессом.

Рисунок 1 – Диаграмма вариантов использования

2.3. Описание компонентов, составляющих систему

Рассмотрение деталей функционирования системы, первым делом, это анализ ее ключевых компонентов, каждый из которых играет свою роль в общей схеме работы приложения.

Фронтенд (клиентская часть). Система предоставляет пользователям клиентский интерфейс, который включает элементы управления и навигации по основным функциям приложения, они представлены ниже [14].

1. Главная страница – навигационная панель и основные действия.
2. Страница профиля – вывод информации о курсах, кнопки для удаления курсов и перехода к выбору нового курса, так же редактирование личных данных и выход из аккаунта.
3. Страница записи на курс – форма для выбора курса, даты, времени и ребенка для записи на курс.
4. Страница расписания – просмотр личного расписания пользователя.
5. Информационная страница – содержит контактную информацию об организации.
6. Страница курсы – страница с общей информацией о курсах.
7. Страница просмотра курсов, на которые записан пользователь – содержит информацию о дате, времени, ребенке и кнопку для того, чтобы отменить запись.

Бэкенд (серверная часть). Серверная часть обеспечивает обработку данных и взаимодействие с базой данных, что является ключевым для функционирования системы. Бэкенд имеет следующие части [12].

1. PHP-скрипты – обработка запросов от клиента, взаимодействие с базой данных.
2. Java-скрипты – используется для создания динамических и интерактивных элементов.
3. MySQL база данных – содержит таблицы, которые хранят все необходимые данные.

Данный проект представляет собой веб-приложение для предоставления услуг, основной функционал системы связан с записью на курсы и просмотром расписания. На рисунке 2 представлена диаграмма взаимодействия для записи на курс.

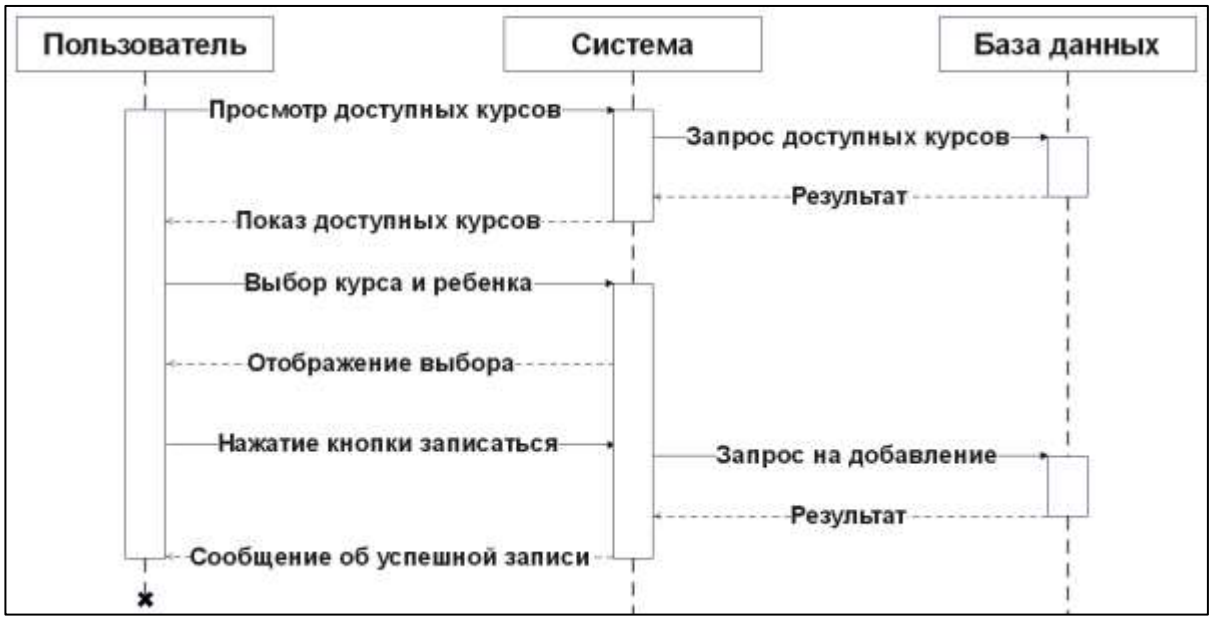

Рисунок 2 – Диаграмма взаимодействия

2.4. Архитектура системы

Компоненты системы определяют ее функциональность. Диаграмма компонентов системы наглядно демонстрирует зависимость компонентов между собой.

Диаграмма компонентов системы представлена на рисунке 3.

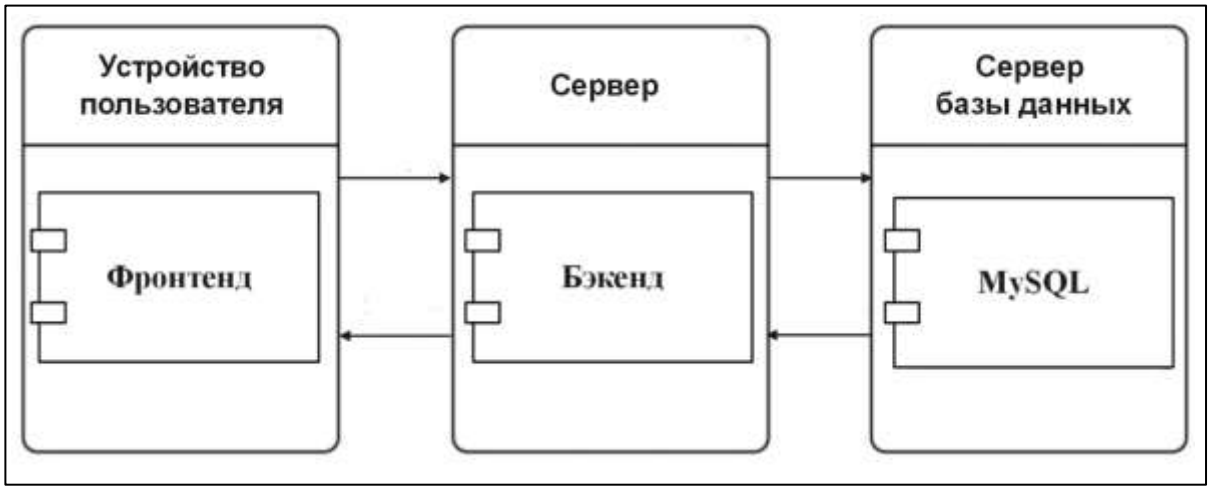

Рисунок 3 – Диаграмма компонентов системы

Устройство пользователя взаимодействует с веб-приложением, где конечные пользователи выполняют свои действия. Веб-приложение служит для отображения пользовательского интерфейса и отправки запросов на веб-сервер, что является основой для взаимодействия пользователя с системой.

Веб-сервер играет ключевую роль в обработке этих запросов и включает в себя файловую систему с необходимыми файлами приложения. Эти файлы включают статические и динамические ресурсы, скрипты, конфигурационные файлы, а также прочие элементы, необходимые для функционирования веб-приложения.

Сервер базы данных MySQL является фундаментом для хранения и обработки данных, поступающих от веб-приложения. Он принимает запросы от веб-сервера и выполняет операции чтения и записи в базу данных, обеспечивая тем самым целостность и безопасность хранимой информации.

2.5. Проектирование фронтенда

Проект включает в себя страницы с веб-интерфейсом для управления процессом обучения, как пользователя, так и администратора.

Все страницы приложения имеют общую стилизацию для поддержания единого визуального стиля. Файл CSS содержит правила для шрифтов, цветовой схемы, отступов и радиусов закругления. Интерфейс создан для максимальной информативности с учетом простоты использования [14].

На главной странице у пользователя предполагается доступ к необходимому функционалу, который предоставляет пользователю обзор доступных действий. На навигационной панели располагаются кнопки на основные разделы приложения: «Профиль», «О нас», «Курсы» и «Расписание». Дизайн страницы создан с использованием светлых тонов и акцентного цвета для легкости восприятия.

На главной странице администратора присутствуют кнопки, отсылающие на основные разделы для управления учебным процессом: «Создать

курс», «Создать расписание», «Редактировать расписание» и «Клиенты».

Макет главной страницы для пользователя представлен на рисунке 4, а макет для главной страницы администратора представлен на рисунке 5.

Рисунок 4 – Макет главной страницы пользователя

Рисунок 5 – Макет главной страницы администратора

2.6. Проектирование базы данных

Проектирование базы данных – это важный этап, включающий в себя определение структуры данных, таблиц и связей между ними. Нужно определить, какая информация будет храниться в базе данных и как она будет организована.

Структура и описание полей базы данных представлены в таблице 3.

Таблица 3 – Структура и описание полей базы данных.

Поле	Тип данных	Описание
id	INT	Уникальный идентификатор записи
user_id	INT	Идентификатор пользователя
child_id	INT	Идентификатор ребенка
schedule_id	INT	Идентификатор расписания
course_id	INT	Идентификатор курса
date	DATE	Дата проведения курса
time	TIME	Время начала курса
available_slots	INT	Доступные места
name	VARCHAR(100)	Имя ребенка
age	INT	Возраст ребенка
gender	CHAR(1)	Пол ребенка
course_name	VARCHAR(100)	Название курса
username	VARCHAR(50)	Логин пользователя
password	VARCHAR(255)	Пароль пользователя
email	VARCHAR(100)	Электронная почта пользователя
full_name	VARCHAR(255)	Полное имя пользователя
is_logged_in	TINYINT(1)	Статус авторизации пользователя
rememberMe	TINYINT(1)	Статус запоминания пользователя
phone_number	VARCHAR(20)	Номер телефона пользователя
registration_date	TIMESTAMP	Дата регистрации пользователя
expiration_date	TIMESTAMP	Дата окончания действия учетной записи
administrator_id	INT	Идентификатор администратора

Таблица предоставляет описание структуры данных, используемой в базе данных данного проекта. В ней перечислены поля таблиц и их типы данных, что позволяет получить общее представление о том, какая информация хранится и как она структурирована.

Схема базы данных на рисунке 6 визуализирует взаимосвязи между различными сущностями и таблицами, что помогает лучше понять организацию и логику работы системы.

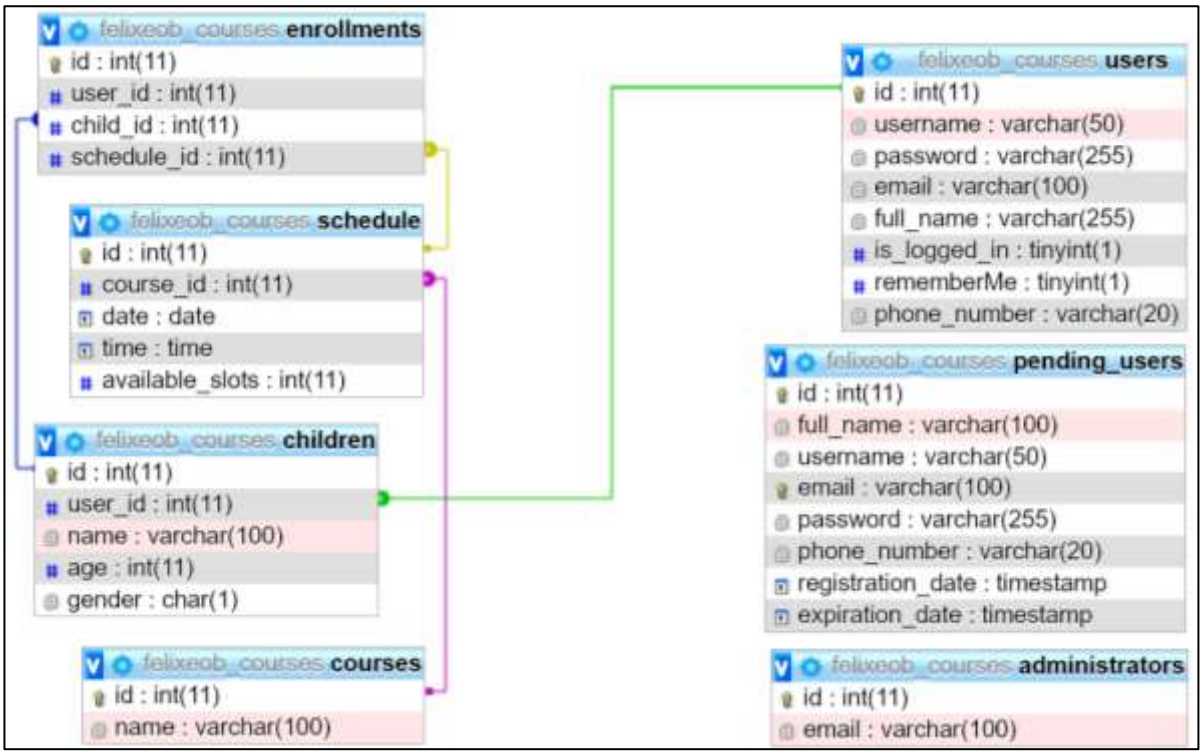

Рисунок 6 – Схема базы данных

На рисунке 7 изображена диаграмма классов для таблиц базы данных.

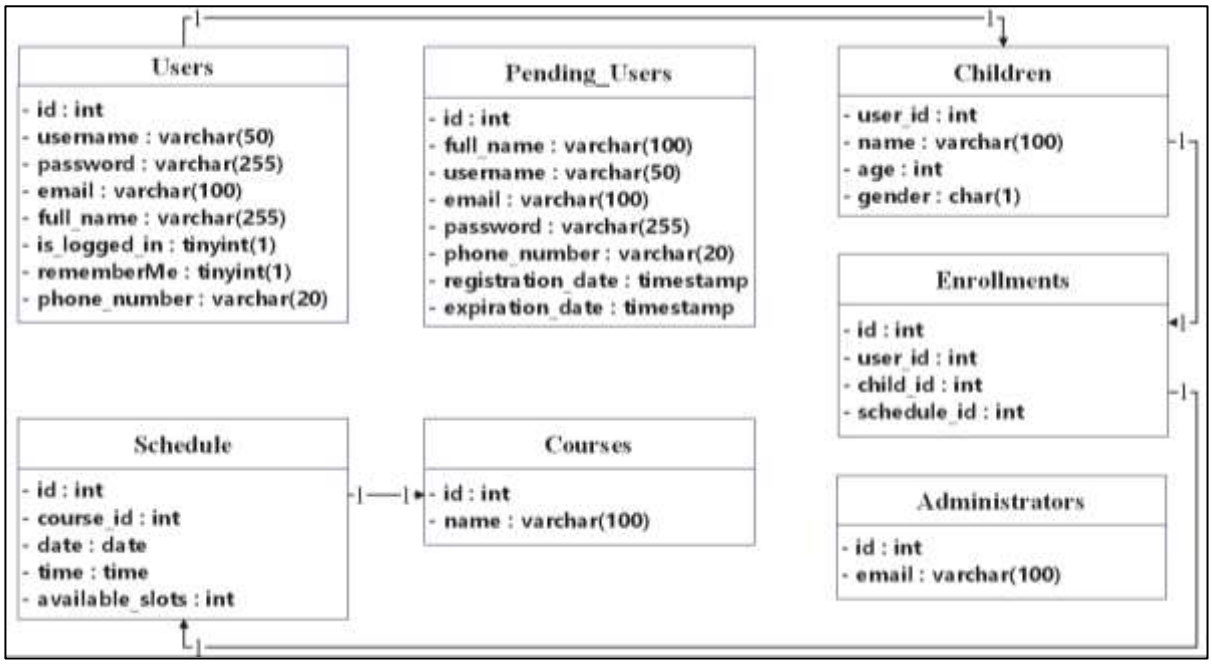

Рисунок 7 – Диаграмма классов таблиц базы данных

Вывод по второй главе

Были определены требования к системе, ее компоненты и варианты использования. Также спроектирован фронтенд и база данных.

3. РЕАЛИЗАЦИЯ СИСТЕМЫ

3.1. Инструменты реализации

Для реализации системы были применены такие средства разработки как:

1) язык программирования JavaScript;
2) язык программирования PHP (Hypertext Preprocessor);
3) язык разметки веб-страниц HTML (HyperText Markup Language);
4) язык описания стилей CSS (Cascading Style Sheets);
5) локальный сервер OpenServer;
6) панель администрирования phpMyAdmin;
7) СУБД MySQL.

Описание используемых средств разработки

JavaScript – это интерпретируемый язык программирования, который используют для написания фронтенд и бэкенд частей сайтов, а также мобильных приложений. JavaScript простыми словами называют языком скриптов или сценариев. Скрипты – это набор инструкций, которые выполняются при загрузке страницы. Браузер самостоятельно интерпретирует код на JavaScript, для этого даже не требуется компиляция (перевод языка программирования в машинный код). Скрипты можно прописать внутри кода страницы или подключить к HTML отдельным файлом [16].

Язык гипертекстовой разметки сайта или HTML (HyperText Markup Language) – код помогающий структурировать содержание каждой веб-страницы. С помощью HTML собирается «скелет». После чего работает с CSS-кодом для стилизации страницы. Далее выполняется взаимодействие с JSON с помощью JavaScript [8].

OpenServer – это портативный локальный сервер, состоящий из многофункциональной управляющей программы с огромным выбором подключаемых компонентов. И она создана для комфортной работы веб-разработчиков [10].

СУБД MySQL – это система управления базами данных (СУБД), позволяющая хранить, организовывать большие объемы данных и манипулировать ими. MySQL является решением для малых и средних приложений. Входит в состав серверов WAMP, AppServ, LAMP и в портативные сборки серверов Денвер, XAMPP, VertrigoServ. Обычно MySQL используется в качестве сервера, к которому обращаются локальные или удаленные клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы [6].

PHP (Hypertext Preprocessor) – это распространенный язык программирования общего назначения с открытым исходным кодом. PHP специально сконструирован для веб-разработок и его код может внедряться непосредственно в HTML [5].

PHPMyAdmin – это бесплатное веб-приложение с открытым исходным кодом, используемое для управления и администрирования баз данных MySQL и MariaDB [6].

CSS (Cascading Style Sheets) – это язык стилей, который используется для описания внешнего вида документа, написанного с использованием языка разметки, такого как HTML или XML. Он позволяет разделить содержимое документа от его внешнего вида, что позволяет легко изменять внешний вид сайта без изменения самого содержимого [12].

3.2. Реализация бэкенда

Реализация бэкенда включает определение тех функций, которые будут в системе. Описание функций для реализации бэкенда показаны в таблице 4.

Таблица 4 – Описание функций

Наименование функции	Назначение
`authorize`	Авторизация пользователя
`displaySchedule`	Отображение раздела расписания
`getMonthName`	Получение названия месяца
`editProfile`	Редактирование профиля
`saveProfile`	Сохранение профиля

Окончание таблицы 4

Наименование функции	Назначение
showDropdown	Отображение выпадающего списка
hideDropdown	Скрытие выпадающего списка
logout	Выход и перенаправление на окно авторизации
loadProfileData	Загрузка данных профиля
addChild	Добавление ребенка в профиль
deleteChild	Удаление ребенка из профиля
enrollCourse	Запись на курс
viewCourses	Просмотр доступных курсов
loadChildren	Загрузка списка детей
getEnrollments	Получение записей на курсы
autoLogin	Автоматический вход пользователя
saveSchedule	Сохранение расписания
deleteSchedule	Удаление расписания
filterSchedules	Фильтрация расписания по дате
renderSchedules	Отображение расписания
showEnrollments	Отображение записей на конкретный курс
confirmPassword	Подтверждение пароля
togglePasswordVisibility	Переключение видимости пароля

Таблица содержит описание функций. Они обеспечивают основные операции, необходимые для работы системы и охватывают, как пользовательскую, так и административную функциональность. Далее приведены описания всех функций.

1. Функция `authorize` позволяет пользователям входить в систему, проверяя их учетные данные.

2. Функция `displaySchedule` отображает пользователям расписание доступных курсов, предоставляя информацию о времени и дате занятий.

3. Функция `getMonthName` используется для получения названия текущего месяца, что облегчает пользователям ориентацию в расписании.

4. Функция `editProfile` позволяет пользователям изменять свои персональные данные.

5. Функция `saveProfile` отвечает за сохранение всех изменений, внесенных пользователем в профиль в базе данных.

6. Функция `showDropdown` активирует отображение выпадающего списка, улучшая удобство навигации в интерфейсе.

7. Функция `hideDropdown` скрывает выпадающий список.

8. Функция `logout` позволяет пользователям безопасно выходить из системы и перенаправляет их на страницу авторизации.

9. Функция `loadProfileData` загружает данные профиля пользователя из базы данных и отображает их в интерфейсе.

10. Функция `addChild` позволяет пользователям добавлять информацию о своих детях, что необходимо для записи на курсы.

11. Функция `deleteChild` предоставляет возможность удалять информацию о детях из профиля пользователя.

12. Функция `enrollCourse` используется для записи детей пользователей на курсы, управляя процессом регистрации и подтверждения.

13. Функция `viewCourses` отображает список всех доступных курсов, предоставляя пользователям возможность выбора.

14. Функция `loadChildren` загружает и отображает список детей, зарегистрированных пользователем.

15. Функция `getEnrollments` позволяет пользователям просматривать свои текущие записи на курсы и управлять ими.

16. Функция `autoLogin` автоматически входит в систему, используя данные, сохраненные в `LocalStorage` для удобства и быстроты доступа.

17. Функция `deleteEnrollment` позволяет пользователям удалять записи на курсы, управляя своим расписанием.

18. Функция `togglePasswordVisibility` переключает видимость пароля в форме ввода, улучшая удобство использования при вводе пароля.

19. Функция `saveSchedule` отвечает за сохранение информации о расписании курсов в базе данных.

20. Функция `deleteSchedule` позволяет удалять записи о расписании курсов из базы данных.

21. Функция `filterSchedules` обеспечивает фильтрацию расписания курсов по дате, облегчая пользователям поиск нужных занятий.

22. Функция `renderSchedules` отвечает за отображение расписания курсов на экране, формируя представление данных для пользователя.

23. Функция `showEnrollments` отображает список записей на конкретный курс, предоставляя полную информацию о записанных пользователях.

24. Функция `confirmPassword` используется для подтверждения пароля пользователя при выполнении важных операций.

Эти функции обеспечивают полную функциональность системы, позволяя пользователям эффективно взаимодействовать с веб-приложением, управлять записями и профилем, а также получать всю информацию. В приложении Б представлены листинги программного кода, включающие все вышеописанные функции.

3.3. Реализация фронтенда

Интерфейс разработанной системы приведен в приложении А. Он отличается чистотой дизайна и интуитивно понятной навигацией. В листинге 22 приложения Б приведен код со стилем [12].

Интерфейс пользователя

Главная страница обеспечивает прямой доступ к основным модулям системы, включая расписание занятий, каталог курсов, личный профиль с индивидуальными настройками и информационный раздел. Элементы управления визуально выделены и позволяют пользователям с легкостью перемещаться по разделам. Яркие и выделяющиеся иконки курсов способствуют быстрому восприятию.

Каждый скриншот иллюстрирует ключевую функциональность, это процесс авторизации, и он выполнен с вводом данных, в то время как система уведомлений об успешной записи на курс предоставляет положительный пользовательский опыт и обратную связь. Управление курсами в личном кабинете предоставляет инструменты для контроля над учебным процессом, включая возможности удаления запланированных занятий. А также просмотр личного расписания.

Интерфейс администратора

Администратор должен иметь полный контроль над всеми аспектами учебного процесса и для этого была реализована панель администратора. Это ключевой элемент системы, предоставляющий административные функции для управления курсами, расписанием и пользователями. Доступ к панели администратора осуществляется через таблицу administrators, где хранятся учетные данные администраторов. Администратор может войти в систему, используя свой email и пароль, после чего ему становятся доступны все функции панели администратора, представленные в таблице 5.

Таблица 5 – Функции панели администратора

Функция	Описание
Создать курс	Добавление нового курса в систему.
Создать расписание	Создание расписания для существующих курсов.
Редактировать расписание	Изменение существующего расписания курсов.
Клиенты	Просмотр и управление клиентами системы.
Выйти	Выход из панели администратора.

Администратор может создавать и редактировать курсы и расписание, что позволяет гибко управлять учебным процессом. Для этого предусмотрены соответствующие формы и элементы управления, которые представлены в приложении А.

Функция создания курса позволяет администратору добавить новый курс. Функция создания расписания предоставляет возможность задать дату, время и количество мест для выбранного курса. Редактирование расписания позволяет изменить существующие записи, а также просмотреть текущие и предстоящие занятия.

Кроме того, администратор может управлять клиентами системы: подтверждать новых пользователей, редактировать их данные или удалять аккаунты. Для фильтрации регистрации нежелательных аккаунтов, данные новых пользователей временно сохраняются в таблице pending_users. Если администратор подтверждает регистрацию, пользователь получает доступ к функционалу системы, а в противном случае его данные удаляются через

неделю или по решению администратора.

Таким образом, панель администратора обеспечивает централизованное управление всеми аспектами системы. Это включает в себя гибкое и эффективное управление, что способствует оптимизации учебного процесса и повышению общего уровня обслуживания.

Подводя итог, можно отметить, что интерфейс не имеет лишних элементов, понятен и прост для конечного пользователя.

3.4. Реализация базы данных

Для эффективного управления данными в системе и обеспечения работы веб-приложения, необходимо правильно организовать структуру базы данных. Это важный шаг, который помогает обеспечить целостность и доступность данных для всех компонентов системы. В системе реализованы несколько таблиц. Это таблицы users, pending_users, administrators, courses, schedule, children, enrollments.

Описание таблиц приведено далее.

1. Таблица users хранит информацию о пользователях системы, включая их учетные данные.

2. Таблица pending_users используется для временного хранения данных пользователей, которые подали заявку на регистрацию, но еще не были подтверждены администратором.

3. Таблица administrators для администраторов хранит учетные данные пользователей с административными привилегиями. Это обеспечивает разграничение доступа и позволяет администраторам управлять системой.

4. Таблица courses для курсов содержит информацию о доступных курсах. Это позволяет пользователям видеть их и выбирать из списка.

5. Таблица schedule хранит информацию о датах, времени и доступных местах для каждого курса. Это ключевой элемент для управления записями на курсы и отображения актуального расписания для пользователей.

6. В таблице children хранятся данные о детях пользователей, которые могут быть записаны на курсы.

7. Таблица enrollments для записей на курсы связывает пользователей и их детей с конкретными курсами и расписаниями. Она помогает отслеживать, кто записан на какой курс и в какое время.

SQL запрос создания базы данных и таблиц праведен в листинге 23 приложения Б. Этот код описывает полноценное создание базы данных со всеми таблицами и типами данных. А SQL запрос вставки данных в таблицу приведен в листинге 24 приложения Б.

Динамическое добавление курсов через веб-форму минимизирует риски ошибок и повышает точность данных в системе.

Ключевым преимуществом такой системы является возможность мгновенного обновления информации, без необходимости глубокого технического вмешательства. Это особенно важно в образовательной среде, где расписание и предложения списка занятий могут часто меняться. Администраторы могут легко управлять курсами, добавлять новые или удалять старые, обеспечивая актуальность и точность информации для всех пользователей [6].

В листинге 1 приведен PHP код с добавлением курса в базу данных.

Листинг 1 – PHP код с добавлением курса в базу данных

```
<?php
$servername = "localhost";
$username = "felixbo_courses";
$password = "Fel22446688";
$dbname = "felixbo_courses";
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
$user_id = 1; // ID пользователя
$child_id = 2; // ID ребенка
$schedule_id = 3; // ID расписания
$sql = "INSERT INTO enrollments (user_id, child_id, schedule_id) VALUES (?,
?, ?)";
$stmt = $conn->prepare($sql);
$stmt->bind_param("iii", $user_id, $child_id, $schedule_id);
if ($stmt->execute() === TRUE) {
    echo "Запись на курс успешно добавлена.";
} else {
    echo "Ошибка при добавлении записи: " . $stmt->error;
}
```

```
$stmt->close();
$conn->close();
?>
```

3.5. Безопасность системы

Безопасность системы является важным аспектом при разработке, особенно когда речь идет о защите данных пользователей и предотвращении несанкционированного доступа. В данном разделе рассмотрены меры, принятые для обеспечения безопасности системы, защиты от SQL-инъекций, фильтрации регистрации нежелательных аккаунтов, хранения паролей и реализации системы авторизации и проверки сессии.

Защита от SQL-инъекций. SQL-инъекции представляют собой одну из наиболее распространенных и опасных уязвимостей приложений. Они возникают, когда злоумышленник имеет возможность вставлять или «инъецировать» вредоносные SQL-запросы через поля ввода данных, такие как формы авторизации или поиска [20]. Это может привести к несанкционированному доступу к данным, их модификации или даже удалению.

Базовый пример SQL-инъекции, когда в форме авторизации пользователь вводит логин и пароль. Пример небезопасного кода представлен в листинге 2.

Листинг 2 – Пример небезопасного кода

```
$username = $_POST['username'];
$password = $_POST['password'];
$sql = "SELECT * FROM users WHERE username=? AND password=?";
$stmt = $conn->prepare($sql);
$stmt->bind_param("ss", $username, $password);
$stmt->execute();
$result = $stmt->get_result();
```

Если злоумышленник введет в поле `username` значение «`'admin' OR '1'='1'`», то запрос станет таким, как приведен в листинге 3.

Листинг 3 – Простой пример SQL-инъекции

```
SELECT * FROM users WHERE username='admin' OR '1'='1' AND password='pass-
word'
```

Более сложный пример SQL-инъекции – это ввод в поле `username` значения «`DROP TABLE users;`», который приведет к выполнению вредоносного запроса. Пример такого кода представлен в листинге 4. Данный пример кода приведет к удалению таблицы users из базы данных.

Листинг 4 – Более сложный пример SQL-инъекции

```
SELECT * FROM users WHERE username=''; DROP TABLE users; --' AND pass-
word='password'
```

Для предотвращения SQL-инъекций используются подготовленные выражения (prepared statements) и привязка параметров. Код для защиты от SQL-инъекций представлен в листинге 5. Подготовленные выражения позволяют параметризировать запросы, отделяя данные от самого запроса, что исключает возможность вмешательства в структуру SQL-запроса.

Листинг 5 – Код защиты от SQL-инъекций

```
<?php
$sql = "SELECT * FROM users WHERE username=?";
$stmt = $conn->prepare($sql);
$stmt->bind_param("s", $username);
$stmt->execute();
$result = $stmt->get_result();
?>
```

Использование подготовленных выражений исключает возможность SQL-инъекций, так как данные не могут изменить структуру SQL-запроса. Даже если злоумышленник попытается вставить вредоносный код, он будет обработан как обычный текст и не сможет повлиять на выполнение запроса.

Фильтрация регистрации нежелательных аккаунтов. Для фильтрации регистрации нежелательных аккаунтов в системе используется таблица pending_users, куда временно сохраняются данные новых пользователей. Администраторы проверяют эти данные, которые хранятся в таблице в течение недели, и только после подтверждения аккаунт становятся активным.

Хранение паролей осуществляется следующим образом: пароли пользователей хранятся в базе данных в зашифрованном виде. Для шифрования используется алгоритм bcrypt, который обеспечивает надежную защиту паролей и их безопасность в случае утечки данных (листинг 6) [18].

Листинг 6 – Хеширование паролей

```
<?php
$hashed_password = password_hash($password, PASSWORD_DEFAULT);
?>
```

Реализация системы авторизации и проверки сессии. Система авторизации и проверки сессии реализована с использованием как серверных, так и клиентских скриптов. На серверной стороне проверка осуществляется с помощью PHP-сессий. В базе данных реализовано отслеживание активных сессий с помощью поля `is_logged_in`, которое показывает, находится ли пользователь в системе (листинг 7). На клиентской стороне используется `LocalStorage` для сохранения данных сессии (листинг 8).

Листинг 7 – Проверка сессии на сервере

```
<?php
session_start();
if (!isset($_SESSION['username']) && !isset($_SESSION['user_id'])) {
    echo "<script>
        if (localStorage.getItem('username') && localStor-
age.getItem('user_id')) {
            fetch('auto_login.php', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json'
                },
                body: JSON.stringify({
                    username: localStorage.getItem('username'),
                    user_id: localStorage.getItem('user_id')
                })
            }).then(response => response.json())
            .then(data => {
                if (data.success) {
                    location.reload();
                } else {
                    window.location.href = 'login.php';
                }
            });
        } else {
            window.location.href = 'login.php';
        }
    </script>";
    exit();
}
?>
```

Листинг 8 – Проверка сессии на стороне клиента

```
<?php
document.addEventListener('DOMContentLoaded', function() {
    if (localStorage.getItem('username') && localStor-
age.getItem('user_id')) {
        fetch('auto_login.php', {
```

```
            method: 'POST',
            headers: {
                'Content-Type': 'application/json'
            },
            body: JSON.stringify({
                username: localStorage.getItem('username'),
                user_id: localStorage.getItem('user_id')
            })
        }).then(response => response.json())
        .then(data => {
            if (data.success) {
                window.location.href = 'Главная.html';
            }
        });
    }
});
?>
```

Ограничения и паттерны ввода. В каждом поле ввода предусмотрены ограничения по длине и формату вводимых данных. Для ввода паролей, номеров телефонов и адресов электронной почты, применяются определенные паттерны ввода и ограничения. Это повышает безопасность и надежность системы, предотвращая ввод некорректных данных. В листинге 9 приведен код с использованием паттернов ввода.

Листинг 9 – Код с использованием паттернов ввода

```
<label for="phone-number">Телефон:</label>
<input type="tel" id="phone-number" name="phone_number" pat-
tern="^\+7\d{10}$" required placeholder="+7XXXXXXXXXX" maxlength="12">
<label for="password">Пароль:</label>
<input type="password" id="password" name="password" required pat-
tern="(?=.*\d)(?=.*[a-z])(?=.*[A-Z]).{6,20}">
```

Эти меры в совокупности обеспечивают надежную защиту системы от SQL-инъекций и других видов атак, обеспечивая безопасность данных пользователей и предотвращение несанкционированного доступа.

3.6. Размещение на сервере

Для размещения веб-приложения, был использован онлайн хостинг beget [20]. Хостинг обеспечивает полный набор инструментов для управления, что позволяет легко и эффективно управлять всеми аспектами работы системы.

Был настроен основной домен для веб-приложения и подключен SSL сертификат для обеспечения безопасного соединения. SSL сертификат позволяет шифровать данные, передаваемые между сервером и пользователями, что повышает уровень безопасности. В результате, веб-приложение доступно по защищенному протоколу HTTPS. На рисунке 8 показана главная страница панели управления хостингом, демонстрирующая информацию о доступных инструментах.

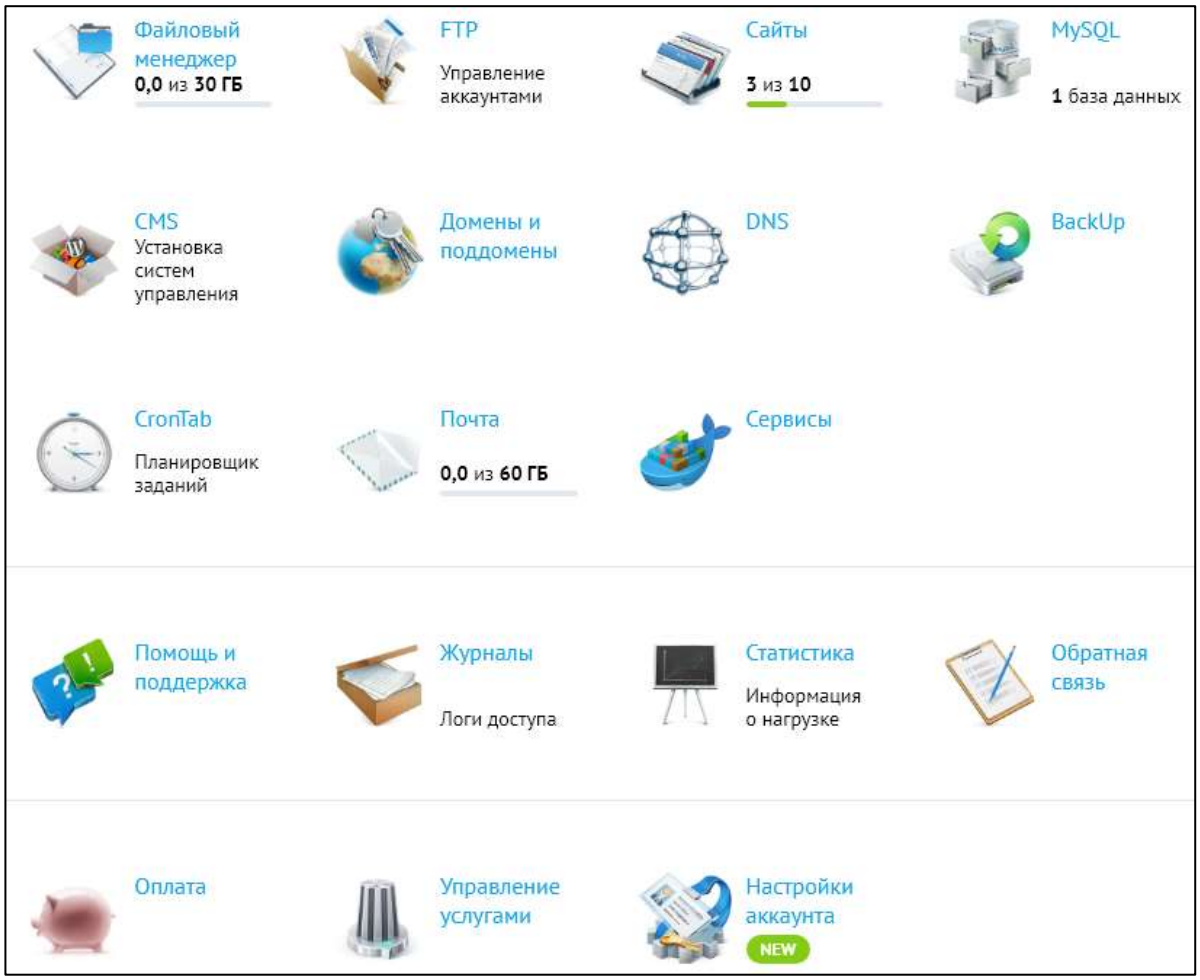

Рисунок 8 – Главная страница панели управления хостингом

Вывод по третьей главе

Были определены инструменты реализации, успешно реализована система, фронтенд и бэкенд, структура базы данных, а также реализована безопасность. Проект был размещен на онлайн хостинге.

4. ТЕСТИРОВАНИЕ СИСТЕМЫ

Было проведено функциональное и юзабилити-тестирования. Тестирование проводилось с целью выявления ошибок и обеспечения работоспособности системы, а также ее удобства.

Функциональное тестирование выполнялось для проверки корректности работы внутренних компонентов системы [16]. Были протестированы следующие сценарии использования, которые приведены в таблице 6.

Таблица 6 – Сценарии функционального тестирования

№	Название теста	Шаги	Ожидаемый результат	Тест пройден?
1	Вход в систему	На странице авторизации ввести валидные данные и нажать кнопку «Вход»	Пользователь успешно авторизуется и перенаправляется на главную страницу	Да
2	Выход из системы	Нажать кнопку «Выход»	Пользователь успешно выходит из системы и перенаправляется на страницу входа	Да
3	Просмотр профиля	Нажать на «Профиль»	Открывается страница профиля с данными	Да
4	Редактирование профиля	В профиле изменить данные и сохранить	Данные профиля успешно обновляются	Да
5	Запись на курс	В расписании выбрать курс и записать ребенка	Ребенок успешно записан на курс	Да
6	Просмотр расписания	Открыть страницу расписания	Расписание отображается корректно	Да
7	Автоматический вход	Авторизироваться и закрыть приложение, а затем открыть снова	Пользователь автоматически входит в систему	Да
8	Просмотр доступных курсов	На странице «Курсы» просмотреть доступные курсы	Отображаются все доступные курсы и информация о них	Да
9	Регистрация пользователя	Заполнить форму регистрации и нажать «Зарегистрироваться»	Пользователь зарегистрирован и может войти в систему	Да
10	Добавление ребенка в профиль	Заполнить форму с данными ребенка и сохранить	Ребенок добавлен в профиль	Да
11	Удаление ребенка из профиля	Нажать на кнопку «Удалить» напротив ребенка в профиле	Ребенок удален из профиля	Да
12	Фильтрация доступных курсов	Выбрать предмет и нажать «Применить»	Отображается выбранный предмет	Да

Юзабилити-тестирование направлено на оценку удобства использования системы, выявление проблем взаимодействия пользователей с интерфейсом и улучшение общего пользовательского опыта. Юзабилити-тестирование проводилось на семи пользователях, которые заведомо не были знакомы с интерфейсом, они отметили, что дизайн и навигация очень простые и интуитивно понятные [17]. Сценарии юзабилити-тестирования приведены в таблице 7.

Таблица 7 – Сценарии юзабилити-тестирования

№	Название теста	Шаги	Ожидаемый результат	Тест пройден?
1	Проверка удобства навигации	Найти раздел с расписанием	Пользователь быстро и без труда находит нужный раздел	Да
2	Проверка простоты задач	Записаться на конкретный курс	Пользователь легко и быстро записывается на курс	Да
3	Оценка визуального восприятия	Оценить внешний вид интерфейса	Пользователь позитивно оценивает визуальные элементы	Да
4	Оценка доступности информации	Найти информацию об организации	Пользователь быстро находит раздел и нужную в нем информацию	Да

Вывод по четвертой главе

Результаты функционального и юзабилити-тестирования показали, что система работоспособна, удобна и интуитивно понятна для пользователей. Система успешно выполняет все заявленные функции и соответствует требованиям проекта. Проблем и багов при работе выявлено не было. Все тесты пройдены успешно, что подтверждает корректную работу системы.

ЗАКЛЮЧЕНИЕ

В заключении данной выпускной квалификационной работы была успешно разработана концепция и функционал мобильного веб-приложения для организации дополнительного образования «Продленка». В ходе работы были учтены основные потребности пользователей, предоставляя им удобные инструменты для взаимодействия с клубом и его ресурсами.

Основные результаты данной работы, следующие:

1) выполнен обзор литературы;
2) изучены аналогичные проекты;
3) составлены требования;
4) система спроектирована;
5) система реализована;
6) проведено тестирование реализованной системы.

Обзор литературы и изучение аналогичных проектов позволили определить современные тенденции и выявить ключевые особенности, востребованные пользователями. На основе этого были сформулированы требования к функциональности и безопасности системы.

Затем была разработана архитектура системы, спроектирован пользовательский интерфейс и структура базы данных. Реализация включала создание и интеграцию базы данных, серверной и клиентской частей приложения, с вниманием к безопасности данных.

Тестирование системы включало функциональные и нефункциональные тесты, подтверждающие корректную работу приложения и его соответствие требованиям.

Таким образом, разработанное приложение отвечает всем требованиям, предоставляя пользователям удобный и инструмент для взаимодействия с организацией дополнительного образования «Продленка».

ЛИТЕРАТУРА

1. Климов А. JavaScript на примерах. // БХВ-Петербург, М., 2017. – 812 с.

2. JavaScript (JS): что это такое, для чего нужен язык программирования. [Электронный ресурс]. URL: https://www.w3schools.com/js/js_intro.asp (дата обращения: 02.02.2024 г.).

3. JavaScript. [Электронный ресурс]. URL: https://blog.skillfatory.ru/glossary/javascript/ (дата обращения: 02.02.2024 г.).

4. Крокфорд Д. JavaScript. Сильные стороны. // Питер, М., 2016. – 262 с.

5. Руководство по PHP. [Электронный ресурс]. URL: https://www.php.net/manual/ru/index.php (дата обращения: 02.02.2024 г.).

6. PHP и MySQL. 25 уроков для начинающих. // БХВ-Петербург, СПб., 2021. – 432 с.

7. Дронов В. JavaScript в Web-дизайне. // БХВ, СПб., 2001. – 880 с.

8. Коржинский С. Настольная книга Web-мастера: эффективное применение HTML, JavaScript. // Кнорус, 2000. – 320 с.

9. Грабер М. Введение в SQL. // Издательство «ЛОРИ», М., 1996. – 382 с.

10. Open Server. [Электронный ресурс]. URL: http://open-server.ru/ (дата обращения: 29.01.2024 г.).

11. Шмитт К. CSS. Рецепты программирования, 3-е изд. // Санкт Петербург, 2011. – 672 стр.

12. Никсон Р. Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5, 4-е изд. // 2016. – 766 с.

13. Томсон Л., Веллинг Л. Разработка web-приложений на PHP и MySQL, 5-е изд. // Москва, Санкт-Петербург, Киев, 2017. – 652 с.

14. Дакетт Дж. HTML и CSS. Разработка и дизайн веб-сайтов, 3-е изд. // Вильямс, Москва, 2019. – 512 с.

15. Флэнаган Д. JavaScript. Подробное руководство, 7-е изд. // О’Рейли, Москва, 2020. – 1096 с.

16. Что такое функциональное тестирование? Типы, примеры, контрольный список и реализация. [Электронный ресурс] URL: https://www.zaptest.com/ru/что-такое-функциональное-тестирован (дата обращения: 04.04.2024 г.).

17. Как провести юзабилити-тестирование. [Электронный ресурс] URL: https://habr.com/ru/companies/lamoda/articles/673884/ (дата обращения: 07.04.2024 г.).

18. Bcrypt Algorithm. [Электронный ресурс] URL: https://www.topcoder.com/thrive/articles/bcrypt-algorithm (дата обращения: 15.04.2024 г.).

19. Защита от SQL-инъекций в PHP и MySQL. [Электронный ресурс] URL: https://habr.com/ru/articles/148701/ (дата обращения: 10.04.2024 г.).

20. Онлайн хостинг beget. [Электронный ресурс] URL: https://beget.com/ru (дата обращения: 20.05.2024 г.).

ПРИЛОЖЕНИЯ

Приложение А. Скриншоты приложения

Скриншоты интерфейса разработанного приложения приведены на рисунках 1–19.

Рисунок 1 – Интерфейс приветственного окна пользователя

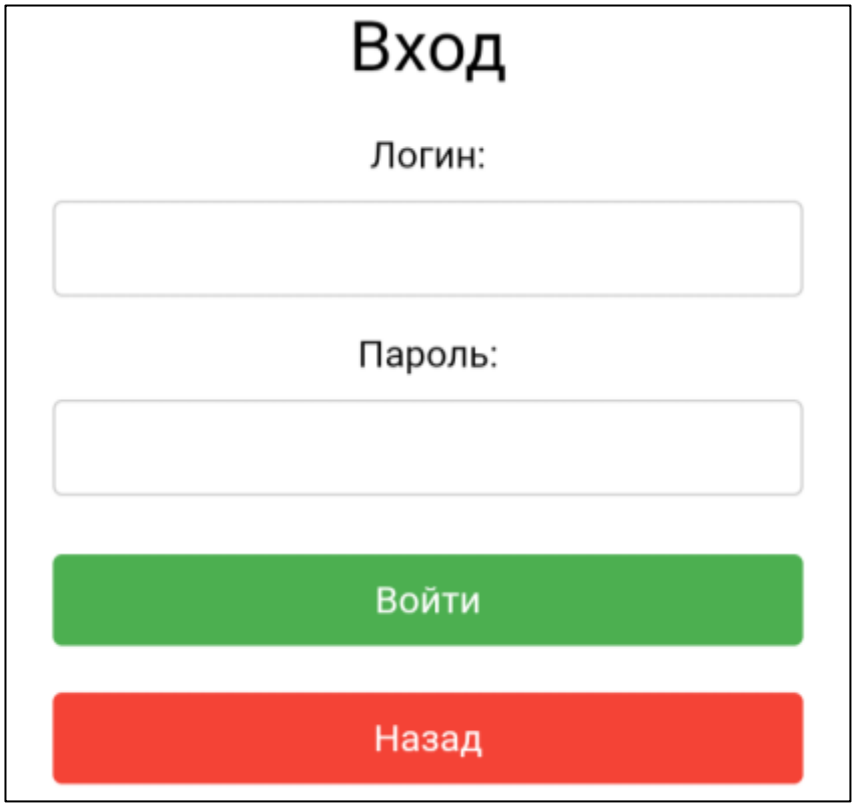

Рисунок 2 – Интерфейс страницы входа

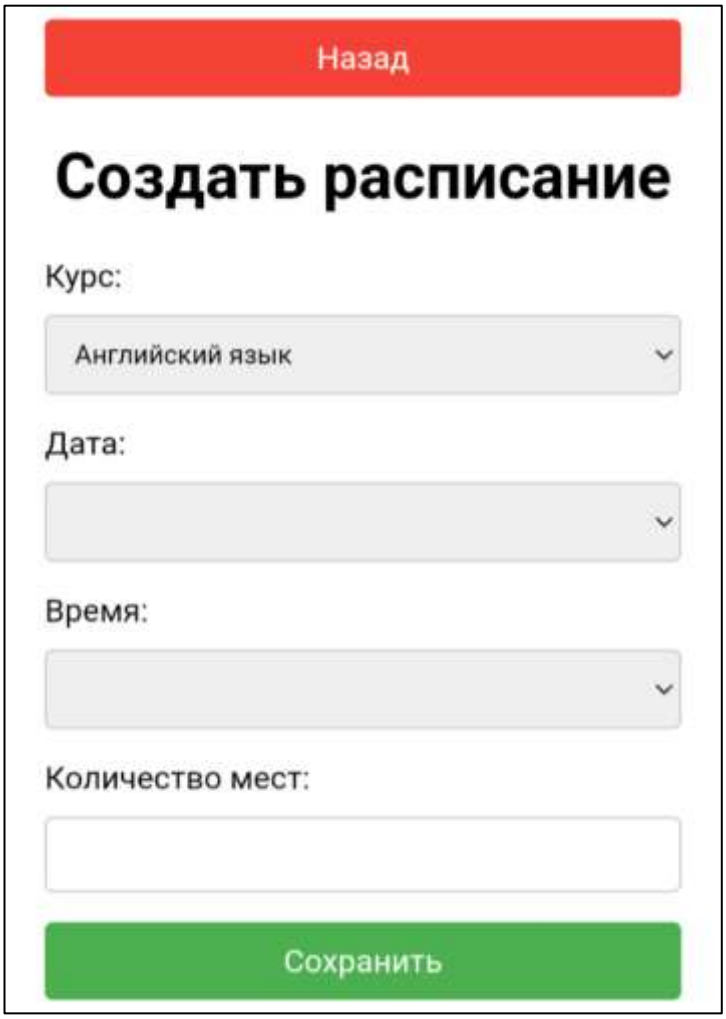

Рисунок 3 – Интерфейс создания расписания у администратора

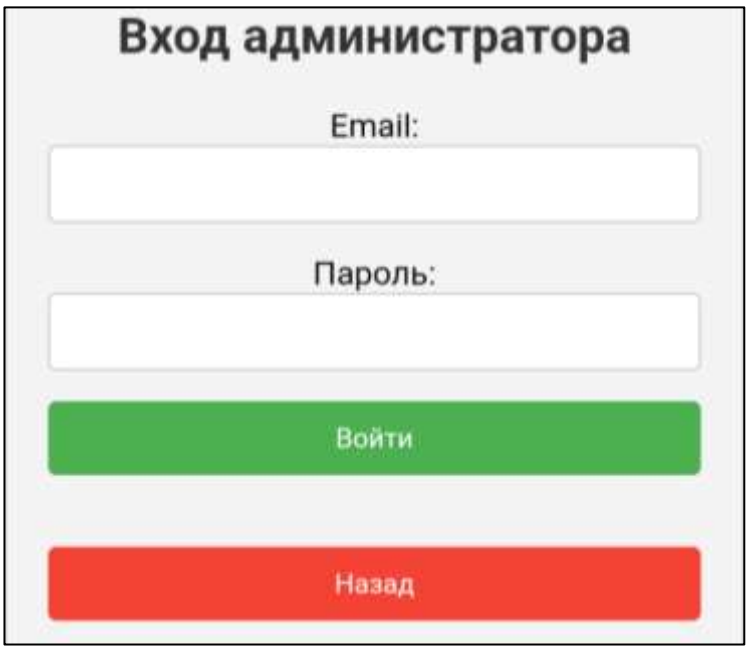

Рисунок 4 – Интерфейс страницы входа администратора

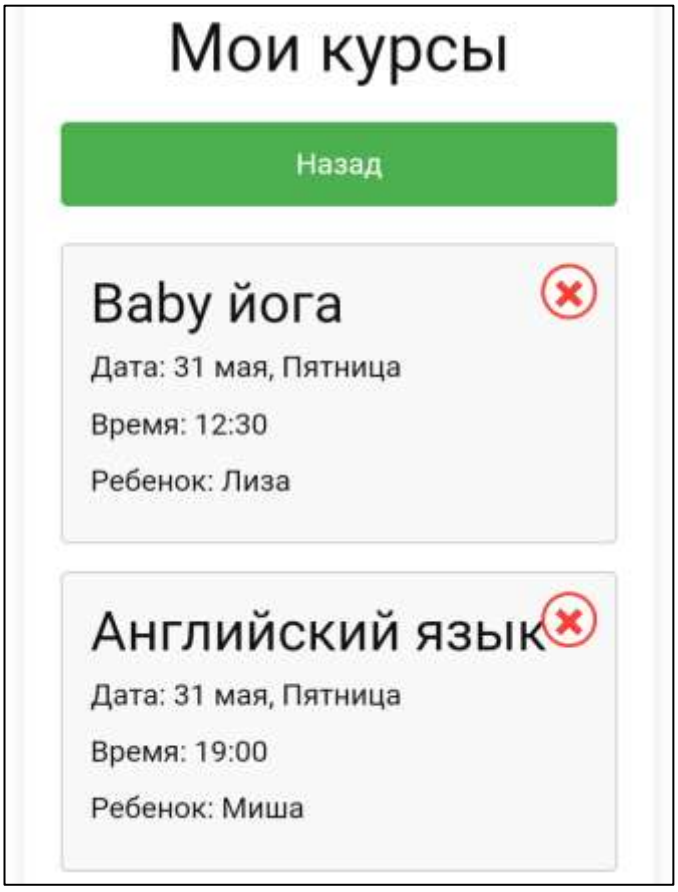

Рисунок 5 – Интерфейс страницы «Мои курсы»

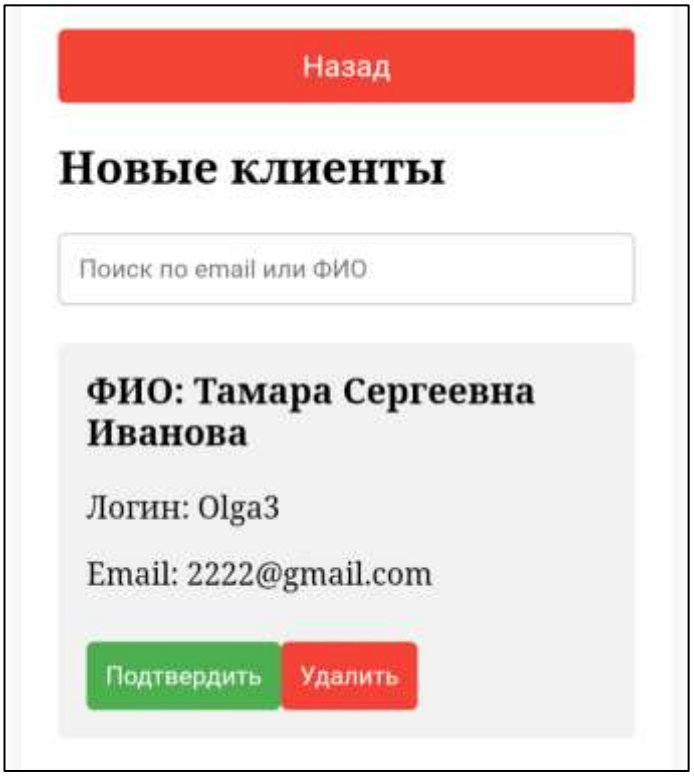

Рисунок 6 – Интерфейс подтверждения администратором нового клиента

Рисунок 7 – Интерфейс просмотра расписания

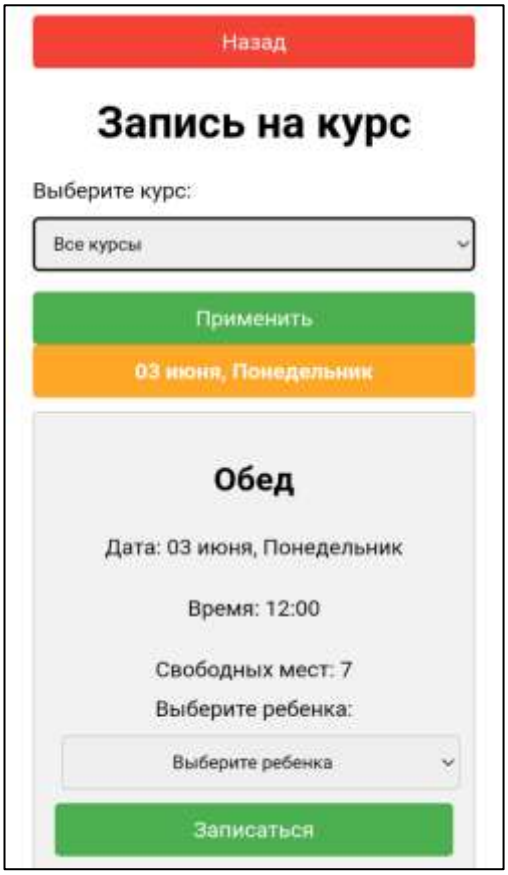

Рисунок 8 – Интерфейс записи на занятие

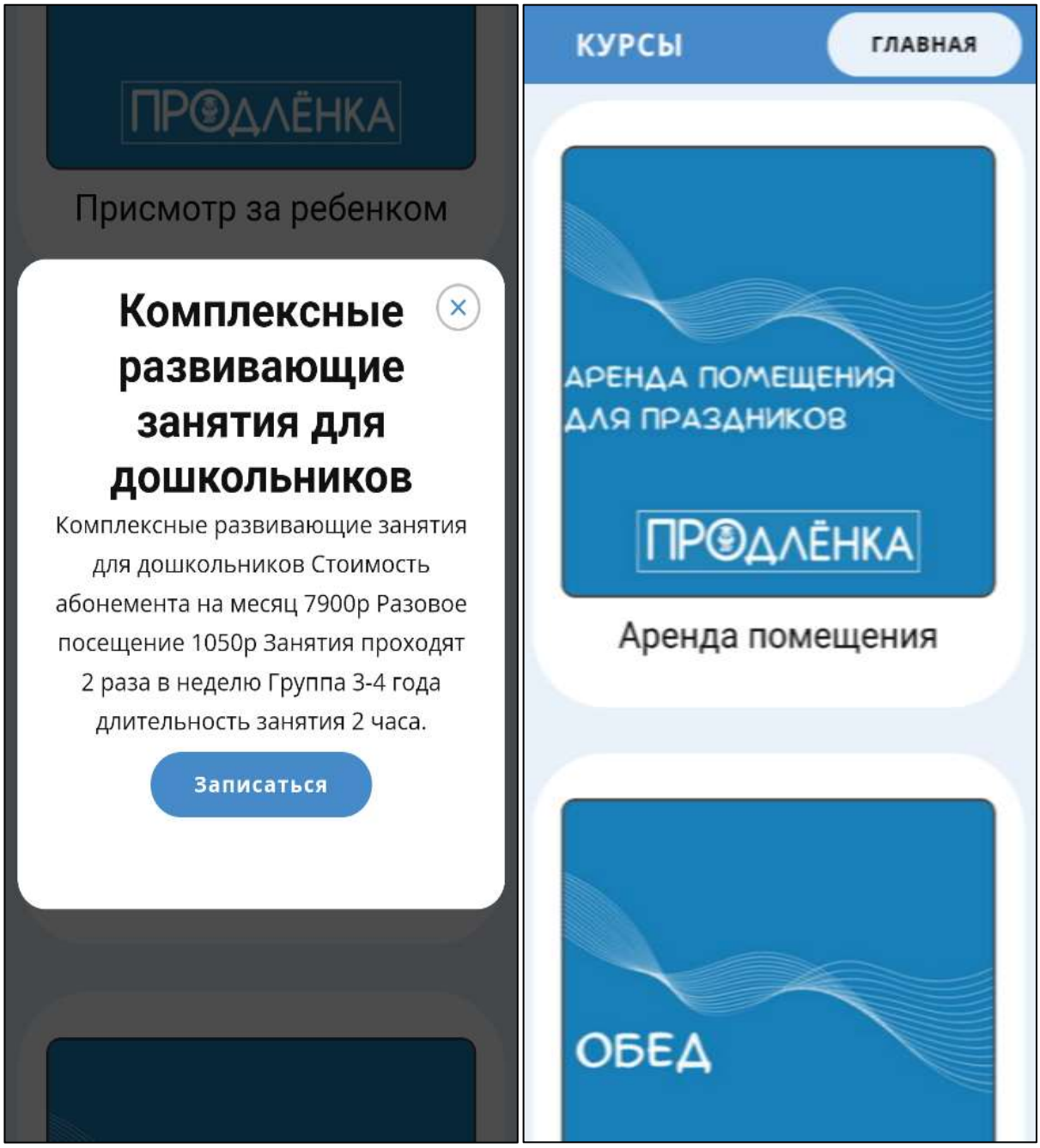

Рисунок 9 – Интерфейс всплывающего информационного окна (слева), интерфейс страницы «Курсы» (справа)

Рисунок 10 – страница «Профиль» с редактированием данных пользователя

Рисунок 11 – Интерфейс страницы «Профиль»

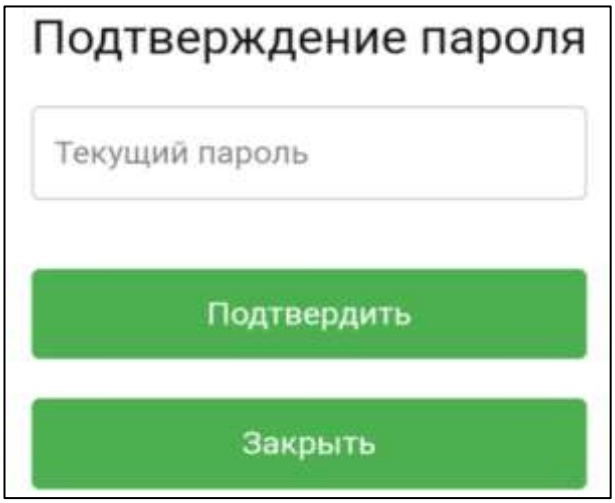

Рисунок 12 – Интерфейс всплывающего окна с подтверждением пароля

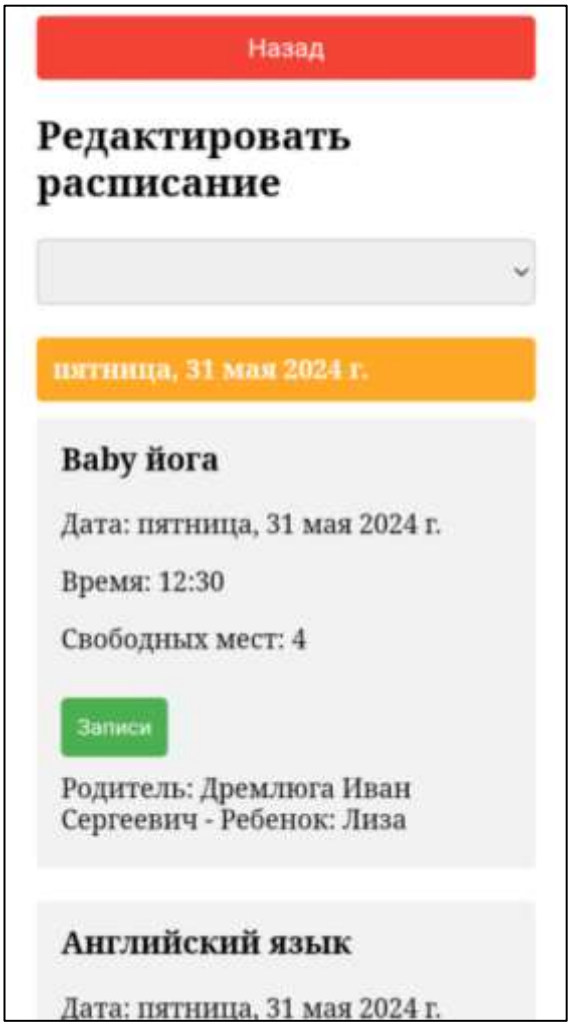

Рисунок 13 – Интерфейс редактирования расписания у администратора

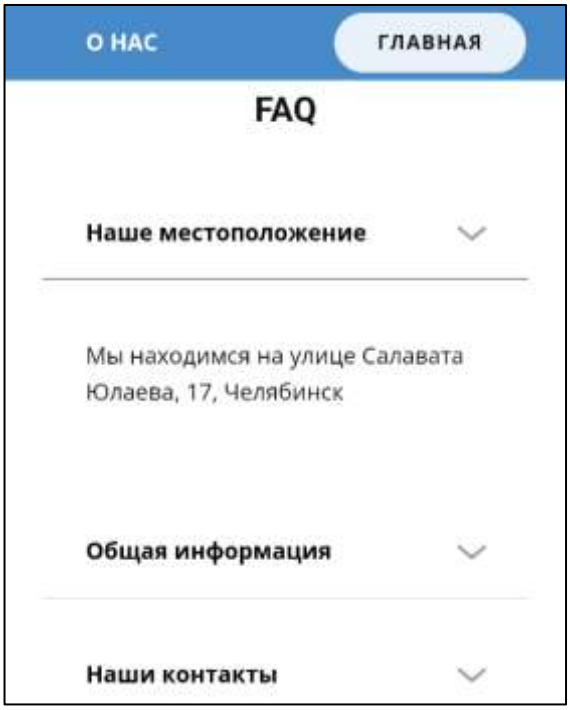

Рисунок 14 – Интерфейс страницы информации

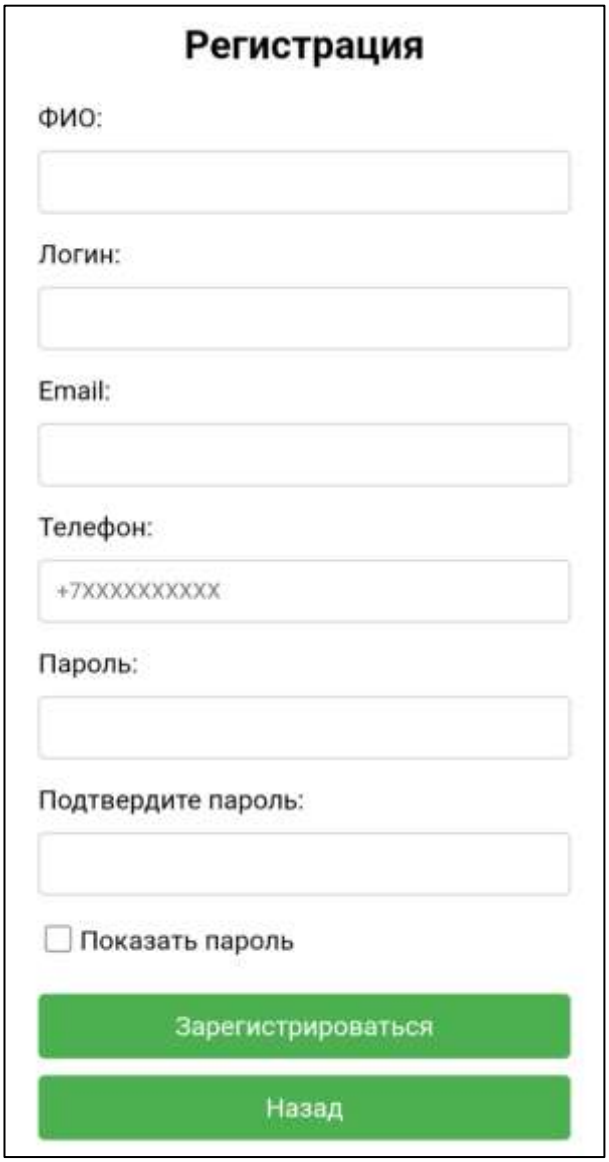

Рисунок 15 – Интерфейс страницы регистрации

Рисунок 16 – Интерфейс информационного окна после регистрации

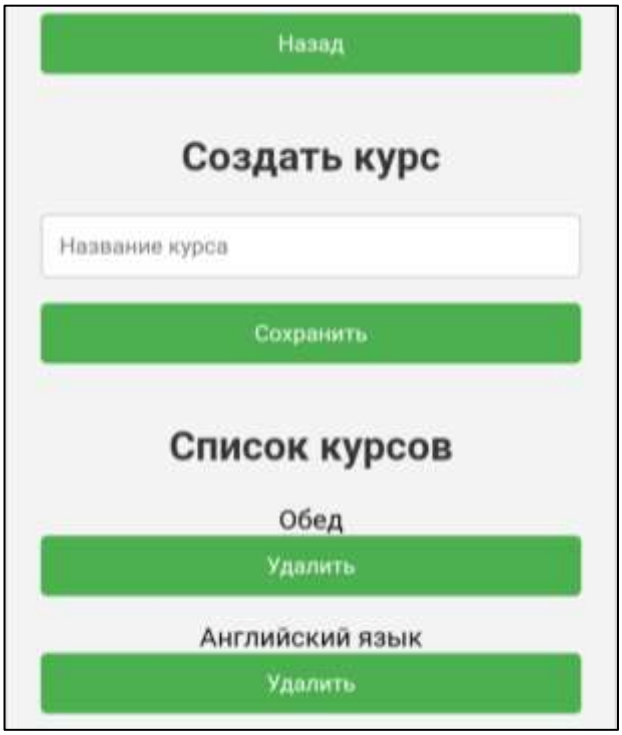

Рисунок 17 – Интерфейс страницы создания курса у администратора

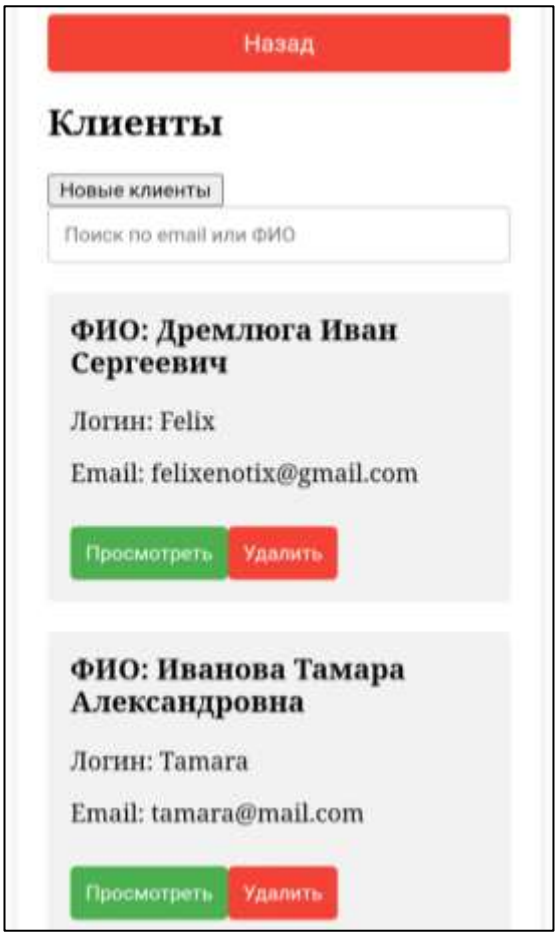

Рисунок 18 – Интерфейс страницы управления клиентами

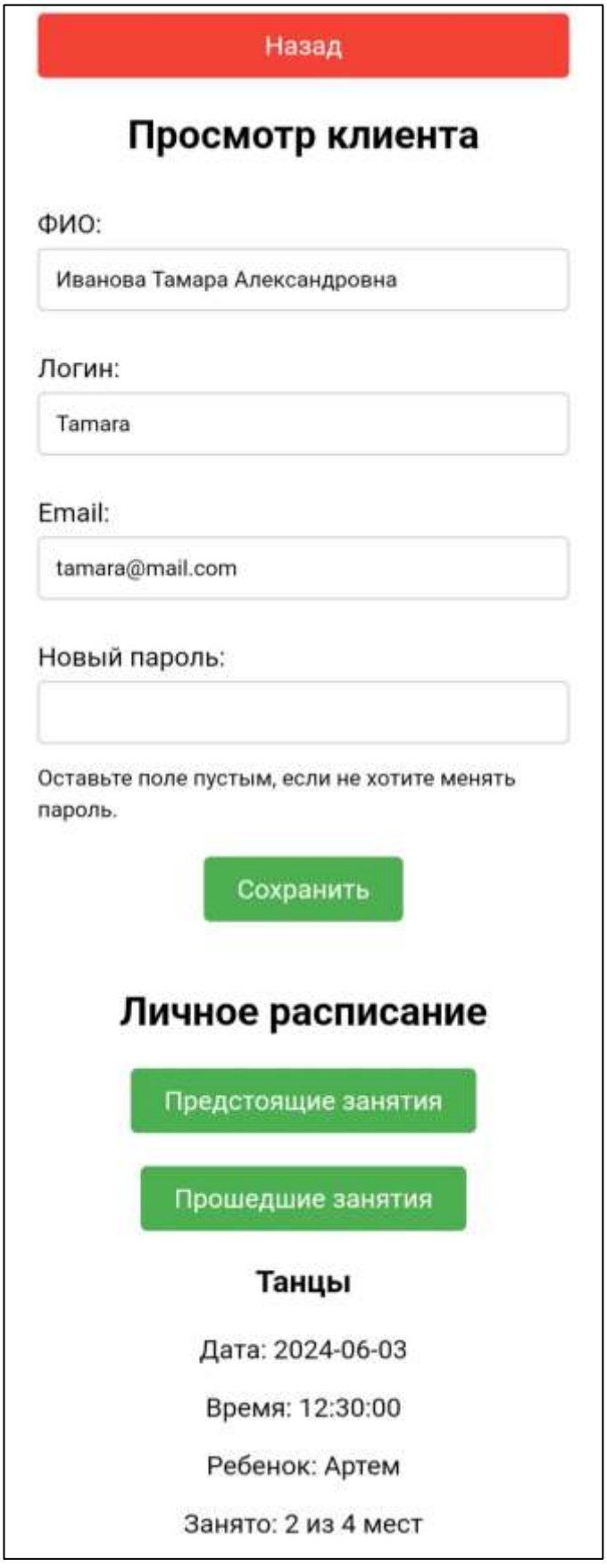

Рисунок 19 – Интерфейс просмотра аккаунта клиента

Приложение Б. Листинг программного кода

Код для основных страниц и скриптов приложения представлен в листингах 1–24.

Листинг 1 – Код страницы авторизации «index.html»

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0,
user-scalable=no">
    <link rel="manifest" href="/manifest.json">
    <title>Вход</title>
    <style>
        .container {
            max-width: 400px;
            margin: 100px auto;
            padding: 20px;
            background-color: #f4f4f4;
            border: 1px solid #ccc;
            border-radius: 8px;
            text-align: center;
        }
        .container h2 {
            color: #333;
        }
        .container button {
            width: 100%;
            margin-bottom: 15px;
            padding: 10px;
            box-sizing: border-box;
            background-color: #4CAF50;
            color: white;
            border: none;
            border-radius: 4px;
            cursor: pointer;
        }
        .container button:hover {
            background-color: #45a049;
        }
    </style>
</head>
<body>
    <div class="container">
        <h2>Авторизация</h2>
        <button onclick="window.location.href='login.php'">Войти</button>
        <button onclick="window.location.href='regis-
ter.html'">Зарегистрироваться</button>
        <button onclick="window.location.href='ad-
min_login.php'">Администратор</button>
    </div>
    <script>
        // Проверка сессии
        document.addEventListener('DOMContentLoaded', function() {
            if (localStorage.getItem('username') && localStor-
age.getItem('user_id')) {
                fetch('auto_login.php', {
                    method: 'POST',
                    headers: {
                        'Content-Type': 'application/json'
```

```
                    },
                    body: JSON.stringify({
                        username: localStorage.getItem('username'),
                        user_id: localStorage.getItem('user_id')
                    })
                }).then(response => response.json())
                  .then(data => {
                      if (data.success) {
                          window.location.href = 'Главная.html';
                      }
                  });
            }
        });
        // Регистрация Service Worker
        if ('serviceWorker' in navigator) {
            navigator.serviceWorker.register('/service-worker.js')
            .then(function(registration) {
                console.log('Service Worker registered with scope:', regis-
tration.scope);
            }).catch(function(error) {
                console.log('Service Worker registration failed:', error);
            });
        }

        // Установка PWA
        let deferredPrompt;
        window.addEventListener('beforeinstallprompt', (e) => {
            e.preventDefault();
            deferredPrompt = e;
            showInstallPromotion();
        });
        function showInstallPromotion() {
            const installButton = document.createElement('button');
            installButton.innerText = 'Установить приложение Продленка';
            installButton.style.position = 'fixed';
            installButton.style.bottom = '20px';
            installButton.style.left = '20px';
            installButton.style.padding = '10px 20px';
            installButton.style.backgroundColor = '#4CAF50';
            installButton.style.color = 'white';
            installButton.style.border = 'none';
            installButton.style.borderRadius = '4px';
            installButton.style.cursor = 'pointer';
            installButton.addEventListener('click', (e) => {
                installButton.style.display = 'none';
                deferredPrompt.prompt();
                deferredPrompt.userChoice.then((choiceResult) => {
                    if (choiceResult.outcome === 'accepted') {
                        console.log('User accepted the install prompt');
                    } else {
                        console.log('Userdismissed the install prompt');
                    }
                    deferredPrompt = null;
                });
            });
            document.body.appendChild(installButton);
        }
    </script>
</body>
</html>
```

Листинг 2 – Код страницы регистрации «register.html»

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="register.css">
    <title>Регистрация</title>
    <style>
        .container {
            max-width: 400px;
            margin: 50px auto;
            background-color: #fff;
            padding: 20px;
            border-radius: 8px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
        }
        h2 {
            margin-bottom: 20px;
        }
        label {
            display: block;
            margin-bottom: 10px;
        }
        input {
            width: 100%;
            padding: 10px;
            margin-bottom: 15px;
            border: 1px solid #ccc;
            border-radius: 4px;
            box-sizing: border-box;
        }
        button {
            display: block;
            width: 100%;
            padding: 10px;
            background-color: #4CAF50;
            color: #fff;
            border: none;
            border-radius: 4px;
            cursor: pointer;
            font-size: 16px;
            margin-top: 10px;
        }
        button:hover {
            background-color: #45a049;
        }
        .error-message {
            color: red;
            margin-top: 10px;
        }
        .notification {
            color: green;
            margin-top: 10px;
        }
        .show-password {
            display: flex;
            align-items: center;
        }
        .show-password input[type="checkbox"] {
            margin-right: 5px;
```

```
        }
    </style>
</head>
<body>
    <div class="container">
        <h2>Регистрация</h2>
        <form id="register-form">
            <label for="full-name">ФИО:</label>
            <input type="text" id="full-name" name="full_name"
maxlength="70" required>
            <label for="username">Логин:</label>
            <input type="text" id="username" name="username" required>
            <label for="email">Email:</label>
            <input type="email" id="email" name="email" required>
            <label for="phone-number">Телефон:</label>
            <input type="tel" id="phone-number" name="phone_number" pat-
tern="^\+7\d{10}$" required placeholder="+7XXXXXXXXXX" maxlength="12">
            <label for="password">Пароль:</label>
            <input type="password" id="password" name="password" required>
            <label for="confirm-password">Подтвердите пароль:</label>
            <input type="password" id="confirm-password" name="con-
firm_password" required>
            <div class="show-password">
                <input type="checkbox" id="show-password">
                <label for="show-password">Показать пароль</label>
            </div>
            <button type="submit">Зарегистрироваться</button>
            <div class="error-message" id="error-message"></div>
            <div class="notification" id="notification"></div>
        </form>
        <button onclick="window.location.href='index.html'">Назад</button>
    </div>
    <script src="jquery-1.9.1.min.js"></script>
    <script>
        $(document).ready(function() {
            $('#show-password').on('change', function() {
                const passwordField = $('#password');
                const confirmPasswordField = $('#confirm-password');
                if ($(this).is(':checked')) {
                    passwordField.attr('type', 'text');
                    confirmPasswordField.attr('type', 'text');
                } else {
                    passwordField.attr('type', 'password');
                }
            });
            $('#register-form').on('submit', function(e) {
                e.preventDefault();
                const fullName = $('#full-name').val().trim();
                const username = $('#username').val().trim();
                const email = $('#email').val().trim();
                const password = $('#password').val();
                const confirmPassword = $('#confirm-password').val();
                const phoneNumber = $('#phone-number').val().trim();
                if (password !== confirmPassword) {
                    $('#error-message').text('Пароли не совпадают.');
                    return;
                }
                if (!/^\+7\d{10}$/.test(phoneNumber)) {
                    $('#error-message').text('Телефон должен быть в формате
+7XXXXXXXXXX.');
                    return;
                }
```

```
                $.ajax({
                    url: 'register.php',
                    type: 'POST',
                    data: JSON.stringify({
                        full_name: fullName,
                        username: username,
                        email: email,
                        password: password,
                        phone_number: phoneNumber
                    }),
                    contentType: 'application/json',
                    success: function(response) {
                        const res = JSON.parse(response);
                        if (res.status === 'success') {
                            window.location.href = 'registration_confirma-
tion.html';
                        } else {
                            $('#error-message').text(res.message);
                        }
                    },
                    error: function() {
                        $('#error-message').text('Ошибка при регистра-
ции.');
                    }
                });
            });
        });
    </script>
</body>
</html>
```

Листинг 3 – Код страницы профиля «Профиль.php»

```
<?php
session_start();

$servername = "localhost";
$username = "felixeob_courses";
$password = "Fel22446688";
$dbname = "felixeob_courses";

$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
if (!isset($_SESSION['username']) && !isset($_SESSION['user_id'])) {
    echo "<script>
        if (localStorage.getItem('username') && localStor-
age.getItem('user_id')) {
            fetch('auto_login.php', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json'
                },
                body: JSON.stringify({
                    username: localStorage.getItem('username'),
                    user_id: localStorage.getItem('user_id')
                })
            }).then(response => response.json())
              .then(data => {
```

```
                if (data.success) {
                    location.reload();
                } else {
                    window.location.href = 'login.php';
                }
            });
        } else {
            window.location.href = 'login.php';
        }
    </script>";
    exit();
}
$user_id = $_SESSION['user_id'];
$sql_check = "SELECT is_logged_in FROM users WHERE id = ?";
$stmt_check = $conn->prepare($sql_check);
$stmt_check->bind_param("i", $user_id);
$stmt_check->execute();
$stmt_check->bind_result($is_logged_in);
$stmt_check->fetch();
$stmt_check->close();

if ($is_logged_in == 0) {
    header('Location: login.php');
    exit();
}
$conn->close();
?>
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="Профиль.css">
    <title>Профиль</title>
</head>
<body>
    <button class="blue-button" onclick="window.loca-
tion.href='Главная.html'">Главная</button>
    <div class="container">
        <div class="btn-group">
            <button onclick="window.location.href='sched-
ule.php'">Записаться на курс</button>
            <button onclick="window.location.href='my_courses.php'">Мои
курсы</button>
        </div>
        <h2>Профиль</h2>
        <form id="profile-form" method="post">
            <div class="form-group">
                <label for="full-name">ФИО:</label>
                <input type="text" id="full-name" name="full_name"
maxlength="50" required pattern="^[А-Яа-яЁё\s]+$">
            </div>
            <div class="form-group">
                <label for="username">Логин:</label>
                <input type="text" id="username" name="username" required
pattern="[A-Za-z]{2,15}">
            </div>
            <div class="form-group">
                <label for="email">Email:</label>
                <input type="email" id="email" name="email" required>
            </div>
            <div class="form-group">
```

```
                <label for="phone-number">Номер телефона:</label>
                <input type="text" id="phone-number" name="phone_number"
pattern="^\+7\d{10}$" required placeholder="+7XXXXXXXXXX" maxlength="12">
            </div>
            <div class="form-group">
                <label for="password">Изменить пароль:</label>
                <input type="password" id="password" name="password" re-
quired pattern="(?=.*\d)(?=.*[a-z])(?=.*[A-Z]).{6,20}">
                <small>Пароль должен содержать прописную и строчную букву,
цифру и быть длиной от 6 до 20 символов.</small>
            </div>
            <button type="button" onclick="showPassword-
Prompt()">Сохранить</button>
            <div id="profile-message" class="notification"></div>
        </form>
        <h2>Дети</h2>
        <div id="children-list"></div>
        <form id="child-form" method="post">
            <div class="form-group">
                <label for="child-name">Имя ребенка:</label>
                <input type="text" id="child-name" name="child_name"
maxlength="35" required pattern="^[А-Яа-яЁё\s]+$">
            </div>
            <div class="form-group">
                <label for="child-age">Возраст:</label>
                <input type="number" id="child-age" name="child_age"
min="3" max="20" required>
            </div>
            <div class="form-group">
                <label for="child-gender">Пол:</label>
                <select id="child-gender" name="child_gender" required>
                    <option value="М">Мальчик</option>
                    <option value="Ж">Девочка</option>
                </select>
            </div>
            <button type="button" onclick="addChild()">Добавить ре-
бенка</button>
            <div id="child-message" class="notification"></div>
        </form>
        <button class="logout-btn red" onclick="window.location.href='log-
out.php'">Выйти</button>
    </div>
    <div id="overlay" class="overlay" style="display: none;">
        <div class="popup">
            <h3>Подтверждение пароля</h3>
            <input type="password" id="current-password" place-
holder="Текущий пароль" maxlength="20" pattern="[A-Za-z]{6,20}" required>
            <button onclick="confirmPassword()">Подтвердить</button>
            <button onclick="closeOverlay()">Закрыть</button>
        </div>
    </div>
    <div id="delete-overlay" class="overlay" style="display: none;">
        <div class="popup">
            <h3>Подтверждение пароля для удаления</h3>
            <input type="password" id="delete-password" place-
holder="Текущий пароль" maxlength="20" pattern="[A-Za-z]{6,20}" required>
            <button onclick="confirmDelete()">Подтвердить</button>
            <button onclick="closeDeleteOverlay()">Закрыть</button>
        </div>
    </div>
    <script src="jquery-1.9.1.min.js"></script>
    <script>
```

```
        let currentDeleteId;
        $(document).ready(function(){
            // Загрузка профиля
            $.ajax({
                url: 'get_profile.php',
                type: 'GET',
                success: function(data) {
                    var profile = JSON.parse(data);
                    $('#full-name').val(profile.full_name);
                    $('#username').val(profile.username);
                    $('#email').val(profile.email);
                    $('#phone-number').val(profile.phone_number);
                },
                error: function() {
                    $('#profile-message').addClass('error').text('Ошибка
при загрузке профиля.');
                }
            });
            // Загрузка списка детей
            loadChildren();
            // Удаление ребенка
            $(document).on('click', '.delete-btn', function() {
                currentDeleteId = $(this).data('id');
                $('#delete-overlay').show();
            });
        });
        function showPasswordPrompt() {
            $('#overlay').show();
        }
        function closeOverlay() {
            $('#overlay').hide();
        }
        function closeDeleteOverlay() {
            $('#delete-overlay').hide();
        }
        function confirmPassword() {
            var currentPassword = $('#current-password').val();
            if (!currentPassword) {
                alert('Введите текущий пароль.');
                return;
            }
            var profileData = {
                full_name: $('#full-name').val(),
                username: $('#username').val(),
                email: $('#email').val(),
                phone_number: $('#phone-number').val(),
                password: $('#password').val(),
                current_password: currentPassword
            };
            $.ajax({
                url: 'save_profile.php',
                type: 'POST',
                data: JSON.stringify(profileData),
                contentType: 'application/json',
                success: function(response) {
                    $('#profile-message').removeClass('error').text(re-
sponse);
                    $('#overlay').hide();
                },
                error: function() {
                    $('#profile-message').addClass('error').text('Ошибка
при сохранении профиля.');
```

```
                }
            });
        }
        function loadChildren() {
            $.ajax({
                url: 'get_children.php',
                type: 'GET',
                success: function(data) {
                    var children = JSON.parse(data);
                    $('#children-list').empty();
                    children.forEach(function(child) {
                        $('#children-list').append(
                            '<div class="form-group">' +
                            '<label>Имя: ' + child.name + '</label>' +
                            '<label>Возраст: ' + child.age + '</label>' +
                            '<label>Пол: ' + (child.gender === 'M' ?
'Мальчик' : 'Девочка') + '</label>' +
                            '<button class="delete-btn red" data-id="' +
child.id + '">Удалить</button>' +
                            '</div>'
                        );
                    });
                },
                error: function() {
                    $('#child-message').addClass('error').text('Ошибка при
загрузке списка детей.');
                }
            });
        }
        function addChild() {
            var childData = {
                name: $('#child-name').val(),
                age: $('#child-age').val(),
                gender: $('#child-gender').val()
            };
            if (!childData.name.match(/^[А-Яа-яЁё\s]+$/)) {
                $('#child-message').addClass('error').text('Имя должно со-
держать только русские буквы.');
                return;
            }
            $.ajax({
                url: 'save_child.php',
                type: 'POST',
                data: JSON.stringify(childData),
                contentType: 'application/json',
                success: function(response) {
                    $('#child-message').removeClass('error').text(re-
sponse);
                    loadChildren();
                },
                error: function() {
                    $('#child-message').addClass('error').text('Ошибка при
добавлении ребенка.');
                }
            });
        }
        function confirmDelete() {
            var deletePassword = $('#delete-password').val();
            if (!deletePassword) {
                alert('Введите текущий пароль.');
                return;
            }
```

Окончание листинга 3 приложения Б

```
            $.ajax({
                url: 'delete_child.php',
                type: 'POST',
                data: JSON.stringify({ id: currentDeleteId, password:
deletePassword }),
                contentType: 'application/json',
                success: function(response) {
                    $('#child-message').removeClass('error').text(re-
sponse);
                    loadChildren();
                    $('#delete-overlay').hide();
                },
                error: function() {
                    $('#child-message').addClass('error').text('Ошибка при
удалении ребенка.');
                }
            });
        }
    </script>
</body>
</html>
```

Листинг 4 – Код страницы расписания «schedule.php»

```
<?php
session_start();
$servername = "localhost";
$username = "felixeob_courses";
$password = "Fel22446688";
$dbname = "felixeob_courses";
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
$user_id = $_SESSION['user_id'];
// Установка локали для русского языка
setlocale(LC_TIME, 'ru_RU.UTF-8');
// Fetch courses with schedule
$sql_courses = "SELECT DISTINCT c.id, c.name FROM courses c JOIN schedule s
ON c.id = s.course_id";
$result_courses = $conn->query($sql_courses);
$courses = [];
if ($result_courses->num_rows > 0) {
    while ($row = $result_courses->fetch_assoc()) {
        $courses[] = $row;
    }
}
$course_id = isset($_GET['course_id']) ? intval($_GET['course_id']) : 0;
$sql_schedule = "SELECT s.id, c.name AS course_name, s.date, s.time,
s.available_slots
                 FROM schedule s
                 JOIN courses c ON s.course_id = c.id
                 WHERE s.date >= CURDATE()";
if ($course_id > 0) {
    $sql_schedule .= " AND s.course_id = $course_id";
}
$sql_schedule .= " ORDER BY s.date, s.time";
$result_schedule = $conn->query($sql_schedule);
$schedule = [];
if ($result_schedule->num_rows > 0) {
```

```
    while ($row = $result_schedule->fetch_assoc()) {
        $schedule[] = $row;
    }
}
$sql_children = "SELECT id, name FROM children WHERE user_id = $user_id";
$result_children = $conn->query($sql_children);
$children = [];
if ($result_children->num_rows > 0) {
    while ($row = $result_children->fetch_assoc()) {
        $children[] = $row;
    }
} else {
    echo "Ошибка при получении данных о детях: " . $conn->error;
}
$conn->close();
?>
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="schedule.css">
    <title>Запись на курс</title>
</head>
<body>
    <div class="container">
        <button class="back-button" onclick="window.loca-
tion.href='Профиль.php'">Назад</button>
        <h1>Запись на курс</h1>
        <div>
            <label for="course-filter">Выберите курс:</label>
            <select id="course-filter" name="course-filter">
                <option value="">Все курсы</option>
                <?php foreach ($courses as $course): ?>
                    <option value="<?= $course['id'] ?>"><?=
$course['name'] ?></option>
                <?php endforeach; ?>
            </select>
            <button onclick="applyFilters()">Применить</button>
        </div>
        <div id="schedule-container">
            <?php
            $lastDate = '';
            foreach ($schedule as $item):
                $currentDate = strftime('%Y-%m-%d', strto-
time($item['date']));
                if ($lastDate !== $currentDate):
                    $formattedDate = strftime('%d %B, %A', strto-
time($item['date']));
                    echo "<div class='date-header'>{$formattedDate}</div>";
                    $lastDate = $currentDate;
                endif;
            ?>
                <div class="schedule-item">
                    <h2><?= $item['course_name'] ?></h2>
                    <p>Дата: <?= strftime('%d', strtotime($item['date']))
?> <?= strftime('%B', strtotime($item['date'])) ?>, <?= strftime('%A',
strtotime($item['date'])) ?></p>
                    <p>Время: <?= date('H:i', strtotime($item['time']))
?></p>
                    <p>Свободных мест: <?= $item['available_slots'] ?></p>
```

```
                    <label for="child-select-<?= $item['id'] ?>">Выберите
ребенка:</label>
                    <select id="child-select-<?= $item['id'] ?>"
class="child-select" data-schedule-id="<?= $item['id'] ?>">
                        <option value="">Выберите ребенка</option>
                        <?php foreach ($children as $child): ?>
                            <option value="<?= $child['id'] ?>"><?=
$child['name'] ?></option>
                        <?php endforeach; ?>
                    </select>
                    <button class="enroll-button" data-schedule-id="<?=
$item['id'] ?>">Записаться</button>
                </div>
            <?php endforeach; ?>
        </div>
    </div>
    <script src="jquery-1.9.1.min.js"></script>
    <script>
        function applyFilters() {
            var courseId = $('#course-filter').val();
            window.location.href = 'schedule.php?course_id=' + courseId;
        }
        $(document).ready(function(){
            $('.enroll-button').on('click', function() {
                var scheduleId = $(this).data('schedule-id');
                var childId = $('#child-select-' + scheduleId).val();
                if (!childId) {
                    alert('Пожалуйста, выберите ребенка.');
                    return;
                }
                $.ajax({
                    url: 'enroll.php',
                    type: 'POST',
                    data: {
                        schedule_id: scheduleId,
                        child_id: childId
                    },
                    success: function(response) {
                        alert(response);
                        location.reload();
                    },
                    error: function() {
                        alert('Ошибка при записи на занятие.');
                    }
                });
            });
        });
    </script>
</body>
</html>
```

Листинг 5 – Код страницы моих курсов «my_courses.php»

```
<?php
session_start();
$servername = "localhost";
$username = "felixeob_courses";
$password = "Fel22446688";
$dbname = "felixeob_courses";
$conn = new mysqli($servername, $username, $password, $dbname);
```

```
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
$user_id = $_SESSION['user_id'];
setlocale(LC_TIME, 'ru_RU.UTF-8');
// Fetch user enrollments
$sql_enrollments = "SELECT c.name AS course_name, s.date, s.time, ch.name
AS child_name, e.id AS enrollment_id
                    FROM enrollments e
                    JOIN schedule s ON e.schedule_id = s.id
                    JOIN courses c ON s.course_id = c.id
                    JOIN children ch ON e.child_id = ch.id
                    WHERE e.user_id = $user_id
                    ORDER BY s.date, s.time";
$result_enrollments = $conn->query($sql_enrollments);
$enrollments = [];
if ($result_enrollments->num_rows > 0) {
    while ($row = $result_enrollments->fetch_assoc()) {
        $enrollments[] = $row;
    }
}
$conn->close();
?>
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="nicepage.css" media="screen">
    <link rel="stylesheet" href="Расписание.css" media="screen">
    <title>Мои курсы</title>
</head>
<body class="u-body">
    <section class="u-clearfix u-section-1">
        <div class="u-clearfix u-sheet u-valign-middle-lg u-valign-middle-
md u-valign-middle-sm u-valign-middle-xl u-sheet-1">
            <div class="u-clearfix u-custom-html u-expanded-width u-custom-
html-1">
                <div class="container">
                    <h1>Мои курсы</h1>
                    <button class="back-button" onclick="window.loca-
tion.href='Профиль.php'">Назад</button>
                    <?php
                    if (count($enrollments) > 0) {
                        foreach ($enrollments as $enrollment) {
                            echo "<div class='schedule-item'>";
                            echo "<div>";
                            echo "<h2>" . $enrollment['course_name'] .
"</h2>";
                            echo "<p>Дата: " . strftime('%d', strto-
time($enrollment['date'])) . " " . strftime('%B', strtotime($enroll-
ment['date'])) . ", " . strftime('%A', strtotime($enrollment['date'])) .
"</p>";
                            echo "<p>Время: " . date('H:i', strtotime($en-
rollment['time'])) . "</p>";
                            echo "<p>Ребенок: " . $enrollment['child_name']
. "</p>";
                            echo "</div>";
                            echo "<button class='delete-btn' onclick='show-
PasswordPrompt(" . $enrollment['enrollment_id'] . ")'></button>";
                            echo "</div>";
                        }
```

Окончание листинга 5 приложения Б

```
                    } else {
                        echo "<p>У вас нет записей на курсы.</p>";
                    }
                    ?>
                </div>
            </div>
        </div>
    </section>
    <div id="overlay" class="overlay" style="display: none;">
        <div class="popup">
            <h3>Подтверждение пароля</h3>
            <input type="password" id="current-password" place-
holder="Текущий пароль" required pattern="[A-Za-z0-9]{6,20}">
            <button_back onclick="confirmDeletion()">Подтвердить</but-
ton_back>
            <button_back onclick="closePasswordPrompt()">Закрыть</but-
ton_back>
        </div>
    </div>
    <script src="jquery-1.9.1.min.js"></script>
    <script>
        var enrollmentIdToDelete;
        function showPasswordPrompt(enrollmentId) {
            enrollmentIdToDelete = enrollmentId;
            $('#overlay').show();
        }
        function closePasswordPrompt() {
            $('#overlay').hide();
        }
        function confirmDeletion() {
            var password = $('#current-password').val();
            if (!password) {
                alert('Введите текущий пароль.');
                return;
            }
            $.ajax({
                url: 'delete_enrollment.php',
                type: 'POST',
                data: JSON.stringify({ id: enrollmentIdToDelete, password:
password }),
                contentType: 'application/json',
                success: function(response) {
                    alert(response);
                    location.reload();
                },
                error: function() {
                    alert('Ошибка при удалении записи.');
                }
            });
        }
    </script>
</body>
</html>
```

Листинг 6 – Код страницы редактирования расписания «edit_schedule.php»

```
<?php
session_start();
?>
<!DOCTYPE html>
```

```
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Редактировать расписание</title>
</head>
<body>
    <div class="container">
        <button class="back-button" onclick="window.location.href='ad-
min_dashboard.html'">Назад</button>
        <h2>Редактировать расписание</h2>
        <input type="date" id="filter-date" class="filter-input" place-
holder="Фильтр по дате" oninput="filterSchedules()">
        <div id="schedule-list"></div>
    </div>
    <script src="jquery-1.9.1.min.js"></script>
    <script>
        var schedules = <?php echo json_encode($schedules); ?>;
        $(document).ready(function() {
            renderSchedules(schedules);
        });
        function renderSchedules(schedules) {
            var scheduleList = $('#schedule-list');
            scheduleList.empty();
            var lastDate = '';
            schedules.forEach(function(schedule) {
                var date = new Date(schedule.date);
                var formattedDate = date.toLocaleDateString('ru-RU', {
year: 'numeric', month: 'long', day: 'numeric', weekday: 'long' });
                if (lastDate !== schedule.date) {
                    scheduleList.append('<div class="date-header">' + for-
mattedDate + '</div>');
                    lastDate = schedule.date;
                }
                var scheduleCard = '<div class="schedule-card" data-date="'
+ schedule.date + '">' +
                    '<h3>' + schedule.course_name + '</h3>' +
                    '<p>Дата: ' + formattedDate + '</p>' +
                    '<p>Время: ' + schedule.time.substring(0, 5) + '</p>' +
                    '<p>Свободных мест: ' + schedule.available_slots +
'</p>' +
                    '<button onclick="showEnrollments(' + schedule.id +
')">Записи</button>' +
                    '<div class="enrollments-list" id="enrollments-' +
schedule.id + '"></div>' +
                    '</div>';
                scheduleList.append(scheduleCard);
            });
        }
        function filterSchedules() {
            var filterDate = $('#filter-date').val();
            if (filterDate) {
                var filteredSchedules = schedules.filter(function(schedule)
{
                    return schedule.date === filterDate;
                });
                renderSchedules(filteredSchedules);
            } else {
                renderSchedules(schedules);
            }
        }
        function showEnrollments(scheduleId) {
```

Окончание листинга 6 приложения Б

```
            $.ajax({
                url: 'get_enrollments.php',
                type: 'POST',
                data: JSON.stringify({ schedule_id: scheduleId }),
                contentType: 'application/json',
                success: function(data) {
                    var enrollments = JSON.parse(data);
                    var enrollmentsList = $('#enrollments-' + scheduleId);
                    enrollmentsList.empty();
                    if (enrollments.length > 0) {
                        enrollments.forEach(function(enrollment) {
                            enrollmentsList.append('<p>Родитель: ' + en-
rollment.parent_name + ' - Ребенок: ' + enrollment.child_name + '</p>');
                        });
                    } else {
                        enrollmentsList.append('<p>Записей нет.</p>');
                    }
                    enrollmentsList.toggle();
                },
                error: function() {
                    alert('Ошибка при загрузке записей.');
                }
            });
        }
    </script>
</body>
</html>
```

Листинг 7 – Код страницы администратора «admin_dashboard.html»

```
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Панель администратора</title>
</head>
<body>
    <div class="container">
        <h1>Панель администратора</h1>
        <button class="green-button" onclick="window.location.href='cre-
ate_course.html'">Создать курс</button>
        <button class="green-button" onclick="window.location.href='cre-
ate_schedule.php'">Создать расписание</button>
        <button class="green-button" onclick="window.loca-
tion.href='edit_schedule.php'">Редактировать расписание</button>
        <button class="green-button" onclick="window.location.href='cli-
ents.html'">Клиенты</button>
        <button class="red-button" onclick="window.location.href='log-
out.php'">Выйти</button>
    </div>
</body>
</html>
```

Листинг 8 – Код страницы входа администратора «admin_login.php»

```
<?php
session_start();
$servername = "localhost";
$username = "felixeob_courses";
$password = "Fel22446688";
$dbname = "felixeob_courses";
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
function validateEmail($email) {
    return filter_var($email, FILTER_VALIDATE_EMAIL) && preg_match("/^[a-
zA-Z0-9@.]{1,45}$/", $email);
}
function validatePassword($password) {
    return preg_match("/^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)[A-Za-z\d]{6,20}$/",
$password);
}
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $email = $_POST['admin_email'];
    $password = $_POST['admin_password'];
    if (!validateEmail($email)) {
        $error_message = "Некорректный email.";
    } elseif (!validatePassword($password)) {
        $error_message = "Некорректный пароль.";
    } else {
        $sql = "SELECT * FROM users WHERE email=?";
        $stmt = $conn->prepare($sql);
        $stmt->bind_param("s", $email);
        $stmt->execute();
        $result = $stmt->get_result();
        if ($result->num_rows > 0) {
            $row = $result->fetch_assoc();
            if (password_verify($password, $row['password'])) {
                $admin_check_sql = "SELECT * FROM administrators WHERE
email=?";
                $admin_stmt = $conn->prepare($admin_check_sql);
                $admin_stmt->bind_param("s", $email);
                $admin_stmt->execute();
                $admin_result = $admin_stmt->get_result();
                if ($admin_result->num_rows > 0) {
                    $_SESSION['admin_id'] = $row['id'];
                    $_SESSION['admin_email'] = $row['email'];
                    header('Location: admin_dashboard.html');
                    exit();
                } else {
                    $error_message = "У вас нет прав администратора.";
                }
                $admin_stmt->close();
            } else {
                $error_message = "Неверный email или пароль.";
            }
        } else {
            $error_message = "Пользователь не найден.";
        }
        $stmt->close();
    }
}
$conn->close();
?>
```

Окончание листинга 8 приложения Б

```
<!DOCTYPE html>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Вход администратора</title>
</head>
<body>
    <div class="container">
        <h2>Вход администратора</h2>
        <form id="admin-login-form" method="post">
            <div class="form-group">
                <label for="admin-email">Email:</label>
                <input type="email" id="admin-email" name="admin_email" re-
quired pattern="[a-zA-Z0-9@.]{1,45}" title="Email должен содержать только
английские буквы и цифры, до 45 символов.">
            </div>
            <div class="form-group">
                <label for="admin-password">Пароль:</label>
                <input type="password" id="admin-password" name="ad-
min_password" required pattern="(?=.*[a-z])(?=.*[A-Z])(?=.*\d)[A-Za-
z\d]{6,20}" title="Пароль должен содержать от 6 до 20 символов, включать
заглавные и строчные буквы, а также цифры.">
            </div>
            <button type="submit">Войти</button>
            <button type="button" class="back-button" onclick="window.loca-
tion.href='index.html'">Назад</button>
            <?php if (isset($error_message)): ?>
                <div class="error-message"><?= $error_message ?></div>
            <?php endif; ?>
        </form>
    </div>
</body>
</html>
```

Листинг 9 – Код страницы входа «login.php»

```
<?php
session_start();
$servername = "localhost";
$username = "felixeob_courses";
$password = "Fel22446688";
$dbname = "felixeob_courses";
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
$message = "";
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    $username = trim($_POST['username']);
    $password = trim($_POST['password']);
    if (empty($username) || empty($password)) {
        $message = "Логин и пароль не могут быть пустыми.";
    } else {
        $sql = "SELECT * FROM users WHERE username=?";
        $stmt = $conn->prepare($sql);
        $stmt->bind_param("s", $username);
        $stmt->execute();
        $result = $stmt->get_result();
        if ($result->num_rows > 0) {
            $user = $result->fetch_assoc();
```

```
            if (password_verify($password, $user['password'])) {
                $_SESSION['username'] = $username;
                $_SESSION['user_id'] = $user['id'];
                $sql_update = "UPDATE users SET is_logged_in = 1 WHERE id =
?";
                $stmt_update = $conn->prepare($sql_update);
                $stmt_update->bind_param("i", $user['id']);
                $stmt_update->execute();
                // Сохранение информации о сессии в LocalStorage и перена-
правление
                echo "<script>
                    localStorage.setItem('username', '$username');
                    localStorage.setItem('user_id', '{$user['id']}');
                    window.location.href = 'index.html';
                </script>";
                exit();
            } else {
                $message = "Неверный пароль";
            }
        } else {
            $message = "Пользователь не найден";
        }
    }
}
$conn->close();
?>
<!DOCTYPE html>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="nicepage.css" media="screen">
    <link rel="stylesheet" href="login.css" media="screen">
    <title>Вход</title>
</head>
<body>
    <div class="container">
        <h2>Вход</h2>
        <form id="loginForm" method="POST" action="login.php">
            <label for="username">Логин:</label>
            <input type="text" id="username" name="username" required>
            <label for="password">Пароль:</label>
            <input type="password" id="password" name="password" required>
            <button type="submit">Войти</button>
            <div id="login-message" class="notification"><?php echo html-
specialchars($message); ?></div>
        </form>
        <button class="back-button" onclick="window.location.href='in-
dex.html'">Назад</button>
    </div>
    <script>
        document.addEventListener('DOMContentLoaded', function() {
            if (localStorage.getItem('username') && localStor-
age.getItem('user_id')) {
                fetch('auto_login.php', {
                    method: 'POST',
                    headers: {
                        'Content-Type': 'application/json'
                    },
                    body: JSON.stringify({
                        username: localStorage.getItem('username'),
                        user_id: localStorage.getItem('user_id')
                    })
```

```
                }).then(response => response.json())
                  .then(data => {
                      if (data.success) {
                          window.location.href = 'Главная.html';
                      }
                  });
            }
        });
        window.addEventListener('beforeunload', function() {
            localStorage.removeItem('username');
            localStorage.removeItem('user_id');
        });
    </script>
</body>
</html>
```

Листинг 10 – Код страницы автоматического входа «auto_login.php»

```
<?php
session_start();
$servername = "localhost";
$username = "felixeob_courses";
$password = "Fel22446688";
$dbname = "felixeob_courses";
// Подключение к базе данных
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
$data = json_decode(file_get_contents("php://input"), true);
$username = $data['username'];
$user_id = $data['user_id'];
$sql = "SELECT * FROM users WHERE username = ? AND id = ? AND is_logged_in
= 1";
$stmt = $conn->prepare($sql);
$stmt->bind_param("si", $username, $user_id);
$stmt->execute();
$result = $stmt->get_result();
$response = [];
if ($result->num_rows > 0) {
    $user = $result->fetch_assoc();
    $_SESSION['username'] = $username;
    $_SESSION['user_id'] = $user_id;
    $response['success'] = true;
} else {
    $response['success'] = false;
}
echo json_encode($response);
$conn->close();
?>
```

Листинг 11 – Код страницы удаления записи «delete_enrollment.php»

```
<?php
session_start();
$servername = "localhost";
$username = "felixeob_courses";
$password = "Fel22446688";
$dbname = "felixeob_courses";
```

```
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
if (!isset($_SESSION['username']) && !isset($_SESSION['user_id'])) {
    echo "<script>
        if (localStorage.getItem('username') && localStor-
age.getItem('user_id')) {
            fetch('auto_login.php', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json'
                },
                body: JSON.stringify({
                    username: localStorage.getItem('username'),
                    user_id: localStorage.getItem('user_id')
                })
            }).then(response => response.json())
              .then(data => {
                  if (data.success) {
                      location.reload();
                  } else {
                      window.location.href = 'login.php';
                  }
              });
        } else {
            window.location.href = 'login.php';
        }
    </script>";
    exit();
}
$servername = "localhost";
$username = "felixeob_courses";
$password = "Fel22446688";
$dbname = "felixeob_courses";
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
$data = json_decode(file_get_contents("php://input"), true);
$enrollment_id = $data['id'];
$input_password = $data['password'];
$user_id = $_SESSION['user_id'];
$sql = "SELECT password FROM users WHERE id='$user_id'";
$result = $conn->query($sql);
if ($result->num_rows > 0) {
    $row = $result->fetch_assoc();
    $hashed_password = $row['password'];
    // Verify the input password with the hashed password
    if (password_verify($input_password, $hashed_password)) {
        // Delete enrollment record
        $sql_delete = "DELETE FROM enrollments WHERE id='$enrollment_id'
AND user_id='$user_id'";
        if ($conn->query($sql_delete) === TRUE) {
            echo "Запись успешно удалена.";
        } else {
            echo "Ошибка при удалении записи: " . $conn->error;
        }
    } else {
        echo "Неправильный пароль.";
    }
} else {
```

```
    echo "Пользователь не найден.";
}
$conn->close();
?>
```

Листинг 12 – Код страницы получения записей «get_enrollments.php»

```
<?php
session_start();
if (!isset($_SESSION['admin_email'])) {
    header('Location: admin_login.php');
    exit();
}
$servername = "localhost";
$username = "felixeob_courses";
$password = "Fel22446688";
$dbname = "felixeob_courses";
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
$data = json_decode(file_get_contents("php://input"), true);
$schedule_id = $data['schedule_id'];
$sql = "SELECT users.full_name AS parent_name, children.name AS child_name
        FROM enrollments
        JOIN users ON enrollments.user_id = users.id
        JOIN children ON enrollments.child_id = children.id
        WHERE enrollments.schedule_id = '$schedule_id'";
$result = $conn->query($sql);
$enrollments = [];
if ($result->num_rows > 0) {
    while ($row = $result->fetch_assoc()) {
        $enrollments[] = $row;
    }
}
$conn->close();
echo json_encode($enrollments);
?>
```

Листинг 13 – Код страницы сохранения профиля «save_profile.php»

```
<?php
session_start();
$servername = "localhost";
$username = "felixeob_courses";
$password = "Fel22446688";
$dbname = "felixeob_courses";
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
if (!isset($_SESSION['username']) && !isset($_SESSION['user_id'])) {
    echo "<script>
        if (localStorage.getItem('username') && localStor-
age.getItem('user_id')) {
            fetch('auto_login.php', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json'
```

```
                },
                body: JSON.stringify({
                    username: localStorage.getItem('username'),
                    user_id: localStorage.getItem('user_id')
                })
            }).then(response => response.json())
              .then(data => {
                  if (data.success) {
                      location.reload();
                  } else {
                      window.location.href = 'login.php';
                  }
              });
        } else {
            window.location.href = 'login.php';
        }
    </script>";
    exit();
}
$user_id = $_SESSION['user_id'];
$sql_check = "SELECT is_logged_in FROM users WHERE id = ?";
$stmt_check = $conn->prepare($sql_check);
$stmt_check->bind_param("i", $user_id);
$stmt_check->execute();
$stmt_check->bind_result($is_logged_in);
$stmt_check->fetch();
$stmt_check->close();
if ($is_logged_in == 0) {
    header('Location: login.php');
    exit();
}
$conn->close();
$servername = "localhost";
$username = "felixeob_courses";
$password = "Fel22446688";
$dbname = "felixeob_courses";
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
$data = json_decode(file_get_contents("php://input"), true);
$full_name = $data['full_name'];
$username = $data['username'];
$email = $data['email'];
$phone_number = $data['phone_number'];
$password = $data['password'];
$user_id = $_SESSION['user_id'];
if (!preg_match("/^(?=.*\d)(?=.*[a-z])(?=.*[A-Z]).{6,20}$/", $password)) {
    echo "Пароль должен содержать прописную и строчную букву, цифру и быть
длиной от 6 до 20 символов.";
    exit();
}
$hashed_password = password_hash($password, PASSWORD_DEFAULT);
$sql = "UPDATE users SET full_name=?, username=?, email=?, phone_number=?,
password=? WHERE id=?";
$stmt = $conn->prepare($sql);
$stmt->bind_param("ssssi", $full_name, $username, $email, $phone_number,
$hashed_password, $user_id);
if ($stmt->execute()) {
    echo "Данные профиля успешно обновлены";
} else {
    echo "Error: " . $stmt->error;
```

```
}
$stmt->close();
$conn->close();
?>
```

Листинг 14 – Код страницы добавления ребенка «save_child.php»

```
<?php
session_start();
$servername = "localhost";
$username = "felixeob_courses";
$password = "Fel22446688";
$dbname = "felixeob_courses";
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
if (!isset($_SESSION['username']) && !isset($_SESSION['user_id'])) {
    echo "<script>
        if (localStorage.getItem('username') && localStor-
age.getItem('user_id')) {
            fetch('auto_login.php', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json'
                },
                body: JSON.stringify({
                    username: localStorage.getItem('username'),
                    user_id: localStorage.getItem('user_id')
                })
            }).then(response => response.json())
              .then(data => {
                  if (data.success) {
                      location.reload();
                  } else {
                      window.location.href = 'login.php';
                  }
              });
        } else {
            window.location.href = 'login.php';
        }
    </script>";
    exit();
}
$user_id = $_SESSION['user_id'];
$sql_check = "SELECT is_logged_in FROM users WHERE id = ?";
$stmt_check = $conn->prepare($sql_check);
$stmt_check->bind_param("i", $user_id);
$stmt_check->execute();
$stmt_check->bind_result($is_logged_in);
$stmt_check->fetch();
$stmt_check->close();
if ($is_logged_in == 0) {
    header('Location: login.php');
    exit();
}
$conn->close();
$servername = "localhost";
$username = "felixeob_courses";
$password = "Fel22446688";
```

```
$dbname = "felixeob_courses";
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
$data = json_decode(file_get_contents("php://input"), true);
$name = $data['name'];
$age = $data['age'];
$gender = $data['gender'];
$user_id = $_SESSION['user_id'];
$sql = "INSERT INTO children (user_id, name, age, gender) VALUES (?, ?, ?,
?)";
$stmt = $conn->prepare($sql);
$stmt->bind_param("isis", $user_id, $name, $age, $gender);
if ($stmt->execute()) {
    echo "Ребенок добавлен!";
} else {
    echo "Ошибка: " . $stmt->error;
}
$stmt->close();
$conn->close();
?>
```

Листинг 15 – Код страницы удаления ребенка «delete_child.php»

```
<?php
session_start();
$servername = "localhost";
$username = "felixeob_courses";
$password = "Fel22446688";
$dbname = "felixeob_courses";
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
if (!isset($_SESSION['username']) && !isset($_SESSION['user_id'])) {
    echo "<script>
        if (localStorage.getItem('username') && localStor-
age.getItem('user_id')) {
            fetch('auto_login.php', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json'
                },
                body: JSON.stringify({
                    username: localStorage.getItem('username'),
                    user_id: localStorage.getItem('user_id')
                })
            }).then(response => response.json())
              .then(data => {
                  if (data.success) {
                      location.reload();
                  } else {
                      window.location.href = 'login.php';
                  }
              });
        } else {
            window.location.href = 'login.php';
        }
    </script>";
```

```
    exit();
}
$user_id = $_SESSION['user_id'];
$sql_check = "SELECT is_logged_in FROM users WHERE id = ?";
$stmt_check = $conn->prepare($sql_check);
$stmt_check->bind_param("i", $user_id);
$stmt_check->execute();
$stmt_check->bind_result($is_logged_in);
$stmt_check->fetch();
$stmt_check->close();
if ($is_logged_in == 0) {
    header('Location: login.php');
    exit();
}
$conn->close();
$servername = "localhost";
$username = "felixeob_courses";
$password = "Fel22446688";
$dbname = "felixeob_courses";
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
$data = json_decode(file_get_contents("php://input"), true);
$child_id = $data['id'];
$user_password = $data['password'];
// Получение хешированного пароля
$sql = "SELECT password FROM users WHERE id = '".$_SESSION['user_id']."'";
$result = $conn->query($sql);
if ($result->num_rows == 1) {
    $row = $result->fetch_assoc();
    if (password_verify($user_password, $row['password'])) {
        // Удаление записей о ребенке
        $conn->query("DELETE FROM enrollments WHERE child_id =
'$child_id'");
        $conn->query("DELETE FROM children WHERE id = '$child_id' AND
user_id = '".$_SESSION['user_id']."'");
        echo "Ребенок и связанные записи удалены";
    } else {
        echo "Неверный пароль";
    }
} else {
    echo "Ошибка при проверке пароля";
}
$conn->close();
?>
```

Листинг 16 – Код страницы регистрации «register.php»

```
<?php
$servername = "localhost";
$username = "felixeob_courses";
$password = "Fel22446688";
$dbname = "felixeob_courses";
// Создаем подключение к базе данных
$conn = new mysqli($servername, $username, $password, $dbname);
// Проверяем подключение
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
```

```
// Получаем данные из запроса
$data = json_decode(file_get_contents("php://input"), true);
$full_name = $data['full_name'];
$username = $data['username'];
$email = $data['email'];
$password = $data['password'];
$phone_number = $data['phone_number'];
// Проверка ФИО
if (!preg_match("/^[А-ЯЁ][а-яё]+(?:\s[А-ЯЁ][а-яё]+){1,4}$/u", $full_name))
{
    echo json_encode(["status" => "error", "message" => "ФИО должно содер-
жать от 2 до 5 слов, каждое слово с заглавной буквы."]);
    exit();
}
// Проверка на существование пользователя с таким же логином или email
$sql = "SELECT id FROM users WHERE username=? OR email=?";
$stmt = $conn->prepare($sql);
$stmt->bind_param("ss", $username, $email);
$stmt->execute();
$stmt->store_result();
if ($stmt->num_rows > 0) {
    echo json_encode(["status" => "error", "message" => "Пользователь с та-
ким логином или email уже существует."]);
    exit();
}
// Хеширование пароля
$hashed_password = password_hash($password, PASSWORD_DEFAULT);
// Вставка данных в базу данных временных пользователей
$sql = "INSERT INTO pending_users (full_name, username, email, password,
phone_number) VALUES (?, ?, ?, ?, ?)";
$stmt = $conn->prepare($sql);
$stmt->bind_param("sssss", $full_name, $username, $email, $hashed_password,
$phone_number);
if ($stmt->execute() === TRUE) {
    echo json_encode(["status" => "success"]);
} else {
    echo json_encode(["status" => "error", "message" => "Ошибка: " . $stmt-
>error]);
}
$conn->close();
?>
```

Листинг 17 – Код страницы подтверждения пароля «check_session.php»

```
<?php
session_start();
$response = array('is_logged_in' => false);
if (isset($_SESSION['username']) && isset($_SESSION['user_id'])) {
    $response['is_logged_in'] = true;
}
echo json_encode($response);
?>
```

Листинг 18 – Код настройки веб-приложения «manifest.json»

```
{
    "name": "Продленка",
    "short_name": "Продленка",
    "description": "Веб-приложение Продленка",
```

```
    "start_url": "index.html",
    "display": "standalone",
    "background_color": "#ffffff",
    "theme_color": "#4CAF50",
    "orientation": "portrait",
    "icons": [
        {
            "src": "icons/icon-192x192.png",
            "sizes": "192x192",
            "type": "image/png"
        },
        {
            "src": "icons/icon-512x512.png",
            "sizes": "512x512",
            "type": "image/png"
        }
    ]
}
```

Листинг 19 – Код обработки сервисов «service-worker.js»

```
self.addEventListener('install', function(event) {
    console.log('Service Worker installing.');
});
self.addEventListener('activate', function(event) {
    console.log('Service Worker activating.');
});
self.addEventListener('fetch', function(event) {
    console.log('Fetching:', event.request.url);
});
```

Листинг 20 – Код завершения сессии «logout.php»

```
<?php
session_start();
$servername = "localhost";
$username = "felixeob_courses";
$password = "Fel22446688";
$dbname = "felixeob_courses";
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
if (isset($_SESSION['user_id'])) {
    $user_id = $_SESSION['user_id'];
    $sql_update = "UPDATE users SET is_logged_in = 0, rememberMe = 0 WHERE
id = ?";
    $stmt_update = $conn->prepare($sql_update);
    $stmt_update->bind_param("i", $user_id);
    $stmt_update->execute();
}
session_unset();
session_destroy();
echo "<script>
    localStorage.removeItem('username');
    localStorage.removeItem('user_id');
    localStorage.removeItem('rememberMe');
    window.location.href = 'index.html';
</script>";
```

```
$conn->close();
?>
```

Листинг 21 – Код загрузки данных профиля «get_profile.php»

```
<?php
session_start();
$servername = "localhost";
$username = "felixeob_courses";
$password = "Fel22446688";
$dbname = "felixeob_courses";
$conn = new mysqli($servername, $username, $password, $dbname);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
if (!isset($_SESSION['username']) && !isset($_SESSION['user_id'])) {
    echo "<script>
        if (localStorage.getItem('username') && localStor-
age.getItem('user_id')) {
            fetch('auto_login.php', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json'
                },
                body: JSON.stringify({
                    username: localStorage.getItem('username'),
                    user_id: localStorage.getItem('user_id')
                })
            }).then(response => response.json())
              .then(data => {
                  if (data.success) {
                      location.reload();
                  } else {
                      window.location.href = 'login.php';
                  }
              });
        } else {
            window.location.href = 'login.php';
        }
    </script>";
    exit();
}
$user_id = $_SESSION['user_id'];
$sql_check = "SELECT is_logged_in FROM users WHERE id = ?";
$stmt_check = $conn->prepare($sql_check);
$stmt_check->bind_param("i", $user_id);
$stmt_check->execute();
$stmt_check->bind_result($is_logged_in);
$stmt_check->fetch();
$stmt_check->close();
if ($is_logged_in == 0) {
    header('Location: login.php');
    exit();
}
$conn->close();
$servername = "localhost";
$username = "felixeob_courses";
$password = "Fel22446688";
$dbname = "felixeob_courses";
$conn = new mysqli($servername, $username, $password, $dbname);
```

```
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
$user_id = $_SESSION['user_id'];
$sql = "SELECT full_name, username, email, phone_number FROM users WHERE
id='$user_id'";
$stmt = $conn->prepare($sql);
$stmt->bind_param("i", $user_id);
$stmt->execute();
$result = $stmt->get_result();
$row = $result->fetch_assoc();
echo json_encode($row);
$stmt->close();
$conn->close();
?>
```

Листинг 22 – Код со стилем

```
body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    background-color: #f2f2f2;
}
.container {
    text-align: center;
    max-width: 600px;
    margin: 20px auto;
    background-color: #fff;
    padding: 20px;
    border-radius: 8px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}
h1, h2, h3 {
    margin-bottom: 20px;
}
button {
    display: block;
    margin: 20px auto;
    padding: 10px 20px;
    background-color: #4CAF50;
    color: #fff;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    font-size: 16px;
}
button:hover {
    background-color: #45a049;
}
button.red {
    background-color: #f44336;
}
button.red:hover {
    background-color: #d32f2f;
}
label {
    display: block;
    margin: 10px 0;
}
```

```
input, select {
    margin-bottom: 10px;
    padding: 10px;
    width: calc(100% - 22px);
    box-sizing: border-box;
    border: 1px solid #ccc;
    border-radius: 4px;
}
.form-group {
    text-align: center;
    margin: 0 auto;
    width: 100%;
}
.course-info {
    border: 1px solid #ddd;
    padding: 10px;
    margin-bottom: 10px;
}
form {
    display: inline-block;
}
.u-section-1 .u-sheet-1 {
  min-height: 215px;
}
.u-section-1 .u-text-1 {
  font-size: 1.125rem;
  font-weight: 400;
  text-transform: none;
  letter-spacing: normal;
  margin: 60px 1048px 0 0;
}
.u-section-1 .u-btn-1 {
  --radius: 50px;
  border-style: solid;
  font-weight: 700;
  text-transform: uppercase;
  font-size: 1rem;
  letter-spacing: 1px;
  background-image: none;
  margin: 17px 11px 60px auto;
}
@media (max-width: 1199px) {
  .u-section-1 .u-sheet-1 {
    min-height: 206px;
  }
  .u-section-1 .u-text-1 {
    width: auto;
    margin-right: 848px;
  }
}
@media (max-width: 991px) {
  .u-section-1 .u-text-1 {
    margin-right: 628px;
  }
}
@media (max-width: 767px) {
  .u-section-1 .u-text-1 {
    margin-right: 448px;
    font-size: 1rem;
  }
}
@media (max-width: 575px) {
```

```
  .u-section-1 .u-sheet-1 {
    min-height: 52px;
  }
  .u-section-1 .u-text-1 {
    --radius: 50px;
    font-size: 1.25rem;
    letter-spacing: 1px;
    font-weight: 700;
    margin-top: 6px;
    margin-right: auto;
    margin-left: -2px;
    padding: 4px 30px 5px;
  }
  .u-section-1 .u-btn-1 {
    font-size: 0.875rem;
    margin-top: -44px;
    margin-right: 0;
    margin-bottom: 3px;
  }
}
.u-section-2 .u-sheet-1 {
  min-height: 333px;
}
.u-section-2 .u-custom-html-1 {
  margin-bottom: 26px;
  height: auto;
  min-height: 282px;
  margin-top: 26px;
}
@media (max-width: 575px) {
  .u-section-2 .u-sheet-1 {
    min-height: 214px;
  }
  .u-section-2 .u-custom-html-1 {
    margin-bottom: 0;
    min-height: 214px;
    margin-top: 0;
  }
}
.u-section-3 .u-sheet-1 {
  min-height: 384px;
}
.u-section-3 .u-custom-html-1 {
  margin-top: 60px;
  margin-bottom: 0;
}
.u-section-3 .u-text-1 {
  margin-top: -30px;
  margin-bottom: 0;
}
.u-section-3 .u-custom-html-2 {
  height: auto;
  min-height: 33px;
  width: 69px;
  margin: 44px auto 12px;
}
@media (max-width: 575px) {
  .u-section-3 .u-sheet-1 {
    min-height: 415px;
  }
  .u-section-3 .u-custom-html-1 {
    height: auto;
```

```
    min-height: 282px;
    width: 412px;
    margin-top: 20px;
    margin-left: -36px;
    margin-right: -36px;
  }
  .u-section-3 .u-text-1 {
    margin-top: 0;
  }
  .u-section-3 .u-custom-html-2 {
    margin-top: 0;
    margin-bottom: 60px;
  }
}
```

Листинг 23 – SQL запрос создания базы данных и таблиц

```
-- Создание базы данных
CREATE DATABASE felixbo_courses;
-- Создание таблицы users
CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(50) NOT NULL,
    password VARCHAR(255) NOT NULL,
    email VARCHAR(100) NOT NULL,
    full_name VARCHAR(255) NOT NULL,
    is_logged_in TINYINT(1) DEFAULT 0,
    rememberMe TINYINT(1) DEFAULT 0,
    phone_number VARCHAR(20) NOT NULL
);
-- Создание таблицы pending_users
CREATE TABLE pending_users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    full_name VARCHAR(100) NOT NULL,
    username VARCHAR(50) NOT NULL,
    email VARCHAR(100) NOT NULL,
    password VARCHAR(255) NOT NULL,
    phone_number VARCHAR(20) NOT NULL,
    registration_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    expiration_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CUR-
RENT_TIMESTAMP
);
-- Создание таблицы administrators
CREATE TABLE administrators (
    id INT AUTO_INCREMENT PRIMARY KEY,
    email VARCHAR(100) NOT NULL
);
-- Создание таблицы courses
CREATE TABLE courses (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100) NOT NULL
);
-- Создание таблицы schedule
CREATE TABLE schedule (
    id INT AUTO_INCREMENT PRIMARY KEY,
    course_id INT NOT NULL,
    date DATE NOT NULL,
    time TIME NOT NULL,
    available_slots INT NOT NULL,
    FOREIGN KEY (course_id) REFERENCES courses(id)
```

```
);
-- Создание таблицы children
CREATE TABLE children (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT NOT NULL,
    name VARCHAR(100) NOT NULL,
    age INT NOT NULL,
    gender CHAR(1) NOT NULL,
    FOREIGN KEY (user_id) REFERENCES users(id)
);
-- Создание таблицы enrollments
CREATE TABLE enrollments (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT NOT NULL,
    child_id INT NOT NULL,
    schedule_id INT NOT NULL,
    FOREIGN KEY (user_id) REFERENCES users(id),
    FOREIGN KEY (child_id) REFERENCES children(id),
    FOREIGN KEY (schedule_id) REFERENCES schedule(id)
);
```

Листинг 24 – SQL запрос вставки данных в таблицу

```
-- Вставка данных в таблицу users
INSERT INTO users (username, password, email, full_name, is_logged_in,
phone_number) VALUES
('user1', 'hashed_password1', 'user1@example.com', 'Иван Иванов', 0,
'+71234567890'),
('user2', 'hashed_password2', 'user2@example.com', 'Петр Петров', 0,
'+79876543210');
-- Вставка данных в таблицу courses
INSERT INTO courses (name) VALUES
('Английский язык'),
('Танцы');
-- Вставка данных в таблицу children
INSERT INTO children (user_id, name, age, gender) VALUES
(1, 'Леша', 9, 'М'),
(2, 'Мария', 11, 'Ж');
-- Вставка данных в таблицу schedule
INSERT INTO schedule (course_id, date, time, available_slots) VALUES
(1, '2023-12-25', '15:45:00', 10),
(2, '2023-12-26', '10:30:00', 5);
-- Вставка данных в таблицу enrollments
INSERT INTO enrollments (user_id, child_id, schedule_id) VALUES
(1, 1, 1),
(2, 2, 2);
-- Вставка данных в таблицу pending_users
INSERT INTO pending_users (full_name, username, email, password, phone_num-
ber) VALUES
('Сергей Сергеев', 'sergei', 'sergei@example.com', 'hashed_password3',
'+79012345678'),
('Андрей Андреев', 'andrei', 'andrei@example.com', 'hashed_password4',
'+79654321098');
-- Вставка данных в таблицу administrators
INSERT INTO administrators (email) VALUES
('admin1@example.com'),
('admin2@example.com');
```