МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра информационных технологий**

**ЛАБОРАТОРНАЯ РАБОТА №8**

Работу выполнил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. А. Иванов

(подпись)

Направление подготовки 02.03.03 Математическое обеспечение и администрирование информационных систем

Направленность Технология проектирования программного обеспечения

Руководитель Н.Ю. Добровольская

(подпись)

Краснодар

2024

Тема: документирование и развёртывание ПС.

Цель: освоение методики документирования ПС.

Задание

1. Собрать документированные материалы, разработанные при выполнении лабораторных работ №1-7 в единый документ с разбивкой на разделы. Темы лабораторных работ обозначить названиями разделов. В разделы не включать цель, задание и выводы из отчётов о выполнении лабораторных работ.
2. Дополнительно отдельным разделом 8 описать назначение, технические характеристики, принцип работы и меры безопасности при эксплуатации ПС.
3. Дополнительно отдельным разделом 9 составить руководство пользователя.

**Раздел 1: анализ предметной области**

**1.1 Объект предметной области**

Объектом информатизации является образовательный центр "ТехноКод", специализирующийся на проведении компьютерных занятий для людей любых возрастов. Центр был основан в 2010 году и успешно функционирует уже более 10 лет. Он предлагает широкий спектр курсов, включающих программирование, веб-разработку и другие компьютерные навыки.

Центр "ТехноКод" имеет несколько филиалов в разных городах России. Один из филиалов находится в городе Краснодаре, по адресу ул. Программистов, д. 10. Контактный телефон филиала: +7 (861) 123-45-67.

Управляющей филиала является Иванова Дарья Сергеевна.

Заместителем директора по учебной работе является Огнёва Варвара Анатольевна.

В списке слушателей курсов "ТехноКод" содержатся данные о каждом студенте, включая его персональные данные (ФИО, дата рождения) и контактную информацию (телефон, электронная почта). Каждый слушатель может быть записан на один или несколько курсов.

Список предметов включает различные компьютерные дисциплины, которые предлагаются в центре "ТехноКод". Это могут быть курсы по языкам программирования (Python, Java, C++), разработке веб-сайтов (HTML, CSS, JavaScript) и другим направлениям. Каждый предмет имеет своего преподавателя.

Список преподавателей включает информацию о каждом преподавателе, включая его ФИО, контактные данные, квалификацию и опыт работы. Каждый преподаватель может вести несколько предметов и заниматься обучением на разных курсах.

Журнал учёта успеваемости содержит информацию о каждом слушателе и его академических достижениях. Для каждого предмета ведётся отдельная запись, включающая оценки за выполнение заданий, тесты и экзамены. Журнал позволяет отслеживать успехи каждого слушателя и оценивать его академическую производительность.

В качестве информатизируемого бизнес-процесса был выбран "Запись на курсы".

Запись на курсы начинается с заполнения заявки на курсы. В ней указываются персональные данные студента (ФИО, дата рождения, номер телефона, адрес электронной почты). При заполнении заявки можно ознакомиться с существующими курсами и ценами. После заполнения заявка отправляется, обрабатывается, присылается ответ на заявку и уведомление преподавателю.

Структурная схема, иллюстрирующая архитектуру заявок в "ТехноКод" представлена на рисунке 1.

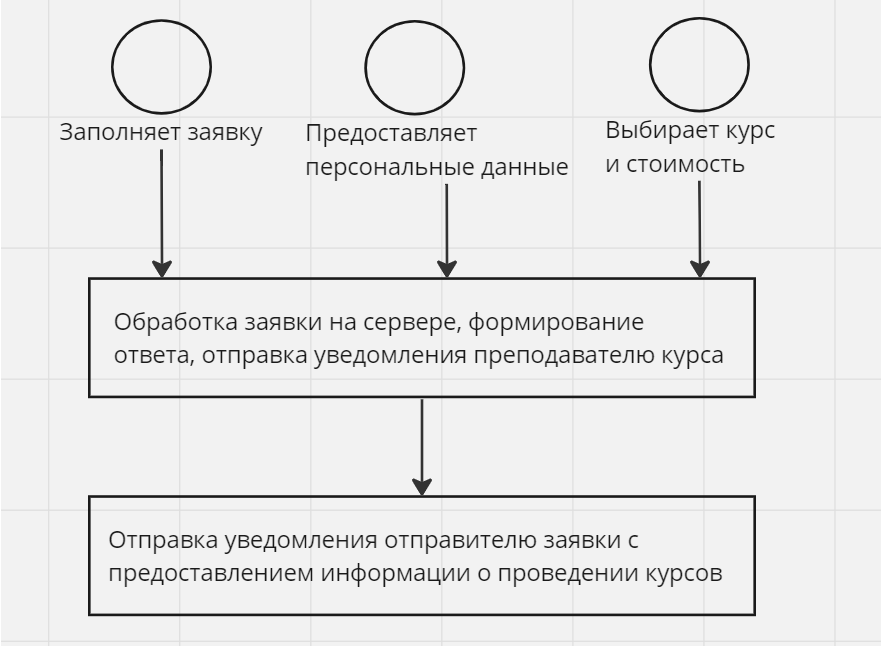


Рисунок 1 – структура заявок "ТехноКод"

Основные показатели объекта, на улучшение которых направлена информатизация: скорость обработки заявки с момента её подачи.

**1.2 Процесс информатизации**

Процесс информатизации включает в себя несколько задач объекта. Такая структура представлена в виде схемы "Чёрный ящик" (рисунок 2).

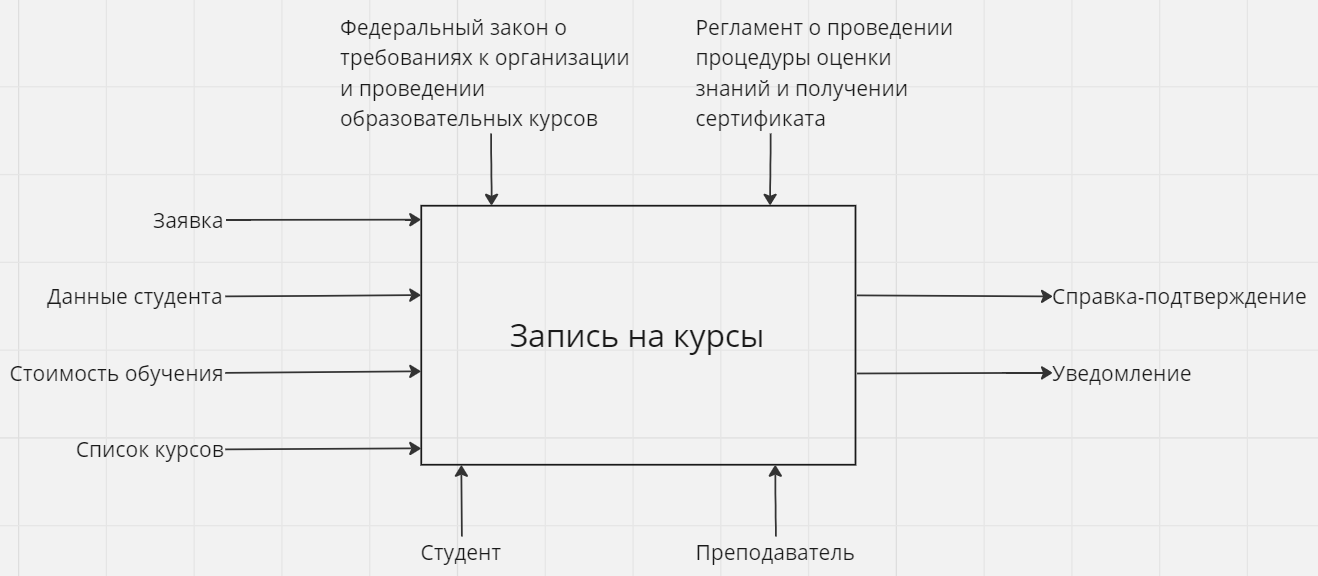


Рисунок 2 – схема "Чёрный ящик" отправки заявок на запись на курсы

Действующие лица:

– Студент (отправитель заявки)

– Преподаватель (который будет вести курсы по данной заявке)

Входная информация:

– Заявка

– Данные студента

– Стоимость обучения

– Список курсов

Управляющая информация:

– Федеральный закон о требованиях к организации и проведении образовательных курсов

– Регламент о проведении процедуры оценки знаний и получении сертификата

Функциональная информация:

– Студент (отправитель заявки)

– Преподаватель (который будет вести курсы по данной заявке)

Выходная информация:

– Справка-подтверждение

– Уведомление для преподавателя

Можно провести декомпозицию для упрощения изучения бизнес-процессов. Тогда исходная функция разделится на более мелкие подзадачи. Таким образом, сформируется иерархия задач. Декомпозиция данной информационной системы изображена на рисунке 3.



Рисунок 3 – декомпозиция записи на курсы

Задачами данного бизнес-процесса являются:

– Задача ввода данных студента

– Задача выбора курсов и цены обучения

– Задача отправки уведомления преподавателю

– Задача формирования справки-подтверждения

Для упрощения взаимодействия с будущими пользователями системы можно составить диаграмму вариантов использования, представленную на рисунке 4.

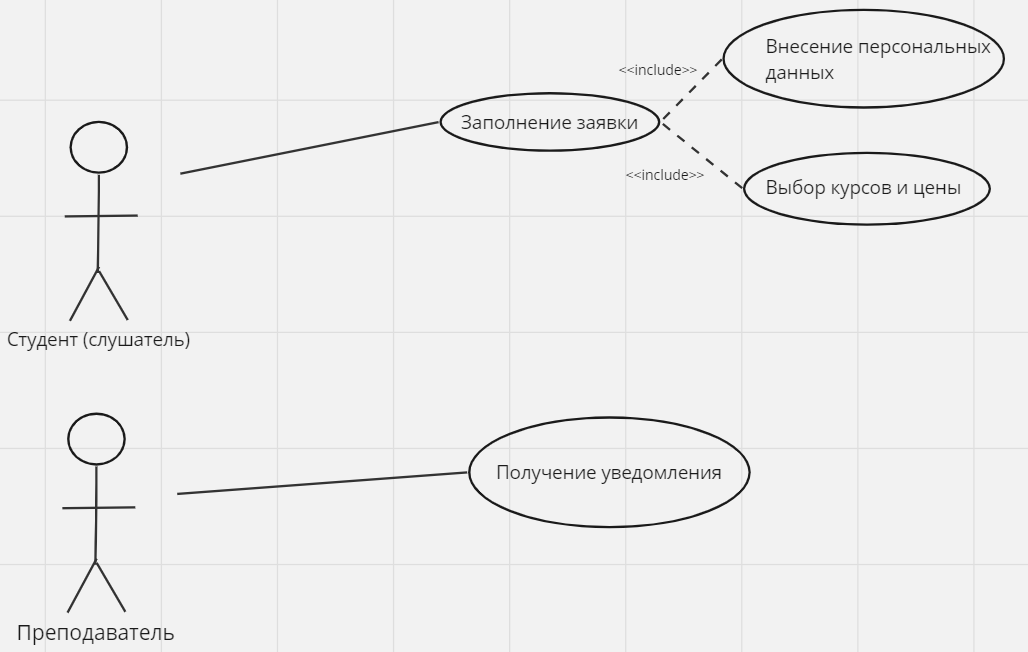


Рисунок 4 – диаграмма вариантов использования

**1.3 Описание информационных потоков**

Вся информация перемещается в рамках данного процесса информатизации в виде информационных потоков, и её нужно описать и систематизировать.

Такое описание представлено в виде таблиц 1 и 2.

Таблица 1. Реестр входных информационных потоков

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Наименование и назначение потока | Форма представления | Обработчик | Корреспондент (откуда поступает) | Характеристики обработки | |
| Трудозатраты, чел ч. | Периодичность, регламент |
| 1 | Заявка | Электронная | Сервер | Студент | 1 | По требованию |
| 2 | Данные студента | Электронная | Сервер | Студент | 1 | По требованию |
| 3 | Стоимость обучения | Электронная | Сервер | Студент | 1 | По требованию |
| 4 | Стоимость курсов | Электронная | Сервер | Студент | 1 | По требованию |

Таблица 2. Реестр выходных информационных потоков

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Наименование и назначение потока | Форма представления | Обработчик | Корреспондент (куда направляется) | Характеристики обработки | |
| Трудозатраты, чел ч. | Периодичность, регламент |
| 1 | Справка-подтверждение | Электронная | Сервер | Студент | 1 | По требованию |
| 2 | Уведомление | Электронная | Сервер | Преподаватель | 1 | По требованию |

Описанная выше система может быть улучшена с помощью программного обеспечения, которое будет автоматизировать обработку заявок и рассылку уведомлений, а также создание справки-подтверждения. В отсутствии такого программного обеспечения все вышеперечисленные действия должен будет выполнять сотрудник тех. поддержки "ТехноКод". Это увеличит время обработки заявок, а также появится возможность допущения ошибок человеческой невнимательности.

Однако, у данного программного обеспечения есть ограничения:

– Ограничение на длину персональных данных (ФИО - 70 символов, номер телефона - 11 символов).

Нормативно-справочная документация: ФЗ о требованиях к организации и проведении образовательных курсов; Регламент о проведении процедуры оценки знаний и получении сертификата.

**Раздел 2: анализ существующих компьютерных разработок**

**2.1 Системное описание существующих подобных программных систем**

1. Клиент-серверная архитектура онлайн-школы "Skillbox".

"Skillbox" является одной из самых популярных онлайн-школ в России. Их клиент-серверная архитектура предназначена для быстрой обработки большого количества запросов записи на курсы и показа полного каталога имеющихся курсов с фильтрами для поиска.

Их система помогает автоматизировать следующие операции:

– регистрация и аутентификация пользователей;  
 – получение информации по каждому курсу;

– заполнение заявки записи на курс;

– обработку заполненных форм;

– выбор нужных фильтров при поиске по странице;

– получение обратной связи от администрации сайта.

Стоит отметить, что пользовательский интерфейс на главном сайте "Skillbox" выглядит приятным и является интуитивно понятным, что, скорее всего, помогает пользователям быстрее находить нужные курсы.

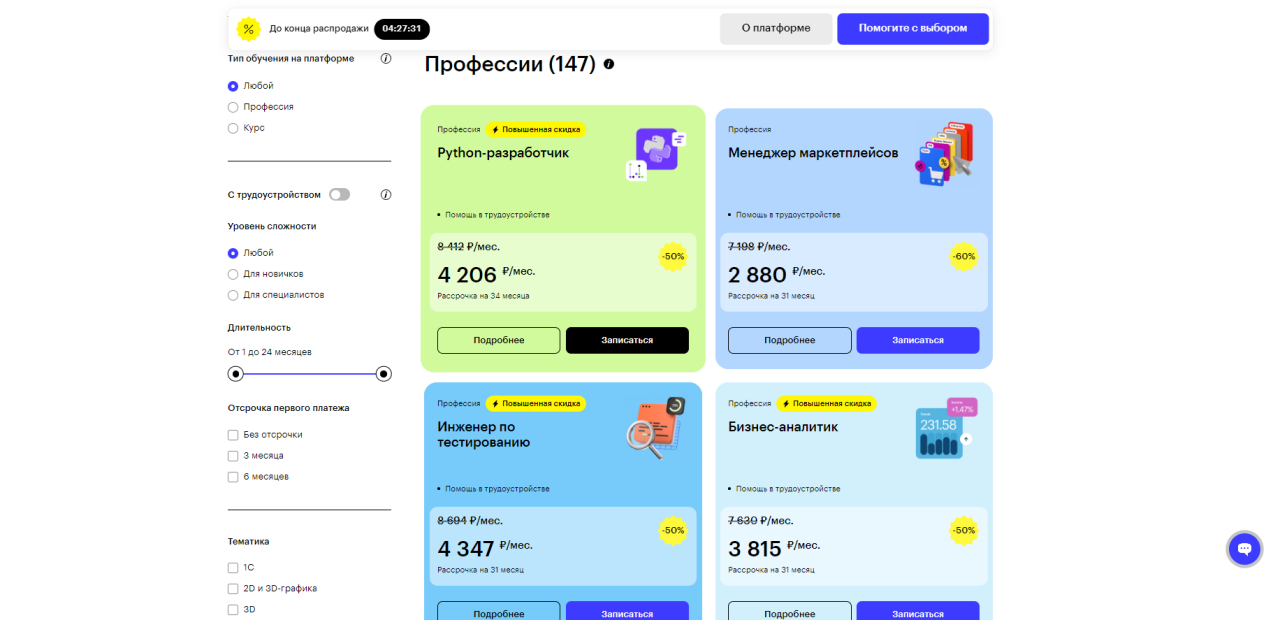


Рисунок 1 – общий вид главной страницы сайта "Skillbox"

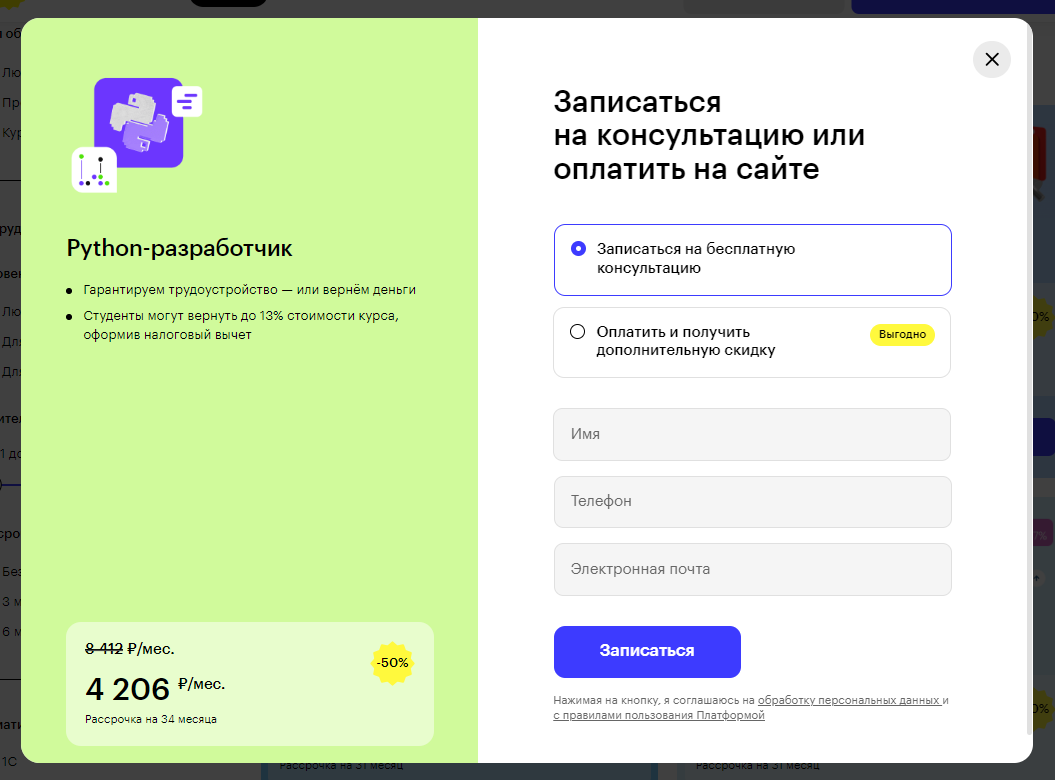


Рисунок 2 – форма записи на курс с сайта "Skillbox"

Достоинства:

– Быстрая обработка запросов: клиент-серверная архитектура "Skillbox" способствует быстрой обработке большого количества запросов на запись на курсы. Это позволяет пользователям получить моментальный отклик и узнать о доступности мест на выбранных курсах.

– Интуитивно понятный пользовательский интерфейс: сайт "Skillbox" предлагает приятный и интуитивно понятный пользовательский интерфейс, что помогает пользователям быстро и легко заполнить форму записи на курсы. Это улучшает пользовательский опыт и способствует удобству использования системы.

– Фильтры и поиск: система "Skillbox" предоставляет возможность использовать фильтры для поиска курсов по различным параметрам, таким как тематика, продолжительность, стоимость и другие. Это позволяет пользователям быстро найти наиболее подходящие им курсы и упрощает процесс выбора.

Недостатки:  
 – Ограниченность функциональности: возможности системы "Skillbox" могут быть ограничены только заполнением формы на запись на курсы и получением обратной связи от администрации. Отсутствуют дополнительные функции, такие как журнал успеваемости.

1. Клиент-серверная архитектура онлайн-школы "GeekBrains".

"GeekBrains" - это популярная онлайн-школа, специализирующаяся на обучении компьютерным наукам и IT-направлениям. Их клиент-серверная архитектура предоставляет удобные инструменты для записи на курсы и управления образовательным процессом.

Система "GeekBrains" предлагает следующие функциональности:

– авторизация и аутентификация пользователей;

– заполнение заявки на курс через веб-интерфейс;

– автоматическую обработку заявок;

– журнал учёта успеваемости.

"GeekBrains" также предоставляет удобный интерфейс для преподавателей, где они могут загружать материалы для курсов, выставлять оценки и взаимодействовать с учениками.

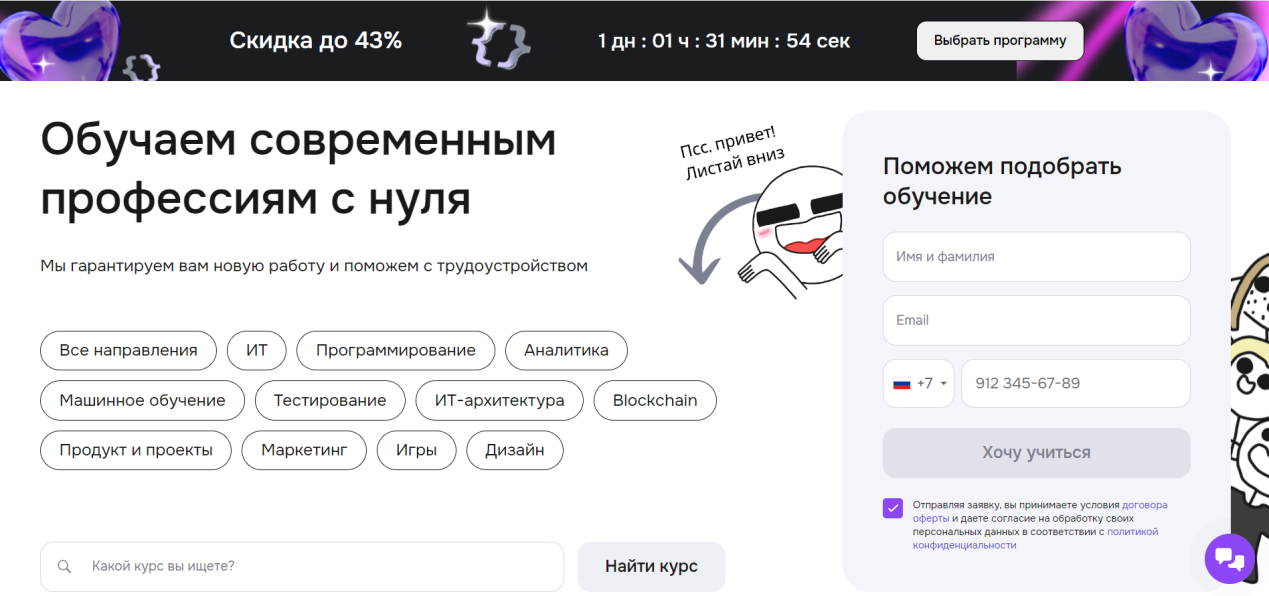


Рисунок 3 – фильтры на сайте "GeekBrains"

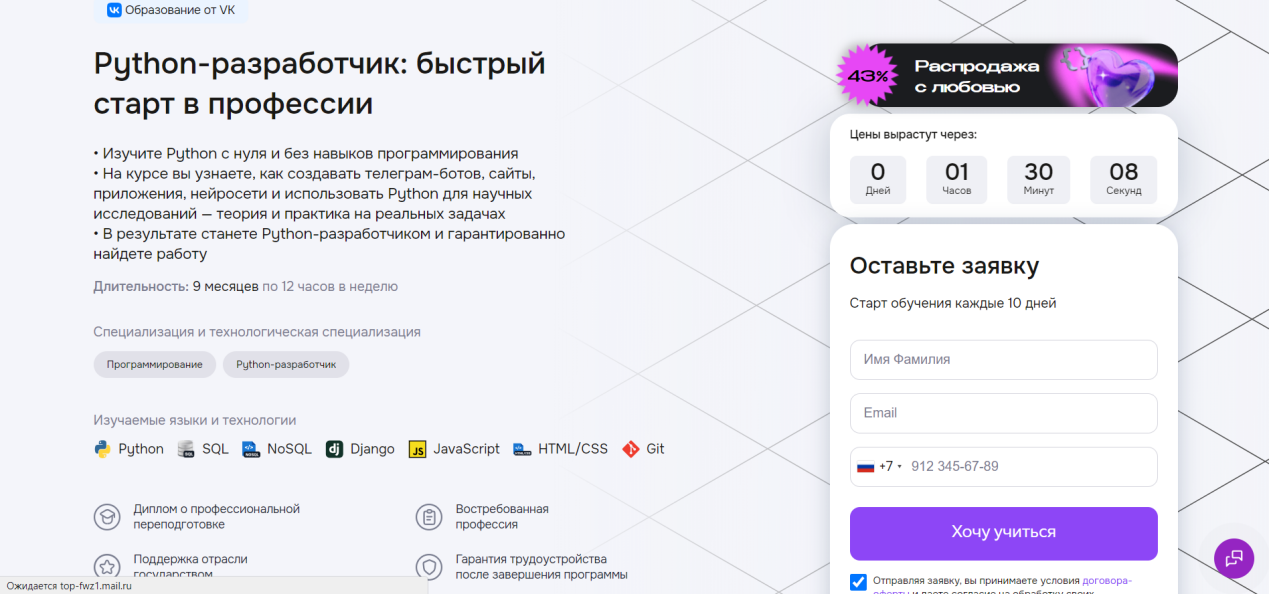


Рисунок 4 – заполнение заявки на курс на сайте "GeekBrains"

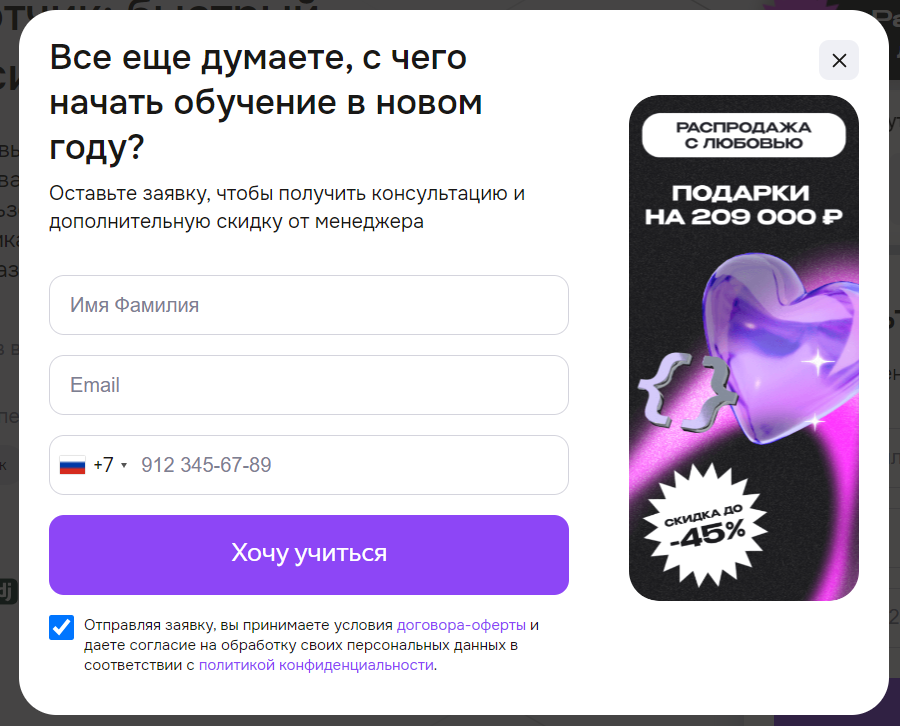


Рисунок 5 – заявка на консультацию на сайте "GeekBrains"

Достоинства:

– Полный каталог курсов: "GeekBrains" предоставляет полный каталог курсов с описанием и рейтингами, что помогает пользователям выбрать наиболее подходящий курс. Это обеспечивает большую прозрачность и информированность при выборе курса.

– Журнал успеваемости: система "GeekBrains" включает журнал успеваемости, где пользователи могут отслеживать свои академические достижения и оценки на каждом курсе. Это позволяет ученикам контролировать свой прогресс и оценивать свои успехи.

– Удобный интерфейс и простота использования: система "GeekBrains" предлагает удобный и простой в использовании интерфейс для заполнения формы на запись на курсы. Это помогает пользователям быстро и легко заполнить необходимую информацию.

Недостатки:

– Ограниченность доступа к информации: система "GeekBrains" не предоставляет достаточно подробной информации о каждом преподавателе, что может затруднить пользователей в принятии решения о записи на курс.

**2.2 Сравнительная характеристика описанных систем**

|  |  |  |
| --- | --- | --- |
| Название | Skillbox | GeekBrains |
| Назначение системы | Онлайн-занятия | Онлайн-занятия |
| Эффективность системы | Высокая | Высокая |
| Гибкость системы | Средняя | Высокая |
| Открытость системы | Высокая | Высокая |
| Удобство пользовательского интерфейса системы | Высокая | Средняя |
| Безотказность системы | Высокая | Высокая |
| Долговечность системы | Высокая | Высокая |
| Сопровождаемость системы | Высокая | Высокая |
| Стоимость системы | Средняя | Средняя |
| Наличие технической поддержки | Низкая | Низкая |

**Раздел 3: техническое задание на создание программного продукта**

**3.1 Назначение подсистемы и цели создания**

Назначением программы является автоматизация процесса "Запись на курсы" для организации "ТехноКод".

Целью программы является оптимизация процесса обработки заявок записи на курсы и информатизация процесса "Запись на курсы".

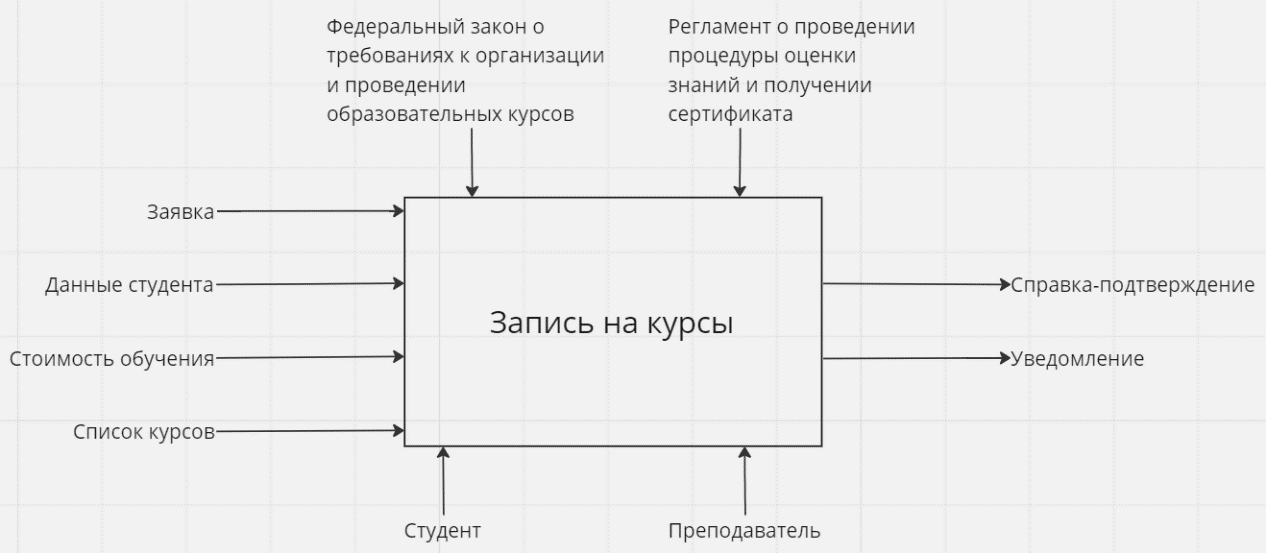


Рисунок 1 – чёрный ящик

**3.2 Функциональные задачи и структура программы**

Задачи:

1 Задача ввода персональных данных

2 Задача выбора курса(ов)

3 Задача формирования справки-подтверждения

4 Задача отправки уведомления преподавателю

Диаграмма таблиц базы данных изображена на рисунке 1.

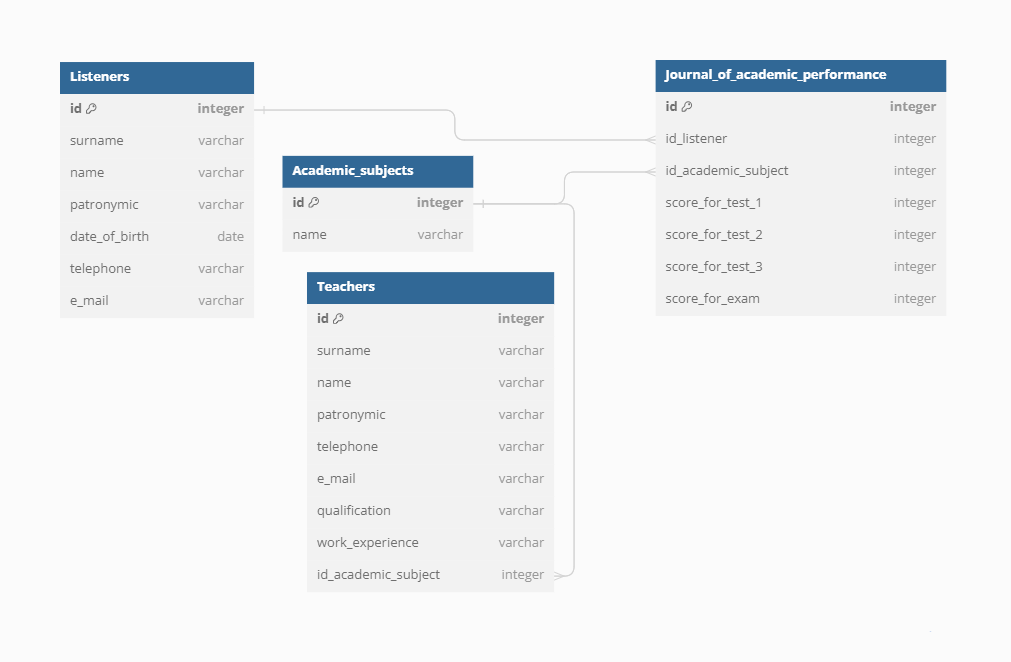


Рисунок 1 - Диаграмма таблиц базы данных

**3.3 Диаграмма вариантов использования**

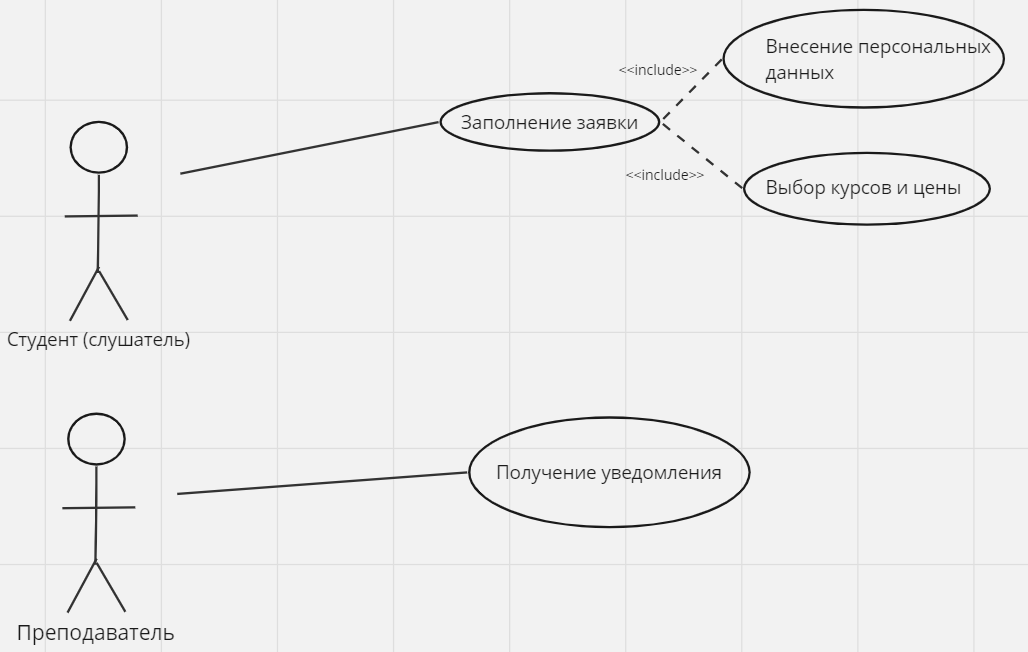
****

Рисунок 2 - Диаграмма вариантов использования

**3.4 Функциональные требования к системе**

Разрабатываемое ПО представляет из себя веб-портал.

1 Требования к задаче ввода персональных данных

Программа должна получить корректные данные из формы, составить на их основе корректный запрос к базе данных и отправить этот запрос.

2 Требования к задаче выбора курсов

После регистрации пользователь может выбрать подходящий курс из каталога и записаться на него, после чего информация об этом пользователе обновится в базе данных.

3 Требования к задаче формирования справки-подтверждения

После записи на курс пользователь получает справку-подтверждение, которая подтверждает его запись на курс.

4 Требования к задаче отправке уведомления преподавателю

После записи на курс пользователя формируется уведомление, которое отправляется преподавателю, который ведёт данный курс, и на основании этих данных обновляется база данных.

Требования к входным данным

Перечень входных данных:

- ФИО пользователя;

- Дата рождения пользователя;

- Контактная информация пользователя (номер телефон, электронная почта).

Требования к программной реализации

Все данные пользователей, записей на курсы, журнала успеваемости должны храниться в реляционной СУБД. Интерфейс пользователя состоит из формы регистрации, каталога курсов и личного кабинета. Все элементы пользовательского интерфейса находятся должны находиться на главной странице веб-портала.

Данный веб-портал должен быть написан на языке программирования Python с использованием Django Rest Framework и React, а также использовать СУБД Postgresql.

**3.5 Требования к СУБД и IDE**

Требования к СУБД разрабатываемого ПО

Разрабатываемое ПО использует СУБД Postgresql

1 Требования к поддержке реляционных таблиц

СУБД должна поддерживать структуру базы данных из 4 таблиц: список слушателей курсов, список предметов, список учителей, журнал учёта успеваемости.

2 Требования CRUD

СУБД должна поддерживать принципы CRUD (Create, Read, Update, Delete) для каждой таблицы.

3 Требования к транзакциям

СУБД должна поддерживать структуру выполнения транзакций для обеспечения целостности данных.

4 Требования надёжности

СУБД должна быть надёжной и безопасной, обеспечивать отказоустойчивость и иметь систему Backup`ов (резервные копии данных).

5 Требования к совместимости

СУБД должна быть совместима с выбранным языком программирования разработки.

Требования к IDE разрабатываемого ПО

Разрабатываемое ПО использует IDE Pycharm и Visual Studio Code

1 Требования к поддержке

IDE должна поддерживать язык программирования, используемый в разработке ПО

2 Требования к интегрированным инструментам

IDE должна иметь широкий спектр интегрированных инструментов для функциональной разработки ПО

3 Требования к веб-технологиям

IDE должна иметь поддержку таких веб-технологий как HTML, CSS, JavaScript.

4 Требования к Git

IDE должна быть совместима с системой контроля версий Git для комфортной разработки ПО

5 Требования к расширениям

IDE должна иметь возможность поддержки плагинов и расширений для повышения производительности и функциональности

6 Требования к совместимости

IDE должна быть совместима с операционной системой Windows 11, на которой ведётся разработка ПО

**3.6 Нефункциональные требования к системе**

Основные нефункциональные требования к ПО:

- Малая стоимость (данное ПО должно быть дешёвым в разработке, чтобы оно могло конкурировать на рынке, но это не должно сказаться на качестве разрабатываемого ПО);

- Быстрота обработки запросов (использование реляционной СУБД должно положительно сказать на скорости обработки запросов данного ПО);

- Простота использования (данное ПО должно иметь интуитивно-понятный интерфейс пользователя);

- Контроль работы ПО (данное ПО должно предоставлять доступ к отслеживанию различных шагов выполнения обработки запросов).

**Раздел 4: проектирование функциональной структуры программного продукта; функционально-ориентированный подход**

**4.1 Функциональная модель**

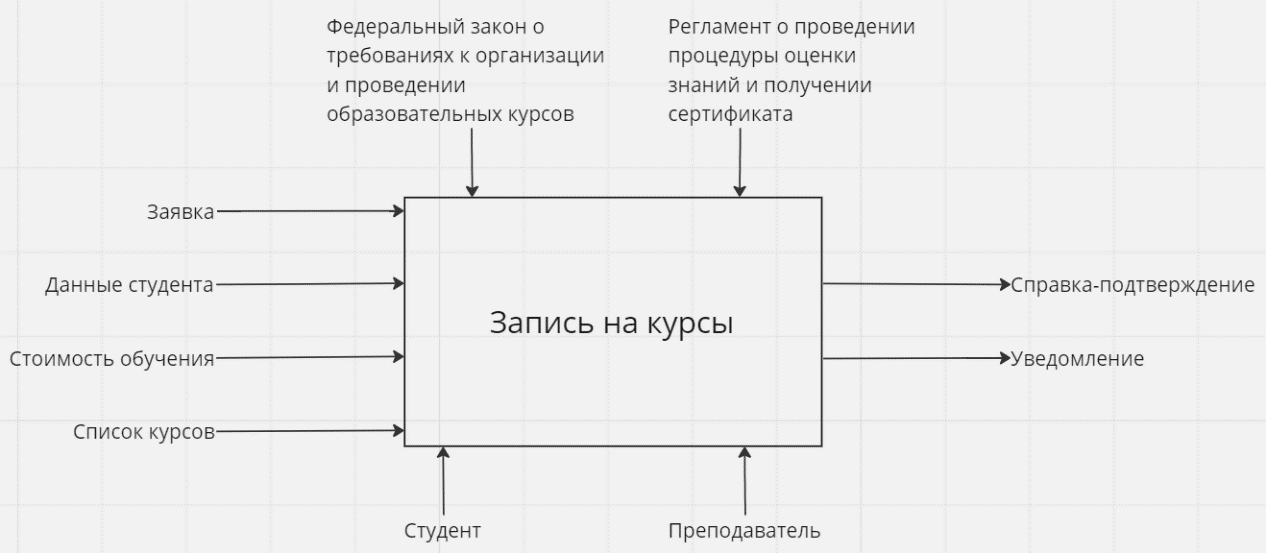


Рисунок 1 – контекстная диаграмма в нотации IDEF0

**4.2 Диаграмма декомпозиции А0**

****

Рисунок 2 – диаграмма декомпозиции на подпроцессы

Описание элементов диаграммы декомпозиции

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование стрелки | Источник стрелки | Тип стрелки-источника | Приемник стрелки | Тип стрелки-приемника |
| Заявка | Внешняя граница | Input | Ввод персональных данных пользователя | Input |
| Данные пользователя | Внешняя граница | Input | Ввод персональных данных пользователя | Input |
| Пользователь | Внешняя граница | Mechanism | Ввод персональных данных пользователя | Mechanism |
| Данные пользователя | Ввод персональных данных пользователя | Output | Сохранение данных о пользователе в базе данных | Input |
| Стоимость обучения | Внешняя граница | Input | Запись на курс(ы) | Input |
| Данные пользователя | Сохранение данных о пользователе в базе данных | Output | Запись на курс(ы) | Input |
| Список курсов | Внешняя граница | Input | Запись на курс(ы) | Input |
| Данные пользователя и информация о курсе(ах) | Запись на курсы | Output | Обновление данных в базе данных и формирование справки-подтверждения | Output |
| Федеральный закон о требованиях к организации и проведении образовательных курсов | Внешняя граница | Control | Обновление данных в базе данных и формирование справки-подтверждения | Control |
| Регламент о проведении процедуры оценки знаний и получении сертификата | Внешняя граница | Control | Обновление данных в базе данных и формирование справки-подтверждения | Control |
| Преподаватель | Внешняя граница | Mechanism | Обновление данных в базе данных и формирование справки-подтверждения | Mechanism |
| Справка-подтверждение | Обновление данных в базе данных и формирование справки-подтверждения | Output | Внешняя граница | Output |
| Отправка уведомления преподавателю | Обновление данных в базе данных и формирование справки-подтверждения | Output | Внешняя граница | Output |

**4.3 Диаграмма декомпозиции А2**

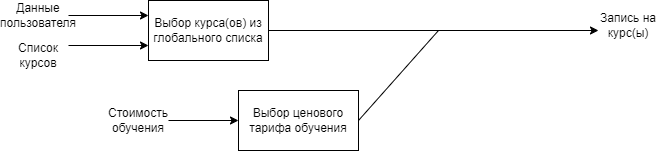


Рисунок 3 – запись на курс(ы)

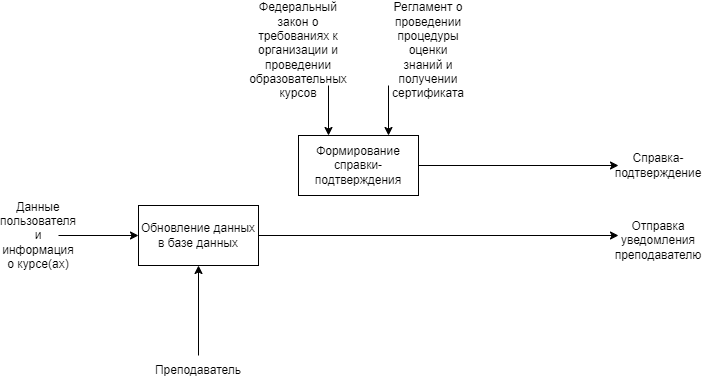


Рисунок 4 – обновление данных в базе данных и формирование справки-подтверждения

**Раздел 5: проектирования функциональной структуры программного продукта; объектно-ориентированный подход**

**5.1 Описание абстракций и их представление**

Проектируемая программная система направлена на создание веб-портала, на котором пользователи могут ознакомиться с каталогом и записаться на курс(ы). Таким образом, основными абстракциями могут являться “Пользователь”, “Курс”, “Заявка записи на курс(ы)”, “Преподаватель”, “Уведомление о записи на курс(ы)”. Абстракция “Пользователь” является основной, так как все остальные (кроме “Преподавателя”) появляются в результате её действий.

Данные абстракции можно разбить на 3 группы и представить в виде таблицы 1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Абстракция | Тип | Класс | Описание |
| 1 | Пользователь | Абстракция сущности | Люди | Основная абстракция, приводит к формированию остальных |
| 2 | Курс | Абстракция сущности | Предметы | Абстракция(и), из которой(ых) формируется абстракция “Заявка записи на курс(ы)” |
| 3 | Заявка записи на курс(ы) | Абстракция поведения | События | Абстракция, которая приводит к изменению базы данных |
| 4 | Преподаватель | Абстракция сущности | Люди | Абстракция, которая получает “Уведомление о записи на курс(ы)” |
| 5 | Уведомление о записи на курс(ы) | Абстракция интерфейса | Концепции | Абстракция, которая нужна для информирования абстракции “Преподаватель” |

Таблица 1 – абстракции подсистемы

Так же можно сформировать возможное поведение каждой абстракции в проектируемой ПС и представить в виде таблицы 2.

|  |  |  |  |
| --- | --- | --- | --- |
| № | Абстракция | Требование согласно модели UML | Описание поведения |
| 1 | Пользователь | Наличие персональных данных | Регистрация, внесение персональных данных, вход в систему, просмотр и запись на курсы |
| 2 | Курс | Наличие цены (плана обучения) | Добавление/удаление курсов |
| 3 | Заявка записи на курс(ы) | Содержит данные пользователя и выбранный(ые) курс(ы) | Создаётся автоматически, изменяет информацию в базе данных |
| 4 | Преподаватель | Наличие списка преподаваемых курсов | Просмотр списка курсов, оценивание слушателей(пользователей) |
| 5 | Уведомление о записи на курс(ы) | Содержит данные пользователя и выбранный(ые) курс(ы) | Создаётся автоматически и отправляется преподавателю |

Таблица 2 – абстракции подсистемы и их поведение

**5.2 Диаграмма классов UML**

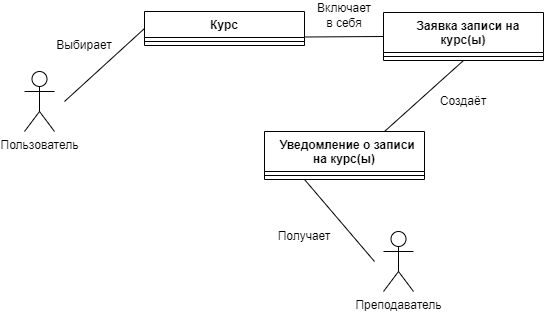


Рисунок 1 – диаграмма классов UML

Теперь можно попробовать выявить атрибуты и операции соответствующих классов, которые можно представить в виде таблицы 3.

|  |  |  |
| --- | --- | --- |
| Имя класса | Список атрибутов | Список операций |
| Курс | Название  Стоимость  Описание | Создать()  Удалить() |
| Заявка записи на курс(ы) | Данные пользователя  Статус | Создать()  Отправить() |
| Уведомление записи на курс(ы) | Данные пользователя  Адрес получателя | Создать()  Отправить() |

Таблица 3 – атрибуты и операции классов

Класс “Заявка записи на курс(ы)” является классом с наиболее частой сменой состояний, потому что на этот класс оказывает влияния наибольшее число внешних факторов, так что для него нужно построить диаграмму состояний.

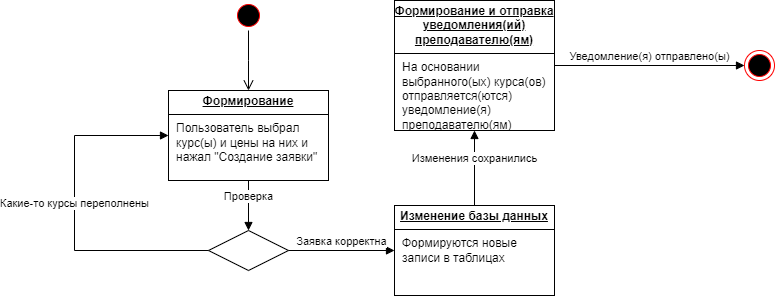


Рисунок 2 – диаграмма состояний класса “Заявка записи на курс(ы)”

**5.3 Диаграмма деятельности**

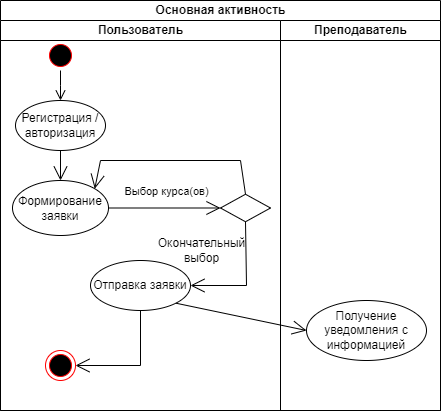


Рисунок 3 – диаграмма деятельности

В таблице 4 представлены объекты, участвующие в данном прецеденте, и их описание.

|  |  |  |  |
| --- | --- | --- | --- |
| № | Прецедент | Объект | Описание объекта |
| 1 | Оформление и отправка заявки записи на курс(ы) | Пользователь | Инициатор данного прецедента |
| Преподаватель | Участник прецедента, который не принимает активных действий |
| Заявка записи на курс(ы) | Первый результат прецедента |
| Уведомление о записи на курс(ы) | Второй результат прецедента |

Таблица 4 – описание прецедента

Для рассмотрения жизненного цикла всего прецедента нужно составить диаграмму последовательности.

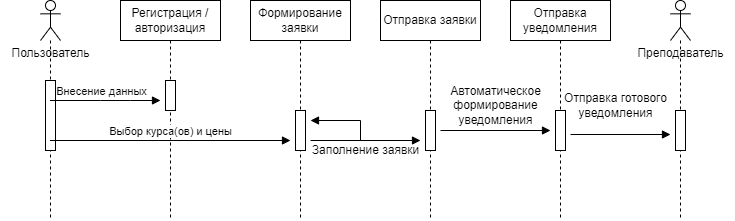


Рисунок 4 – диаграмма последовательности

**Раздел 6: проектирование базы данных программной системы**

**6.1 Идентификация сущностей**

На основе модели классов UML, представленной в лабораторной работе 5, была произведена идентификация сущностей информационной базы ПС (таблица 1) и связей между ними (рисунок 1), а также логическая модель данных (таблица 2).

|  |  |
| --- | --- |
| Документ | Сущность |
| Слушатель | Слушатель курсов |
| Академический предмет | Преподаваемый предмет |
| Преподаватель | Преподаватель |
| Журнал успеваемости | Журнал успеваемости слушателей |

Таблица 1 – идентификация сущностей



Рисунок 1 – связи между сущностями

|  |  |  |
| --- | --- | --- |
| Сущность | Атрибуты | Описание |
| Справочные | | |
| Слушатели | ID слушателя, фамилия, имя, отчество, дата рождения, телефон, почта | Данные слушателей курсов |
| Академические предметы | ID предмета, название предмета | Список всех академических предметов |
| Преподаватели | ID преподавателя, фамилия, имя, отчество, телефон, почта, квалификация, опыт работы, ID академического предмета | Данные преподавателей курсов |
| Оперативные | | |
| Журнал успеваемости | ID записи, ID слушателя, ID преподавателя, оценка за первый тест, оценка за второй тест, оценка за третий тест, оценка за экзамен | Информация об успеваемости всех слушателей |

Таблица 2 – логическая модель данных

**6.2 ER-диаграммы логической и физической модели данных**

ER-диаграмма логической модели данных представлена на рисунке 2.

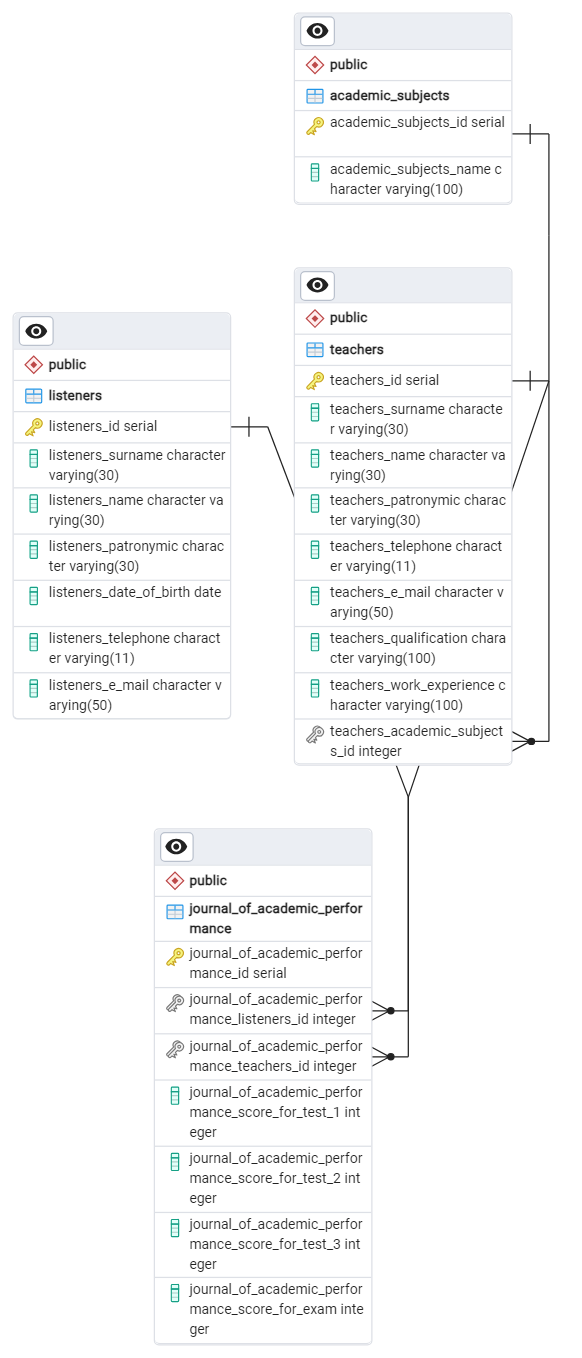


Рисунок 2 – ER-диаграмма логической модели данных

После изучения логической модели данных выяснилось, что нормализация сущностей не требуется.

ER-диаграмма физической модели данных представлена на рисунке 3.

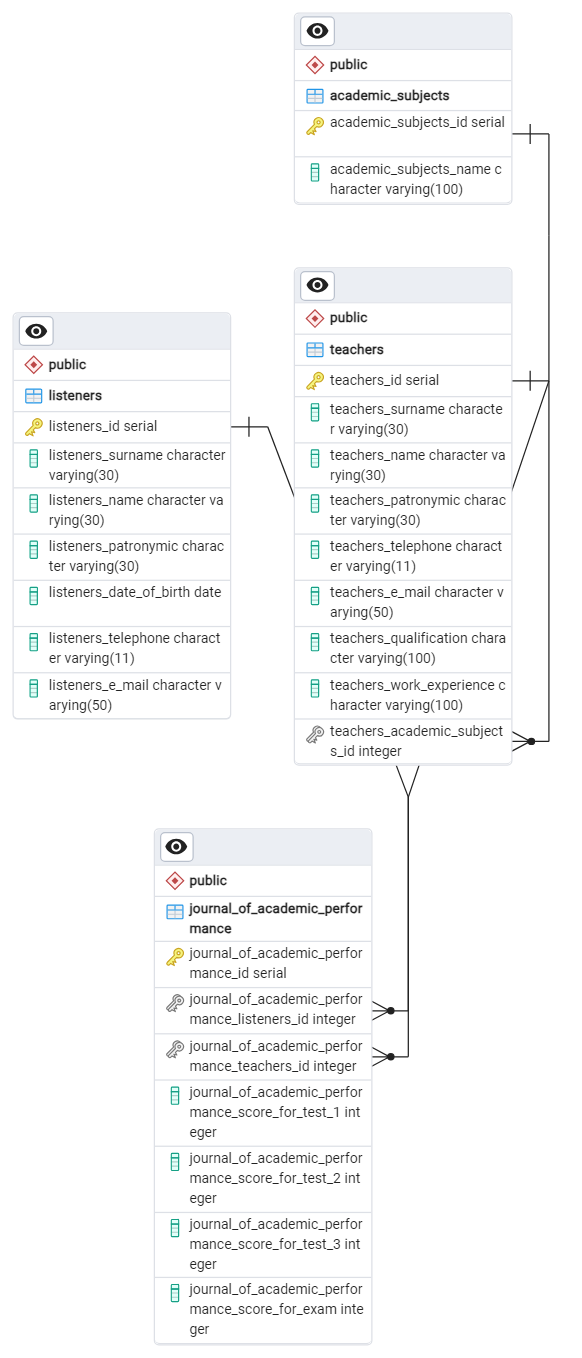


Рисунок 3 – ER-диаграмма физической модели данных

|  |  |  |
| --- | --- | --- |
| Сущность | Атрибуты | Описание |
| Справочные | | |
| Слушатели | ID слушателя, фамилия, имя, отчество, дата рождения, телефон, почта | Данные слушателей курсов |
| Академические предметы | ID предмета, название предмета | Список всех академических предметов |
| Преподаватели | ID преподавателя, фамилия, имя, отчество, телефон, почта, квалификация, опыт работы, ID академического предмета | Данные преподавателей курсов |
| Оперативные | | |
| Журнал успеваемости | ID записи, ID слушателя, ID преподавателя, оценка за первый тест, оценка за второй тест, оценка за третий тест, оценка за экзамен | Информация об успеваемости всех слушателей |

Таблица 3 – физическая модель данных

**6.3 SQL-код базы данных**

Код, реализующий базу данных, был написан в pgAdmin4.

CREATE TABLE academic\_subjects

(

academic\_subjects\_id SERIAL NOT NULL PRIMARY KEY,

academic\_subjects\_name VARCHAR(100) NOT NULL

);

CREATE TABLE listeners

(

listeners\_id SERIAL NOT NULL PRIMARY KEY,

listeners\_surname VARCHAR(30) NOT NULL,

listeners\_name VARCHAR(30) NOT NULL,

listeners\_patronymic VARCHAR(30) NOT NULL,

listeners\_date\_of\_birth DATE NOT NULL,

listeners\_telephone VARCHAR(11) NOT NULL,

listeners\_e\_mail VARCHAR(50) NOT NULL

);

CREATE TABLE teachers

(

teachers\_id SERIAL NOT NULL PRIMARY KEY,

teachers\_surname VARCHAR(30) NOT NULL,

teachers\_name VARCHAR(30) NOT NULL,

teachers\_patronymic VARCHAR(30) NOT NULL,

teachers\_telephone VARCHAR(11) NOT NULL,

teachers\_e\_mail VARCHAR(50) NOT NULL,

teachers\_qualification VARCHAR(100) NOT NULL,

teachers\_work\_experience VARCHAR(100) NOT NULL,

teachers\_academic\_subjects\_id INT NOT NULL,

CONSTRAINT teachers\_academic\_subjects\_id\_fk FOREIGN KEY (teachers\_academic\_subjects\_id) REFERENCES academic\_subjects (academic\_subjects\_id) ON DELETE CASCADE

);

CREATE TABLE journal\_of\_academic\_performance

(

journal\_of\_academic\_performance\_id SERIAL NOT NULL PRIMARY KEY,

journal\_of\_academic\_performance\_listeners\_id INT NOT NULL,

journal\_of\_academic\_performance\_teachers\_id INT NOT NULL,

journal\_of\_academic\_performance\_score\_for\_test\_1 INT,

journal\_of\_academic\_performance\_score\_for\_test\_2 INT,

journal\_of\_academic\_performance\_score\_for\_test\_3 INT,

journal\_of\_academic\_performance\_score\_for\_exam INT,

CONSTRAINT journal\_of\_academic\_performance\_listeners\_id\_fk FOREIGN KEY (journal\_of\_academic\_performance\_listeners\_id) REFERENCES listeners (listeners\_id) ON DELETE CASCADE,

CONSTRAINT journal\_of\_academic\_performance\_teachers\_id\_fk FOREIGN KEY (journal\_of\_academic\_performance\_teachers\_id) REFERENCES teachers (teachers\_id) ON DELETE CASCADE

);

**6.4 Выбор СУБД**

Для реализации задуманного проекта была выбрана именно СУБД PostgreSQL по следующим причинам:

– бесплатная реляционная СУБД с открытым исходным кодом;

– поддерживает расширяемость и имеет богатый набор типов данных;

– хорошо масштабируется и не имеет завышенных требований по производительности;

– поддерживается на всех операционных системах;

– имеет удобный интерфейс pgAdmin для разработки.

**6.5 Описание таблиц и связей между ними**

Описание таблиц и связей между ними приведено в таблицах 4 – 6.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Таблица | Имя поля | Тип данных | Размер, байт | Условие на значение | Значение по умолчанию | Примечание |
| listeners | listeners\_id | SERIAL | - | NOT NULL | - | - |
| listeners\_surname | VARCHAR | 30 | NOT NULL | - | - |
| listeners\_name | VARCHAR | 30 | NOT NULL | - | - |
| listeners\_patronymic | VARCHAR | 30 | NOT NULL | - | - |
| listeners\_date\_of\_birth | DATE | - | NOT NULL | - | - |
| listeners\_telephone | VARCHAR | 11 | NOT NULL | - | - |
| listeners\_e\_mail | VARCHAR | 50 | NOT NULL | - | - |
| academic\_subjects | academic\_subjects\_id | SERIAL | - | NOT NULL | - | - |
| academic\_subjects\_name | VARCHAR | 100 | NOT NULL | - | - |
| teachers | teachers\_id | SERIAL | - | NOT NULL | - | - |
| teachers\_surname | VARCHAR | 30 | NOT NULL | - | - |
| teachers\_name | VARCHAR | 30 | NOT NULL | - | - |
| teachers\_patronymic | VARCHAR | 30 | NOT NULL | - | - |
| teachers\_telephone | VARCHAR | 11 | NOT NULL | - | - |
| teachers\_e\_mail | VARCHAR | 50 | NOT NULL | - |  |
| teachers\_qualification | VARCHAR | 100 | NOT NULL | - | - |
| teachers\_work\_experience | VARCHAR | 100 | NOT NULL | - | - |
| teachers\_academic\_subjects\_id | INT | - | NOT NULL | - | - |
| journal\_of\_academic\_performance | journal\_of\_academic\_performance\_id | SERIAL | - | NOT NULL | - | - |
| journal\_of\_academic\_performance\_listeners\_id | INT | - | NOT NULL | - | - |
| journal\_of\_academic\_performance\_teachers\_id | INT | - | NOT NULL | - | - |
| journal\_of\_academic\_performance\_score\_for\_test\_1 | INT | - | - | - | Может быть пустым |
| journal\_of\_academic\_performance\_score\_for\_test\_2 | INT | - | - | - | Может быть пустым |
| journal\_of\_academic\_performance\_score\_for\_test\_3 | INT | - | - | - | Может быть пустым |
| journal\_of\_academic\_performance\_score\_for\_exam | INT | - | - | - | Может быть пустым |

Таблица 4 – описание структуры таблиц

|  |  |  |
| --- | --- | --- |
| № | Имя таблицы | Описание |
| 1 | listeners | Описывает сущность слушателя |
| 2 | academic\_subjects | Описывает сущность учебного предмета |
| 3 | teachers | Описывает сущность преподавателя |
| 4 | journal\_of\_academic\_performance | Описывает сущность журнала успеваемости |

Таблица 5 – описание списка таблиц

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Родительская таблица | | Дочерняя таблица | | Тип связи |
| Название | Атрибут | Название | Атрибут |
| academic\_subjects | academic\_subjects\_id | teachers | teachers\_academic\_subjects\_id | Один-ко-многим |
| listeners | listeners\_id | journal\_of\_academic\_performance | journal\_of\_academic\_performance\_listeners\_id | Один-ко-многим |
| teachers | teachers\_id | journal\_of\_academic\_performance | journal\_of\_academic\_performance\_teachers\_id | Один-ко-многим |

Таблица 6 – описание связей таблиц

**Раздел 7: разработка программного обеспечения ПС**

**7.1 Выбор средств разработки специального ПО и схема общесистемного ПО**

Выбор средств разработки был подробно изложен в лабораторной работе №3, так что перечислим выбранные средства разработки: IDE Visual Studio Code, IDE PyCharm, СУБД PostgreSQL (pgAdmin4), используемые ЯП – HTML, CSS, JavaScript, библиотека для JavaScript – React, фреймворк Django Rest Api Framework.

Структурная схема разработанного ПО представлена на рисунке 1.

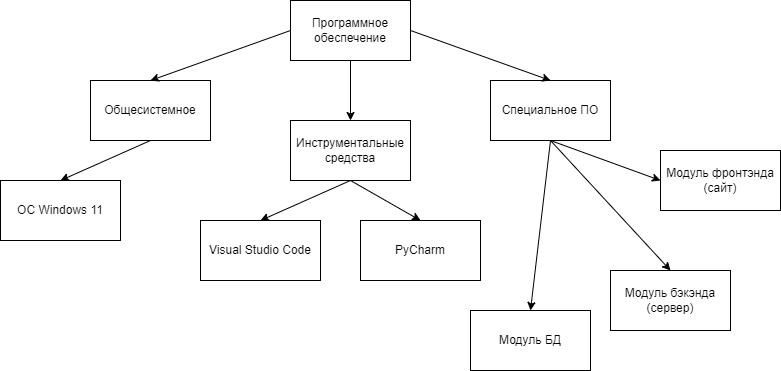


Рисунок 1 – структурная схема ПО

**7.2 Разработанное специальное программное обеспечение ПС**

Разработанное специальное программное обеспечение является веб-сервисом: объект информатизации – образовательный центр "ТехноКод". Информатизируемый бизнес-процесс - "Запись на курсы". На рисунках 2-10 представлен пользовательский интерфейс разработанного ПС, а также фрагмент базы данных.

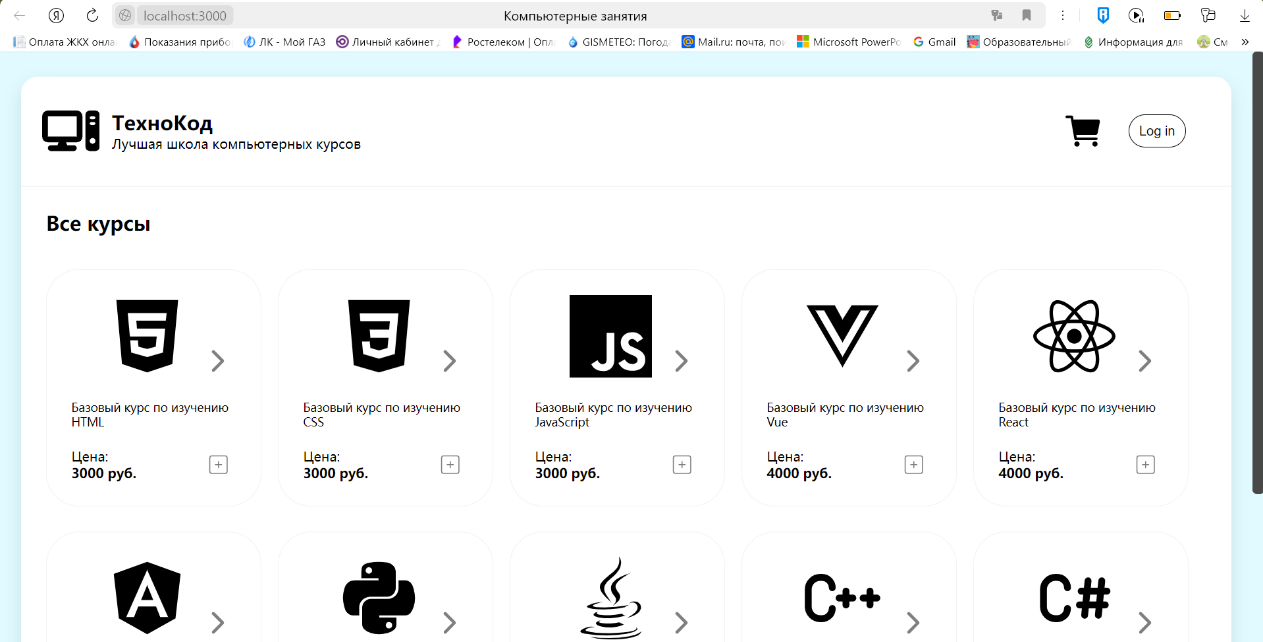


Рисунок 2 – вид главной страницы сайта

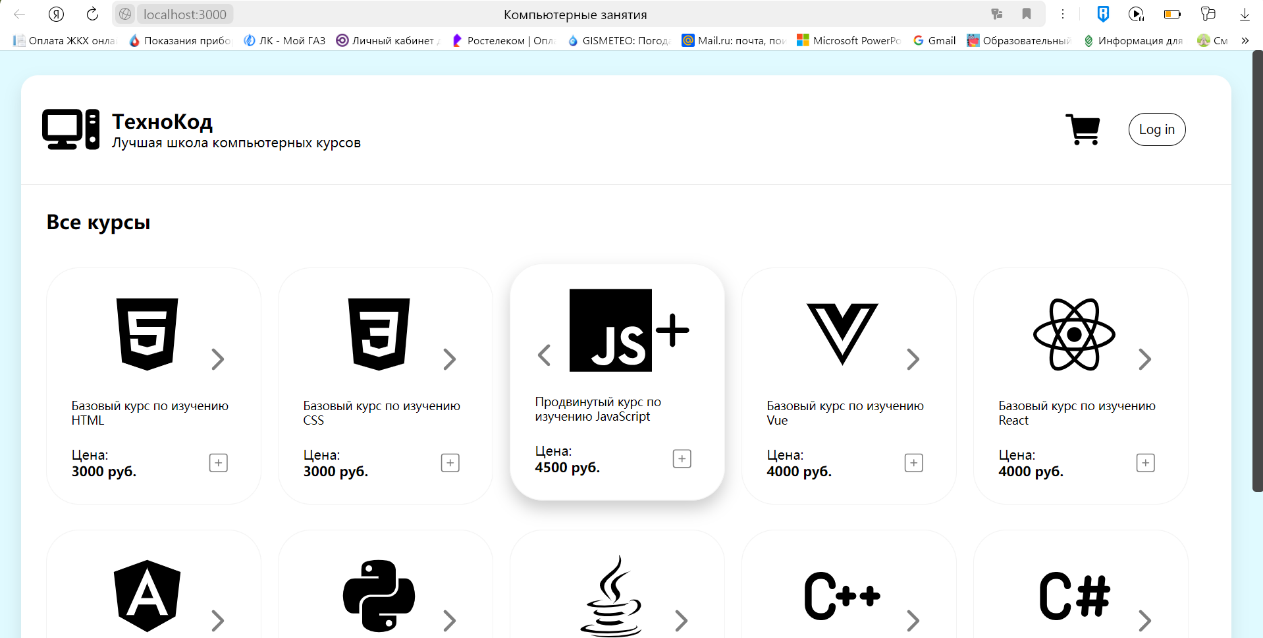


Рисунок 3 – вид главной страницы сайта (изменённая цена курса)

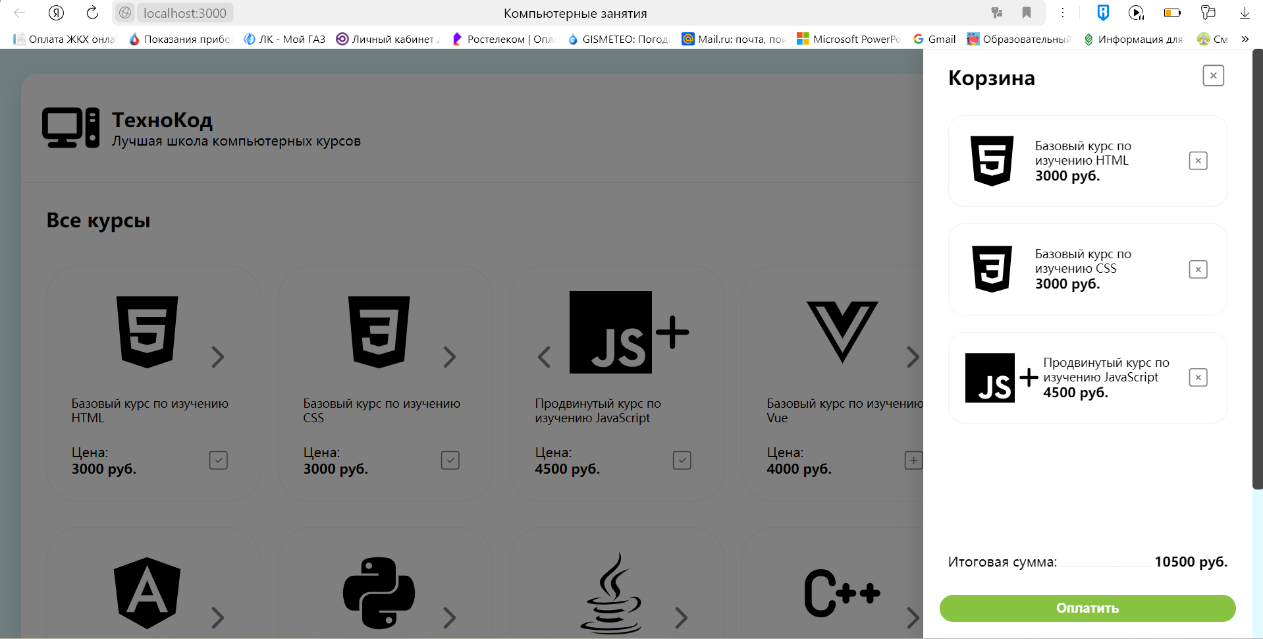


Рисунок 4 – вид корзины сайта

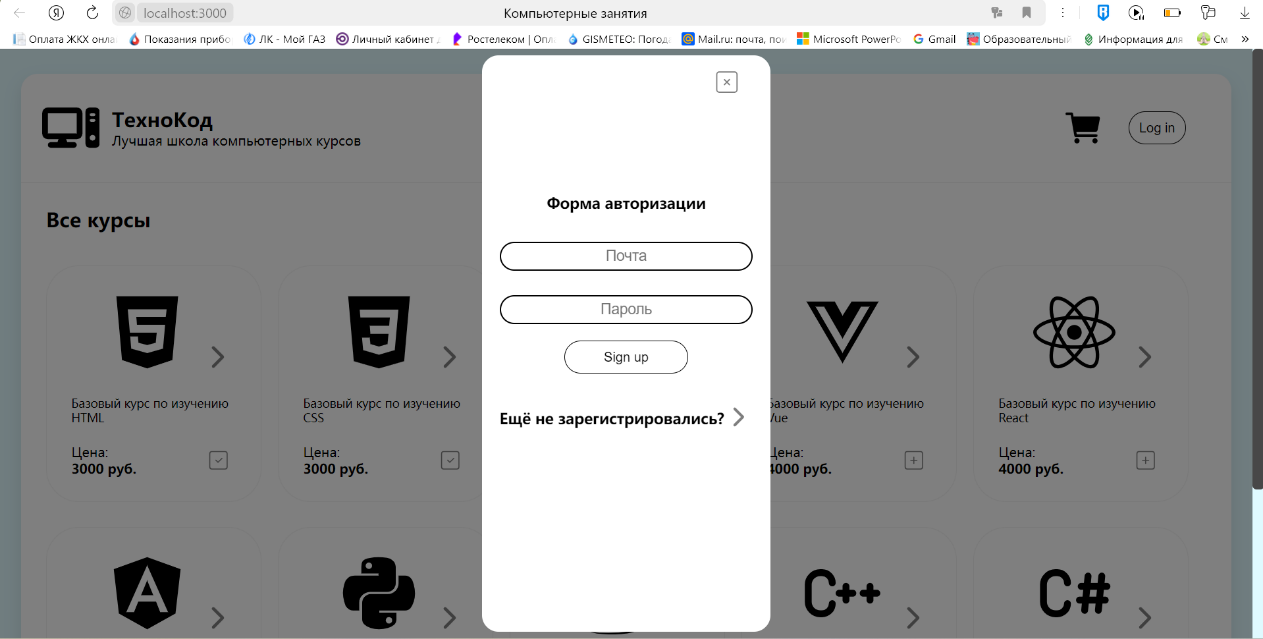


Рисунок 5 – вид формы авторизации

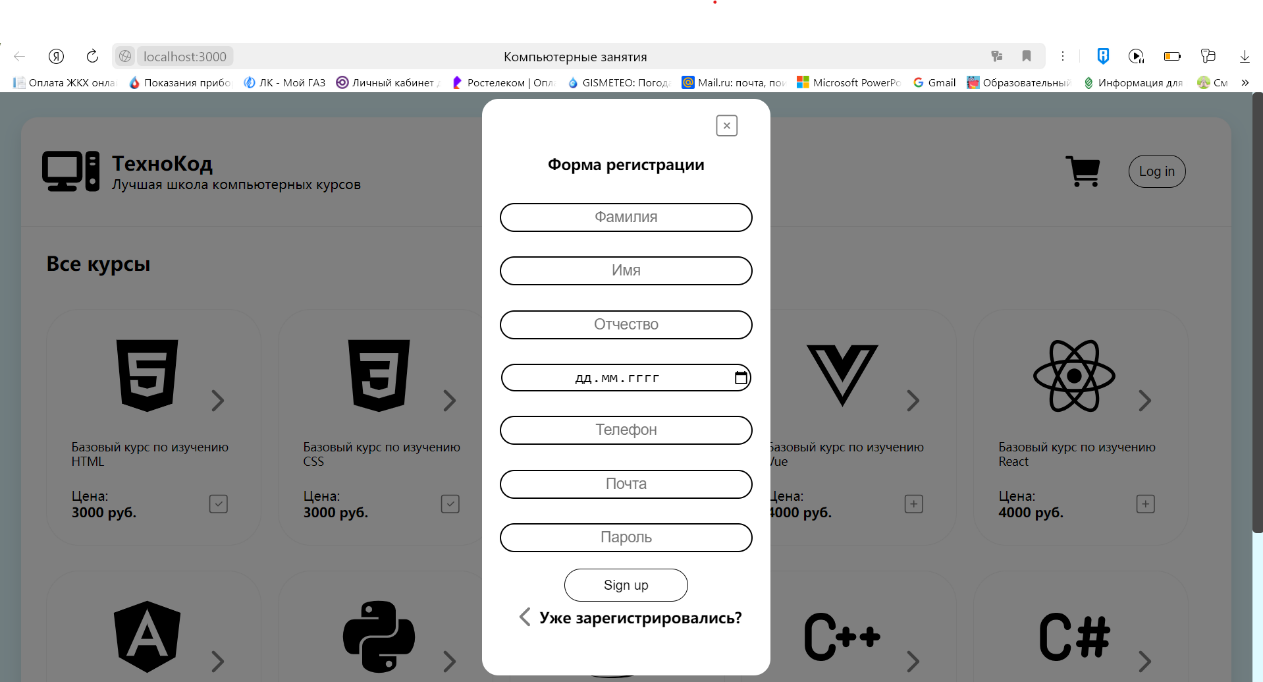


Рисунок 6 – вид формы регистрации

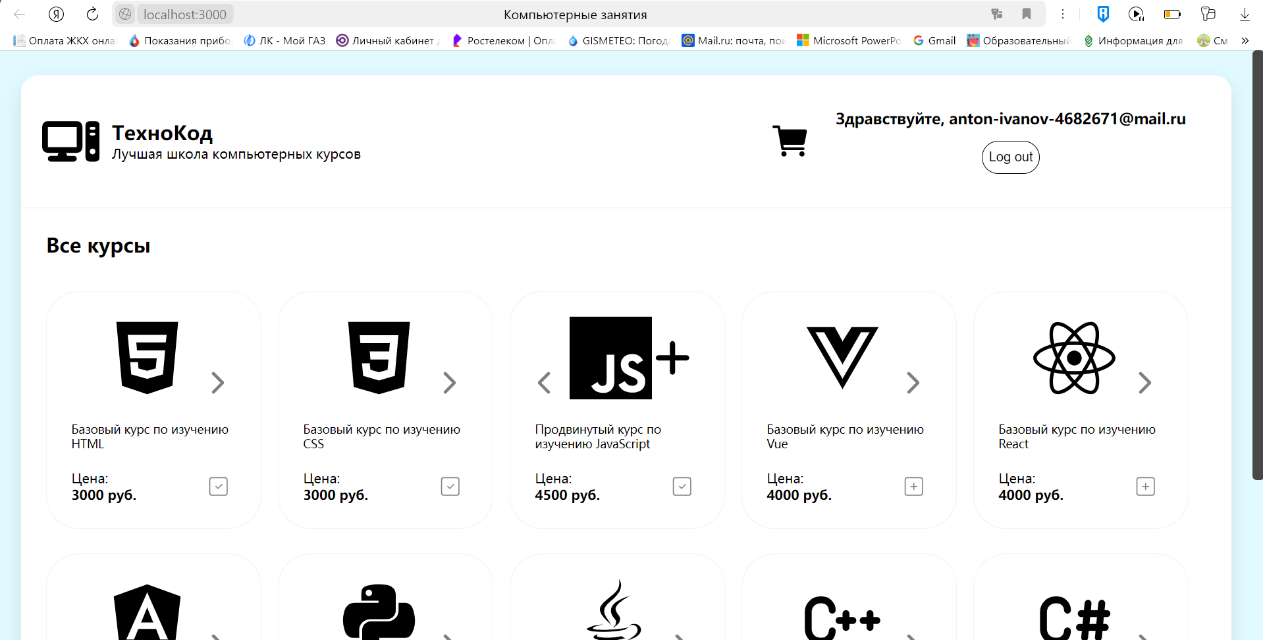


Рисунок 7 – вид главной страницы сайта после входа в систему

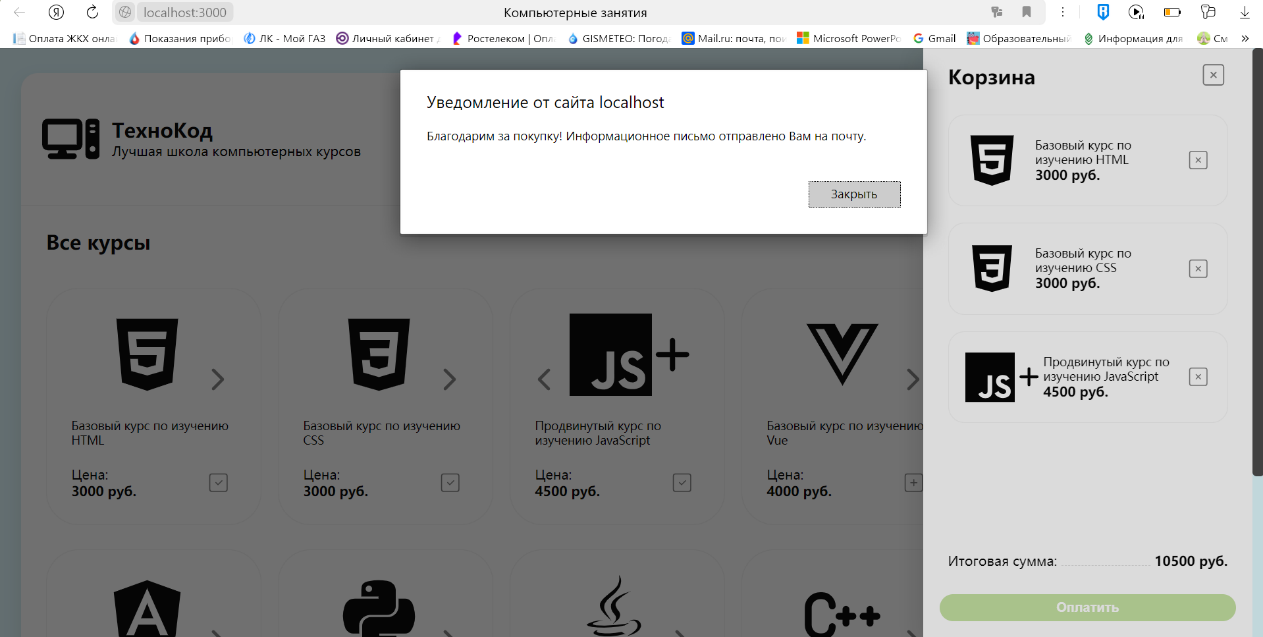


Рисунок 8 – вид уведомления после покупки

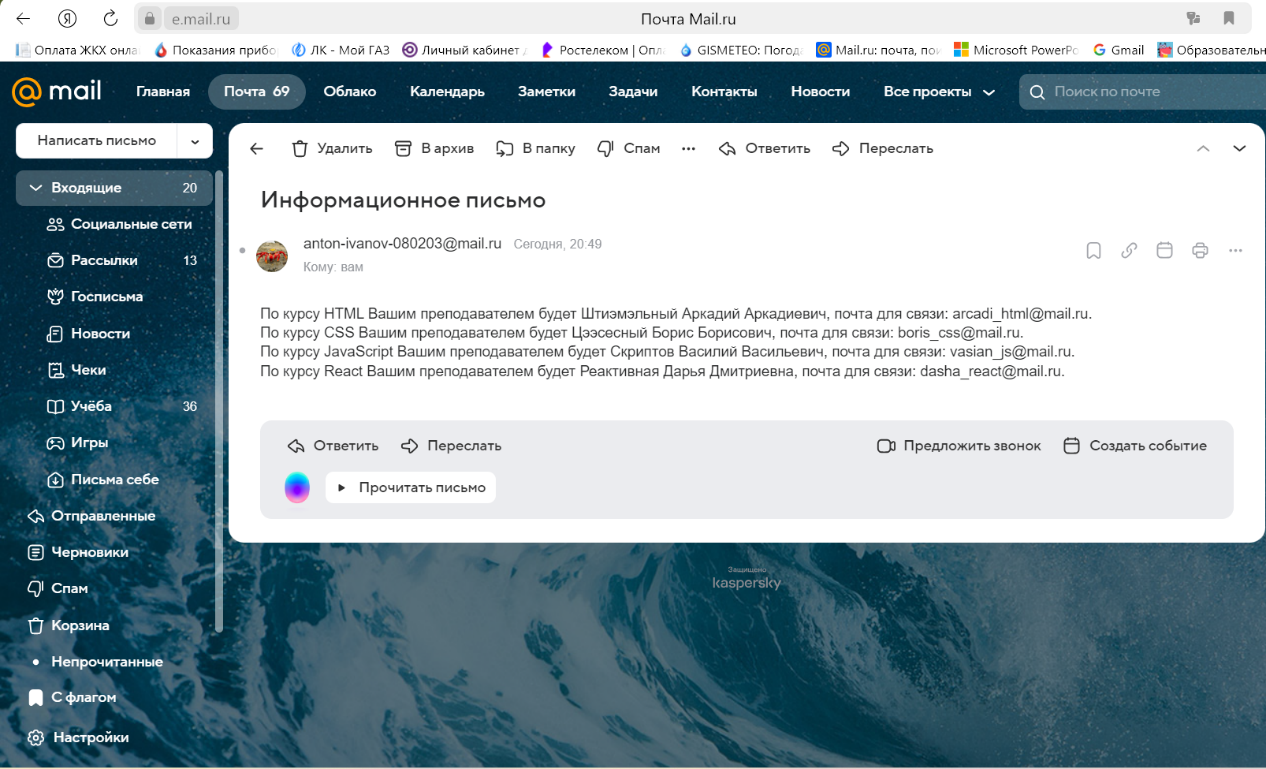


Рисунок 9 – вид информационного сообщения



Рисунок 10 – вид таблицы пользователей

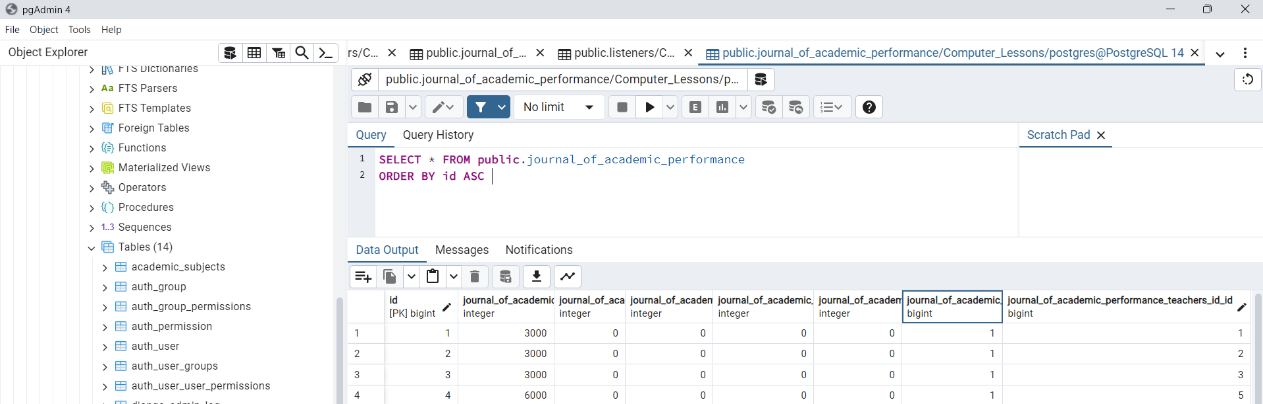


Рисунок 11 – вид таблицы журнала успеваемости

**7.3 Описание разработанных компонентов приложения**

В таблице 1 перечислены компоненты приложения (файлы).

|  |  |  |  |
| --- | --- | --- | --- |
| № | Имя | Тип | Описание |
| 1 | index.js | source | Базовый файл-скрипт, который отвечает за запуск других компонентов-скриптов |
| 2 | index.css | source | Базовый файл стилей, отвечает за стилизацию объектов в App.js |
| 3 | App.js | source | Главный файл-скрипт, который управляет рендерингом прочих компонентов |
| 4 | Lesson.js | source | Компонент, который отвечает за реализацию курсов как карточек на сайте |
| 5 | Lesson.module.css | source | Файл стилей, отвечает за стилизацию объектов в Lesson.js |
| 6 | Header.js | source | Компонент, который отвечает за реализацию шапки сайта (панель управления) |
| 7 | Header.module.css | source | Файл стилей, отвечает за стилизацию объектов в Header.js |
| 8 | Drawer.js | source | Компонент, который отвечает за реализацию корзины, в которой отображаются выбранные курсы |
| 9 | Drawer.module.css | source | Файл стилей, отвечает за стилизацию объектов в Drawer.js |
| 10 | Form.js | source | Компонент, который отвечает за реализацию форм авторизации и регистрации |
| 11 | Form.module.js | source | Файл стилей, отвечает за стилизацию объектов в Form.js |
| 12 | index.html | source | Базовый файл вёрстки, который отвечает за общее отображение страницы |
| 13 | settings.py | source | Главный файл настроек Django-сервера, отвечает за подключение к БД, настройку отправки сообщений, настройку cors-заголовков |
| 14 | models.py | source | Главный файл для работы с моделями таблиц базы данных |
| 15 | urls.py | source | Главный файл, который отвечает за переадресацию запросов (направляет запрос на корректную функцию-обработчик) |
| 16 | views.py | source | Главный файл, который реализует весь функционал сервера (содержит все функции для работы с запросами) |
| 17 | База данных | block | Блок, который отвечает за хранение всех данных |

**7.4 Диаграмма компонентов**

Связь компонентов представлена на рисунке 12.

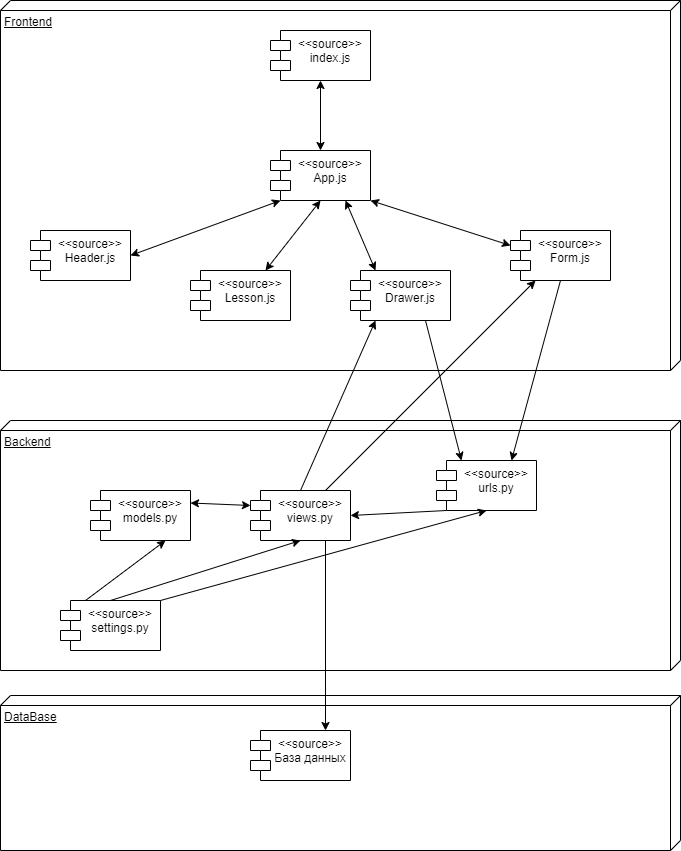


Рисунок 12 – связь компонентов

**7.5 Запуск и проверка приложения**

После окончания разработки приложения было запущено и протестировано. Ошибки, который были обнаружены, были исправлены. На данном этапе разработки приложение выполняет свой заявленный спектр задач.

**7.6 Сложность кода приложения**

Данное приложение представляет собой архитектуру средней сложности из-за выбранного стека технологий, однако подвержено улучшению и расширению.

В разработанном приложении:

– 16 основных файлов (в самом приложении несколько тысяч файлов, однако 16 указанных – те, в которых писался программный код, остальные же файлы представляют собой библиотеки и базовые файлы при создании React- и Django-приложений);

– 100 (примерно) объектов (те объекты и переменные, который были явно прописаны в вышеуказанных 16 файлах);

– 12 состояний рендеринга;

– 22 функции.

**7.7 Диаграмма развёртывания**

Диаграмма развёртывания разработанного ПО указана на рисунке 13.

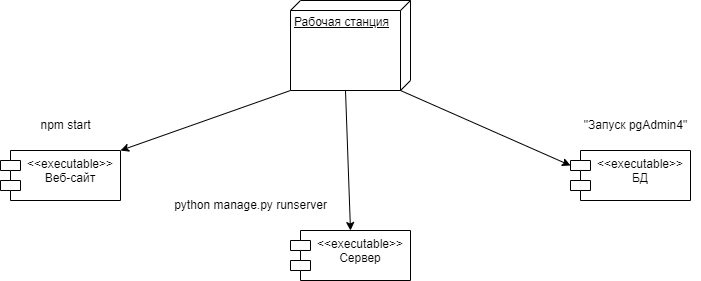


Рисунок 13 – диаграмма развёртывания ПО

**Раздел 8: назначение, технические характеристики, принцип работы и меры безопасности при эксплуатации ПС**

**8.1 Назначение**

Разработанное программное средство “ТехноКод” является веб-сервисом для покупки курсов с последующим получением уведомления. На данном этапе разработки программное средство выполняет поставленные задачи, однако может использоваться только как тренировочный макет приложения, потому что не обладает полным функционалом школы онлайн-курсов.

**8.2 Технические характеристики**

1. Архитектура ПС:

ПС представляет собой 2 приложения (веб-сайт и сервер), которые обмениваются между собой запросами, а также модуль базы данных для хранения данных. Используемые технологии: HTML, CSS, JavaScript, Python, React, Django Rest Api Framework.

2. Функциональные возможности:

Функционал, доступный пользователям: регистрация, авторизация, покупка курсов, получение уведомлений на почту.

3. Интеграции:

Интеграция с почтовым сервисом Mail.ru для отправки сообщений пользователям.

4. Безопасность:

Хэширование пароля.

5. Требования к аппаратному и программному обеспечению:

– процессор: 12th Gen Intel(R) Core(TM) i5-12500-H 2.50 GHz

– оперативная память: 16 GB

– дисковое пространство: 1.50 GB

– тип системы: х64

– операционная система: Windows 11

**8.3 Принцип работы**

Принцип работы вышеописанного ПС: запускается сервер и клиентская часть, запускается база данных, к которой выполняется подключение. После выполнения данных действий пользователи могут использовать данный веб-сервис.

**8.4 Меры безопасности при эксплуатации**

Данное приложение не предназначено для коммерческого использования из-за отсутствия некоторого функционала. Из-за этого фактически не представляется возможным получить данные пользователя. Обобщая всё вышесказанное, можно сказать следующее: ПС может использоваться в обучающих целях.

**Раздел 9: руководство пользователя**

1. Регистрация и вход в систему:

При заходе на сайт пользователь оказывается на главной странице. В правом верхнем углу находится кнопка “Sign up”, при нажатии на которую появляется форма авторизации, которую можно сменить на форму регистрации кликом на стрелку “Вправо” внизу формы авторизации.

1. Покупка курсов:

После регистрации/авторизации в системе пользователь может совершить покупку курсов. Для этого необходимо на главной странице выбрать интересующий курс, нажать на кнопку “Добавить” (на сайте изображена в виде плюса в квадратной рамке), после чего выбранный курс появится в корзине.

Чтобы открыть корзину нужно нажать на соответствующую картинку тележки вверху главной страницы. После нажатия в правой части страницы появится корзина, внизу которой есть кнопка “Оплатить”, при нажатии на которую произойдёт оплата курсов.

1. Получение уведомления о купленных курсах:

После того, как пользователь выполнит оплату курсов, на главной странице сайта он увидит уведомление, появляющееся в верхней части страницы (уведомление появляется не сразу, нужно немного подождать). В уведомлении сказано, что на почту пользователю было отправлено информационное письмо (под почтой пользователя имеется ввиду та почта, которую пользователь указывал в форме регистрации и та, под которой пользователь выполняет авторизацию). При переходе на вышеуказанную почту пользователь увидит информационное письмо, в котором будет указана основная информация о купленных курах (ФИО преподавателя, который будет проводить курсы и его почта для связи).