

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Российский экономический университет им. Г.В. Плеханова»
Московский приборостроительный техникум

Курсовой проект

ПМ 02 Осуществление интеграции программных модулей

МДК 02.01 Технология разработки программного обеспечения

Специальность 09.02.07 «Информационные системы и
программирование»

Квалификация: Специалист по тестированию в области информационных
технологий

Тема: «Разработка мобильного приложения для изучения языков
программирования»

Пояснительная записка

Листов: 50

Руководитель

_____ / Г.Ю. Волкова

« ____ » _____ 2024 год

Исполнитель

_____ /З.Р. Закареева

« ____ » _____ 2024 год

ВВЕДЕНИЕ.....	4
1. ОБЩАЯ ЧАСТЬ.....	6
1.1. Цель разработки.....	6
1.2. Средства разработки	6
2. СПЕЦИАЛЬНАЯ ЧАСТЬ	8
2.1. Постановка задачи.....	8
2.1.1. Входные данные	8
2.1.2. Выходные данные	8
2.1.3. Подробные требования к проекту	8
2.2. Внешняя спецификация	9
2.2.1. Описание задачи	9
2.2.2. Потоки данных.....	Ошибка! Закладка не определена.
2.2.3. Диаграмма классов UML	Ошибка! Закладка не определена.
2.2.4. Диаграмма прецедентов.....	Ошибка! Закладка не определена.
2.2.5. Входные и выходные данные.....	13
2.2.6. Методы	15
2.2.7. Тесты.....	17
2.2.8. Контроль целостности данных.....	17
2.3. Проектирование	18
2.3.1. Схема архитектуры приложения	18
2.3.2. Логическая схема данных.....	19
2.3.3. Физическая схема данных	20
2.3.4. Структурная схема	24
2.3.5. Функциональная схема	25
2.3.6. Диаграмма классов	Ошибка! Закладка не определена.
2.3.7. Схема тестирования	25
2.3.8. Схема пользовательского интерфейса	25
2.4. Результат работы программы.....	26
3. ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ	29
3.1. Инструментальные средства	29
3.2. Отладка программы.....	29
3.3. Защитное программирование.....	30
3.4. Характеристики программы.....	32

ЗАКЛЮЧЕНИЕ	33
СПИСОК ИСПОЛЬЗУЕМЫХ МАТЕРИАЛОВ.....	34
ПРИЛОЖЕНИЕ А. Техническое задание	
ПРИЛОЖЕНИЕ Б. Текст программы	
ПРИЛОЖЕНИЕ В. Сценарий и результаты тестовых испытаний	
ПРИЛОЖЕНИЕ Г. Руководство пользователя	
ПРИЛОЖЕНИЕ Д. Скрипт базы данных	

ВВЕДЕНИЕ

Разработка мобильного приложения для изучения языков программирования представляет собой уникальный и актуальный проект, который сочетает в себе возможности обучения и практического применения полученных знаний. В современном мире, где технологии развиваются с невероятной скоростью, способность быстро обучаться новым навыкам, особенно в области программирования, становится не просто ценным активом, а необходимостью. Именно поэтому создание мобильного приложения, нацеленного на изучение языков программирования, является своевременным и важным начинанием.

В основе концепции данного приложения лежит предоставление пользователям доступа к широкому спектру лекций и сопутствующих материалов по различным языкам программирования. От Python и JavaScript до более специализированных, как Haskell или Rust, приложение стремится охватить как популярные, так и нишевые языки, предлагая пользователям всеобъемлющую базу знаний. Особое внимание уделяется не только теоретическому изложению материала, но и практическим аспектам программирования, что позволяет пользователям не только изучать новый материал, но и закреплять его на практике.

Одной из ключевых особенностей приложения является интеграция системы тестирования, позволяющей пользователям проверять свои знания по пройденным темам. Эта функция не только способствует более глубокому усвоению материала, но и мотивирует пользователей продолжать обучение, видя конкретные результаты своих усилий. Тесты разрабатываются таким образом, чтобы оценивать не только запоминание фактов, но и понимание принципов работы языков программирования, а также умение применять полученные знания на практике.

Разработка такого приложения представляет собой немалую задачу, требующую не только глубоких знаний в области программирования, но и понимания методик обучения и психологии. Приложение должно быть не только функциональным, но и удобным в использовании, предлагая пользователю дружелюбный интерфейс и персонализированный путь обучения. Важно учитывать различные уровни подготовки пользователей, предоставляя материалы, подходящие как для начинающих, так и для продвинутых программистов.

1. ОБЩАЯ ЧАСТЬ

1.1. Цель разработки

Цель разработки данного мобильного приложения для изучения языков программирования заключается в предоставлении доступного, эффективного и комплексного средства для самообразования в области программирования. Основная задача — сделать процесс изучения языков программирования максимально гибким, интерактивным и адаптированным под индивидуальные потребности каждого пользователя, независимо от его предыдущего опыта и уровня знаний. Приложение стремится не просто передать теоретические знания, но и обеспечить практическое их применение через систему тестирования и заданий, тем самым поддерживая мотивацию и активное обучение. В конечном итоге, проект направлен на формирование у пользователей сильной основы в программировании, способствуя личному и профессиональному росту в сфере информационных технологий.

1.2. Средства разработки

Для проектирования, разработки и тестирования программного обеспечения, а также оформления документации по проекту использовались программные средства, представленные в таблице 1.

Таблица 1 - Программные средства разработки

№	Тип средства	Название средства	Назначений
1	2	3	4
1	Операционная система	Microsoft Windows 11 PRO 21H2	Организация взаимодействия программ и пользователя
2	Среда разработки мобильного приложения	IntelliJ IDEA 2023.2.1	Разработка приложения для телефонов на базе операционной системы Android и IOS.
3	Построение схем БД	SqlDBM	Построение логической и физической модели базы данных
4	СУБД	Microsoft SQL Server	Создание и манипуляция базой данных, обеспечение безопасности, надежности хранения и целостности данных, предоставления

			средств администрирования БД.
5	Интерфейс взаимодействия	Mircosoft SQL Server Management Studio 18	Для взаимодействия СУБД

В качестве средств вычислительной техники при разработке ПО использовался ноутбук Asus TUF Gaming. Характеристики представлены в таблице 2. В качестве средства для тестирования программы использовался телефон Samsung

Таблица 2 - Технические характеристики устройств используемых при разработке ПО

№	Тип средства	Название средства
1	2	3
Ноутбук Asus TUF Gaming		
1	Размер экрана:	15,6
2	Разрешение экрана:	1920x1080
3	Линейка процессора:	Intel i5 11400f
4	Количество ядер процессора:	6
5	Оперативная память:	16 ГБ
6	Тип видеокарты:	Дискретная
7	Видеокарта:	RTX 3050
8	Конфигурация накопителей:	SSD
9	Общий объем всех накопителей:	2 ТБ
Телефон Samsung		
1	Размер экрана:	6,2
2	Разрешение экрана:	2400x1080
3	Линейка процессора:	Snapdragon 888 Gen 1
4	Количество ядер процессора:	8
5	Оперативная память:	8 ГБ
6	Общий объем всех накопителей:	128 ГБ

2. СПЕЦИАЛЬНАЯ ЧАСТЬ

2.1. Постановка задачи

Разработать мобильное приложение, предназначенное для изучения языков программирования, которое будет служить комплексным инструментом для пользователей на разных уровнях подготовки. Основная задача приложения — предоставление пользователям доступа к широкому спектру учебных материалов, включая лекции, текстовые руководства по различным языкам программирования. В дополнение к обучающим материалам, приложение должно включать систему тестирования для проверки знаний и отслеживания прогресса пользователя.

2.1.1. Входные данные

Входные данные представлены в следующем виде:

- Регистрация пользователя (Фамилия, Имя, Пароль, Повтор пароля, Почта, Курс, Группа);
- Авторизация пользователя (Почта, Пароль);
- Загрузка, обновление, редактирование фотографий (Фотография);
- Редактирование профиля (Имя, Фамилия, Отчество, Почта)
- Тест: (Вопросы, ответы).

2.1.2. Выходные данные

Выходные данные представлены в следующем виде:

- Зарегистрированные пользователи (Фамилия, Имя, Почта)
- Авторизованные пользователи (Фамилия, Имя, Почта)
- Информация о пользователе (Фамилия, Имя, Пароль, Почта)

Подробные требования к проекту

В данной информационной системе должны быть реализованы такие функции как:

- Авторизация пользователя
- Регистрация пользователя
- Выход из приложения
- Изменение персональных данных
- Загрузка фото
- Создание теста
- Изменение теста
- Удаление теста
- Создание курсов
- Изменение курсов
- Удаление курсов
- Создание и сохранение резервных копий базы данных

2.2. Внешняя спецификация

2.2.1. Описание задачи

Мобильное приложение:

При открытии приложения пользователю должен открываться экран авторизации, если пользователь был зарегистрирован ранее, он может войти под своими данными. Если же пользователя нет в системе, он должен пройти по кнопке «регистрация», после чего откроется экран регистрации. После регистрации пользователя перенесет на окно «авторизации» для дальнейшей авторизации. Если ранее пользователь авторизовался в приложении, его автоматически при запуске приложения будет переносить на экран курсов

В нижней части экрана должны быть расположены кнопки, для перемещения по вкладкам приложения (Курсы, Профиль).

Приложения для компьютера:

При открытии приложения пользователю должно быть открыто окно ASP.NET.

В ходе разработки был проведен анализ предметной области, чтобы понять, как организация работает. В результате были выявлены процессы, которые можно автоматизировать. Они представлены на рисунках 1-6.

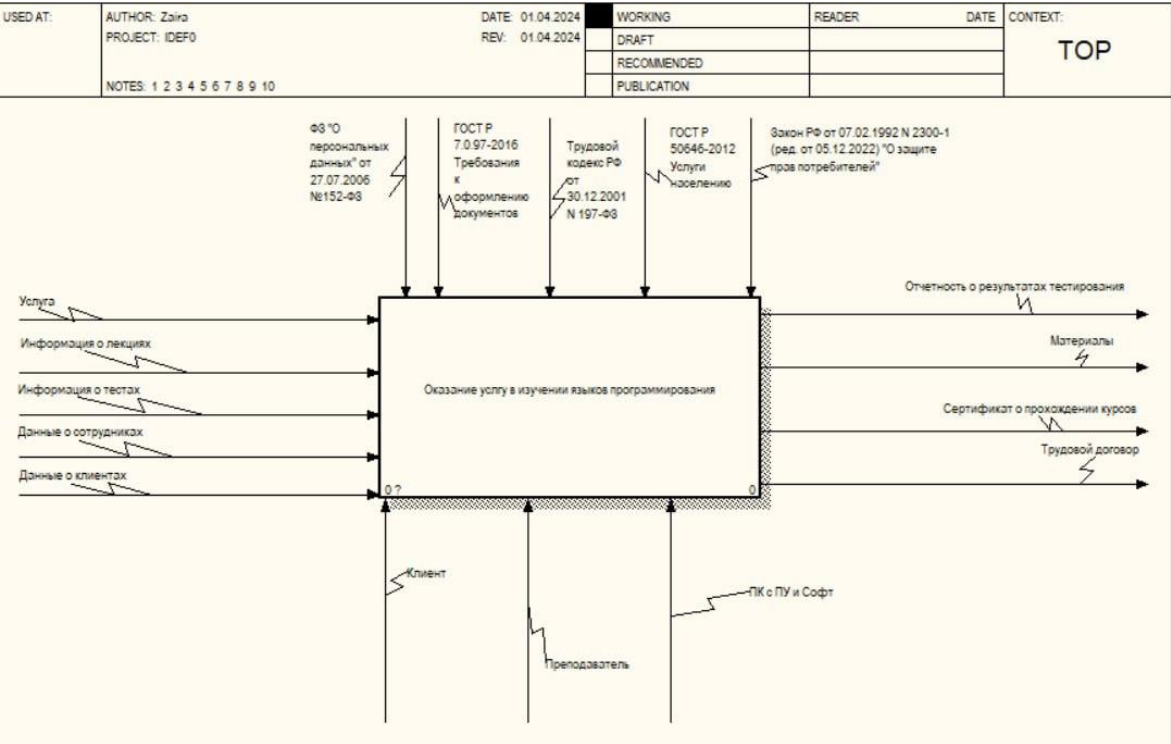


Рисунок 1 IDEF0

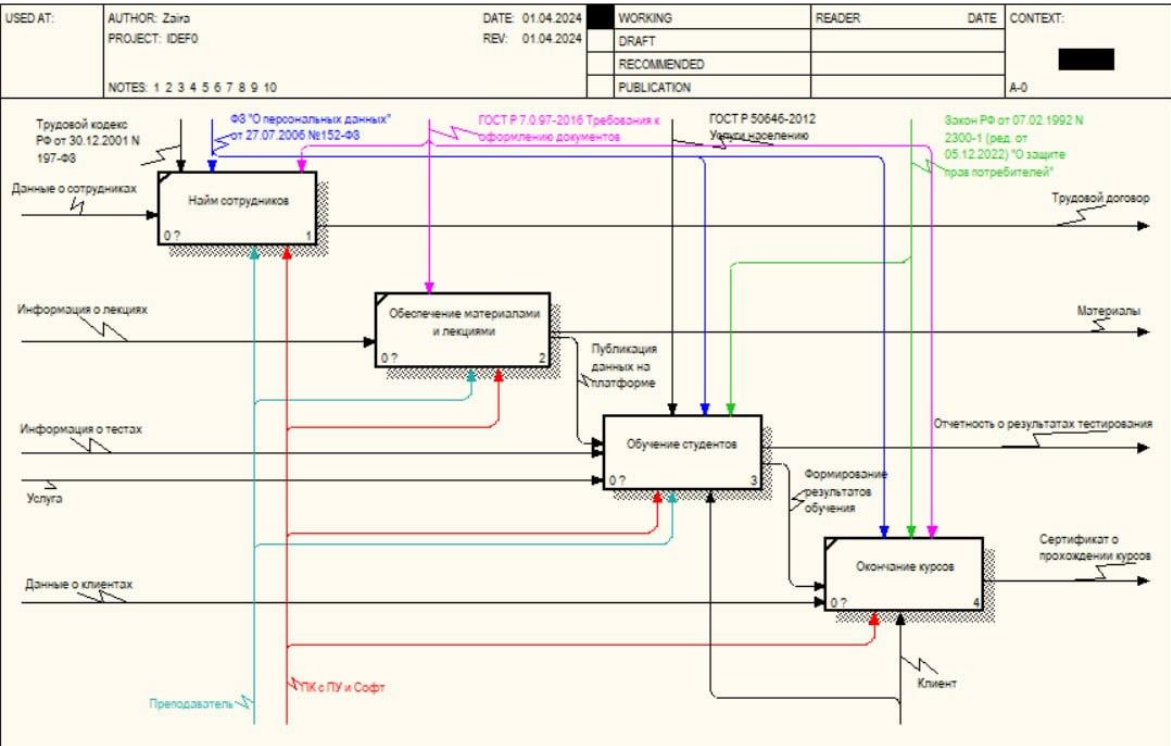


Рисунок 2 Декомпозиция второго уровня IDEF0

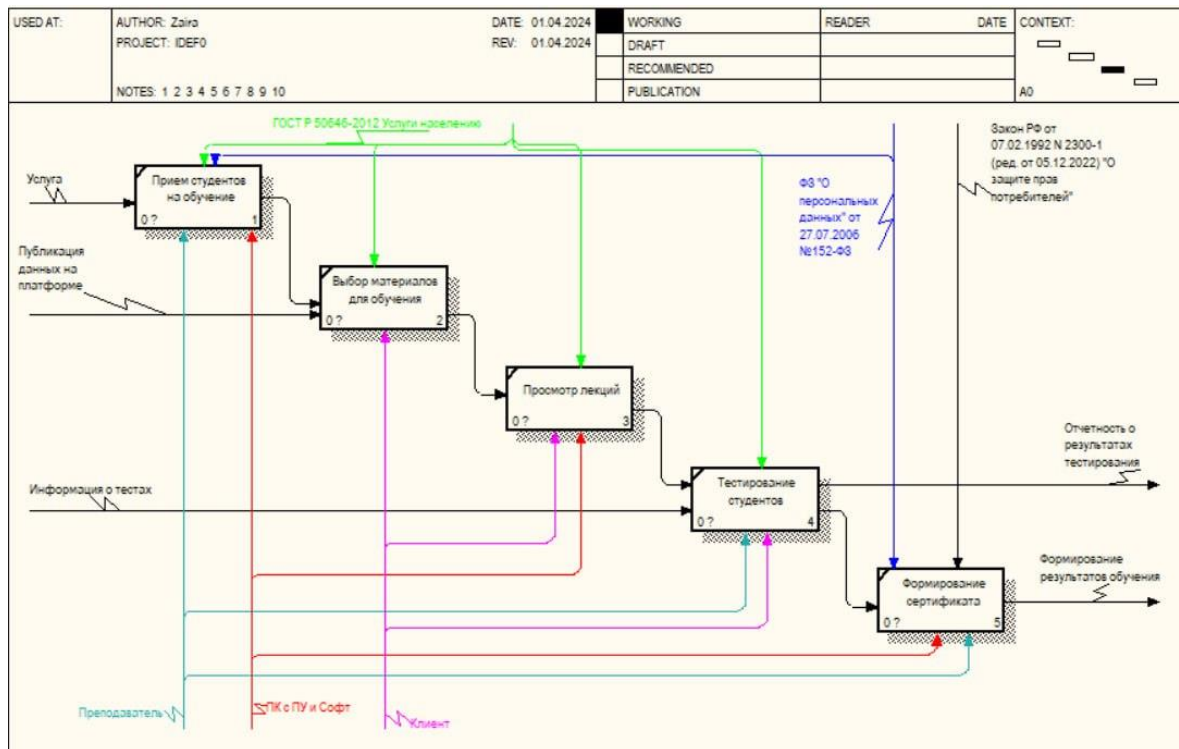


Рисунок 3 Декомпозиция третьего уровня IDEF0

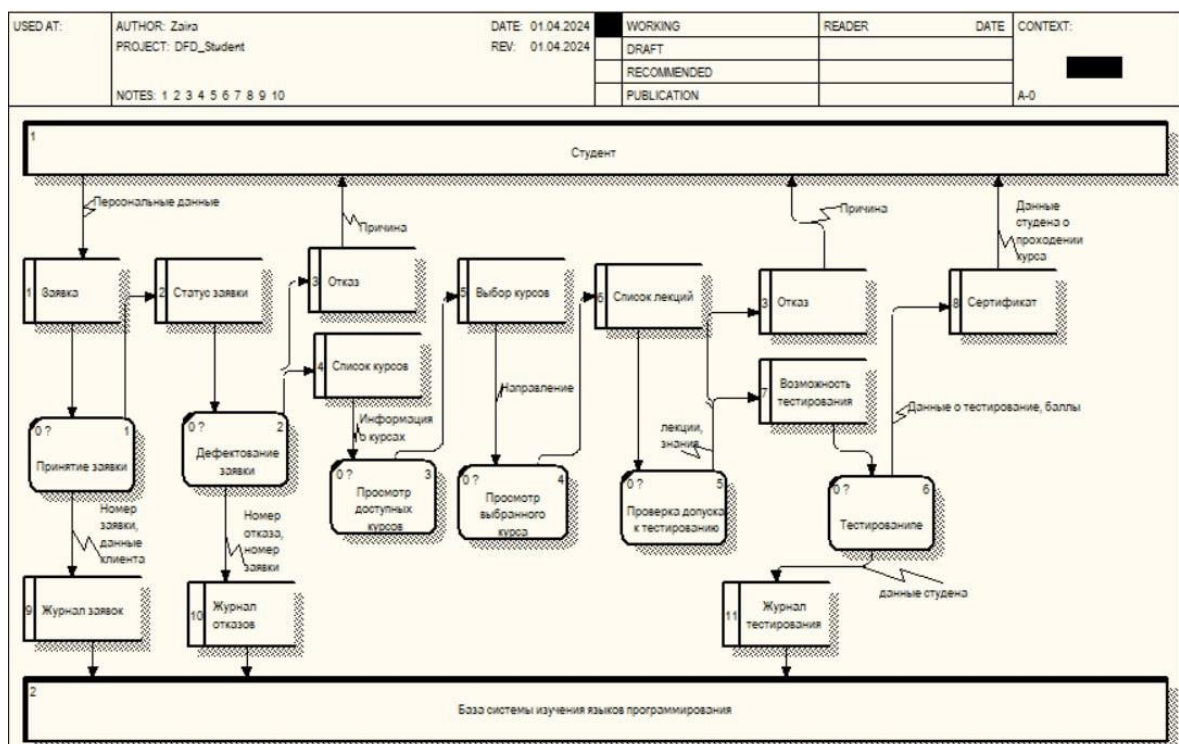
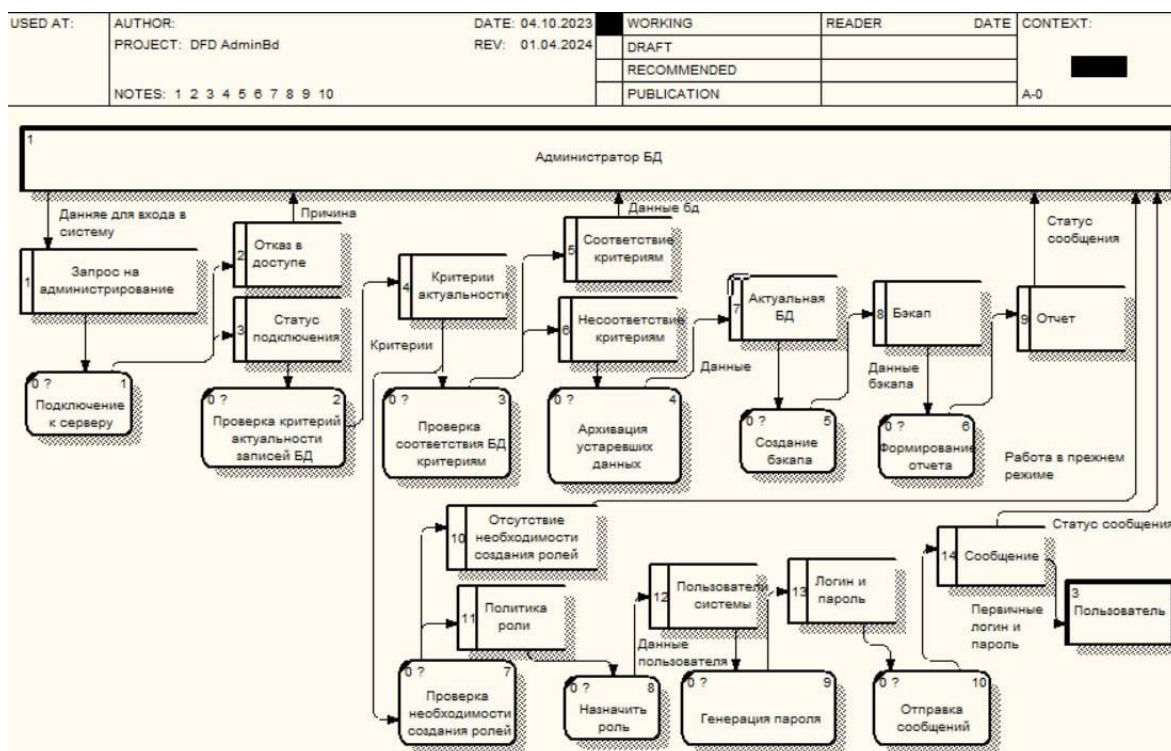
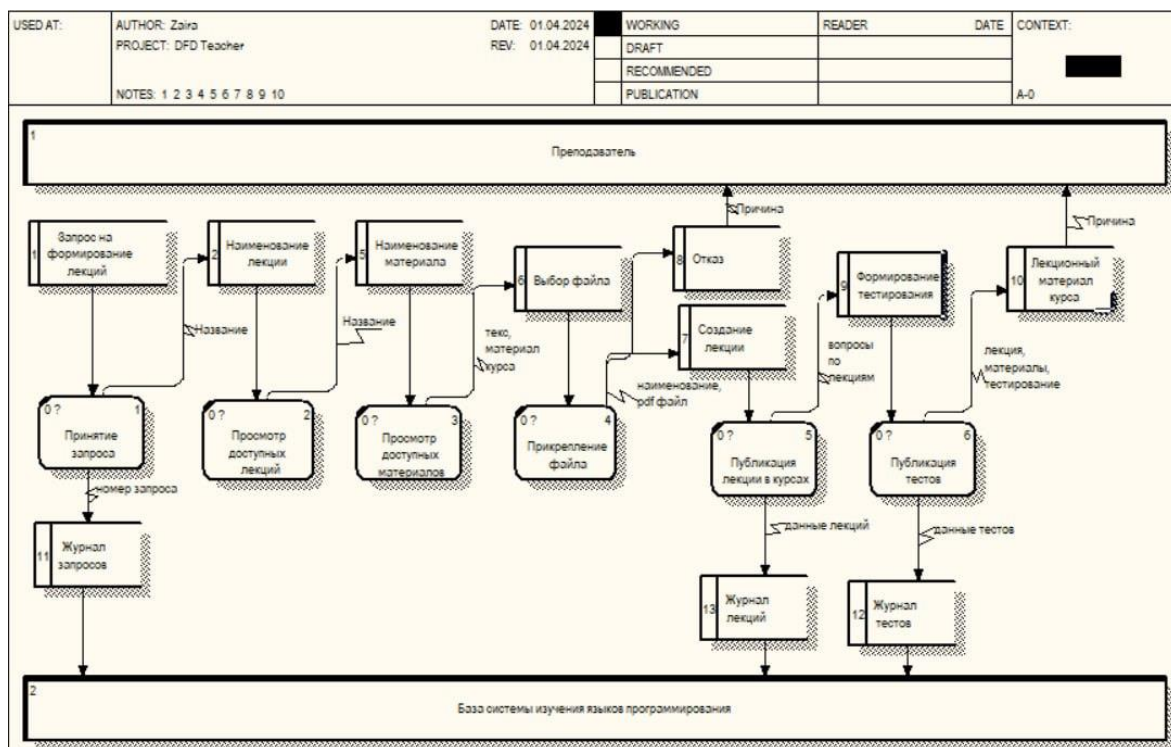


Рисунок 4 DFD студента



2.2.2. Входные и выходные данные

В таблицах под номерами 3-4 представлены входные и выходные данные приложения.

Таблица 3 - Входные данные приложения

Поле	Ограничение	Тип	Форма ввода	Описание
1	2	3	4	5
Авторизация				
Логин пользователя	{5,255}	Строка	Поле ввода	Поле ввода для логина пользователя при авторизации
Пароль пользователя	{6,255}	Строка	Поле ввода	Поле ввода для пароля пользователя при авторизации
Регистрация				
Имя	{5,255}, Только буквы латинского алфавита	Строка	Поле ввода	Поле ввода для имени пользователя при регистрации
Фамилия	{5,255}, Только буквы латинского алфавита	Строка	Поле ввода	Поле ввода для фамилии пользователя при регистрации
Логин пользователя	{5,255}	Строка	Поле ввода	Поле ввода для логина пользователя при авторизации
Пароль	{6,255}, Только буквы латинского алфавита	Строка	Поле ввода	Поле ввода для пароля пользователя при регистрации
Повторный пароль	{6,255}, Только буквы латинского алфавита	Строка	Поле ввода	Поле ввода для повторного пароля пользователя при регистрации
Загрузка, обновление, редактирование фотографий				
Фотография	Ограничения не предоставляются	Строка	Поле ввода	Поле ввода фотографии

Таблица 4 - Выходные данные приложения

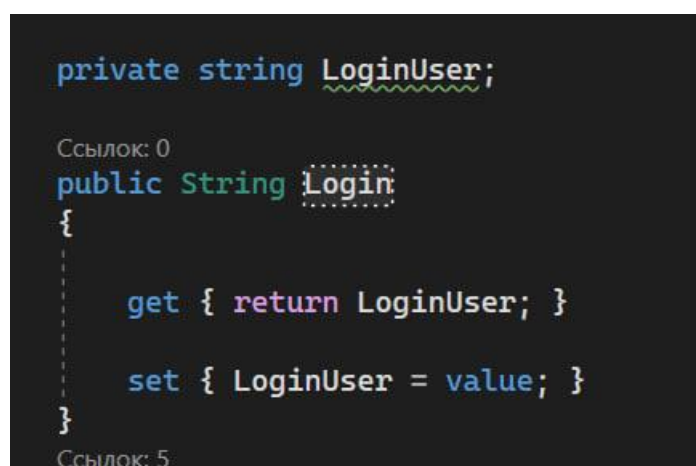
Поле	Ограничение	Тип	Форма ввода	Описание
1	2	3	4	5
Авторизация				
Логин пользователя	{5,255}	Строка	Поле ввода	Поле ввода для логина пользователя при авторизации
Пароль пользователя	{6,255}	Строка	Поле ввода	Поле ввода для пароля пользователя при авторизации
Регистрация				
Имя	{5,255}, Только буквы латинского алфавита	Строка	Поле ввода	Поле ввода для имени пользователя при регистрации
Фамилия	{5,255}, Только буквы латинского алфавита	Строка	Поле ввода	Поле ввода для фамилии пользователя при регистрации
Логин пользователя	{5,255}	Строка	Поле ввода	Поле ввода для логина пользователя при авторизации
Пароль	{6,255}, Только буквы латинского алфавита	Строка	Поле ввода	Поле ввода для пароля пользователя при регистрации
Повторный пароль	{6,255}, Только буквы латинского алфавита	Строка	Поле ввода	Поле ввода для повторного пароль пользователя при регистрации
Загрузка, обновление, редактирование фотографий				
Фотография	Ограничения не предоставляются	Строка	Поле ввода	Поле ввода фотографии

2.2.3. Методы

При разработке приложений использовались среды разработки IntelliJ IDEA и Visual Studio. Visual studio использовалось для написания API при использовании языка C#. IntelliJ IDEA использовалось для написания мобильного приложения с использованием языка программирования Dart.

Данные приложения были написаны с использованием основных парадигм объектно-ориентированного программирования, таких как, инкапсуляция (будет использоваться для ограничения доступа одних компонентов программы к другим); наследование (необходимо для повторного использования кода и способствует независимому расширению программного обеспечения и интерфейсов); полиморфизм (для обработки различных типов данных).

Ниже на рисунке 7 приведён пример инкапсуляции.



```
private string LoginUser;  
  
Ссылка: 0  
public String Login  
{  
    get { return LoginUser; }  
    set { LoginUser = value; }  
}  
Ссылка: 5
```

Рисунок 7 - Пример инкапсуляции

Следующий рисунок 8 представляет наследование.


```

namespace ApiPortfolio.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    Ссылка: 1
    public class UsersController : ControllerBase
    {
        private readonly PortfolioContext _context;

        Ссылка: 0
        public UsersController(PortfolioContext context)
        {

```

Рисунок 8 - Пример наследования

В создании API использовался паттерн MVC (Model-view-controller). Из представления (View) пользователь осуществляет вызов методов контроллера. Из контроллера осуществляется обновление модели в соответствии с действиями пользователя. Затем данная модель передаётся в представление, в свою очередь представление может осуществлять запрос данных из модели.

На рисунке 9 представлена модель.

```

namespace ApiLearningLanguages.Models
{
    public partial class Question
    {
        public int QuestionId { get; set; }

        public int TestId { get; set; }

        public string QuestionText { get; set; } = null!;

        public virtual ICollection<Option> Options { get; set; } = new HashSet<Option>();

        public virtual Test? Test { get; set; } = null!;
    }
}

```

Рисунок 9 - Пример модели из парадигмы MVC

На рисунке 10 представлен контроллер.


```

namespace ApiLearningLanguages.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    Ссылка 1
    public class UsersController : ControllerBase
    {
        private readonly Learning_languagesContext _context;
        private readonly JwtTokenService _jwtTokenService;

        Ссылка 0
        public UsersController(Learning_languagesContext context, JwtTokenService jwtTokenService)
        {
            _context = context;
            _jwtTokenService = jwtTokenService;
        }

        [HttpPost("login")]
        Ссылка 0
        public async Task<ActionResult<object>> Login([FromBody] LoginModel loginModel)
        {
            var user = await _context.Users
                .FirstOrDefaultAsync(u => u.Email == loginModel.Login && u.Password == loginModel.Password);

            if (user == null)
            {
                return Unauthorized("Неверные учетные данные");
            }

            var token = _jwtTokenService.GenerateToken(user);
            // Возвращаем токен и ID пользователя
            return Ok(new { token, userId = user.UserId });
        }

        [HttpPost("register")]
    }
}

```

Рисунок 10 - Пример контроллера из парадигмы MVC

2.2.4. Тесты

Для тестирования программы будет использоваться ручное тестирование по тестовым сценариям, также будет применено нагрузочное тестирование на разработанное API.

Сценарий и результаты тестовых испытаний представлены в Приложении В.

2.2.5. Контроль целостности данных

В ходе разработки информационной системы были применены методы валидации входных данных, которые обеспечивают целостность этих данных. Этот подход включает контроль целостности данных, который определяет ситуации и действия приложения при выполнении задач, связанных с сохранением, изменением или удалением данных.

Таблица 5 - Контроль целостности данных

№	Ситуация	Аномалия	Реакция программы
1	2	3	4
1	Регистрация пользователя	Введена существующая почта	Программа после попытки регистрации сообщает о провале регистрации.
		Введен некорректный пароль	Программа после попытки регистрации сообщает о провале регистрации.
2	Добавление, изменение курса	Введено пустое название при изменении	Программа после попытки изменения данных сообщает о провале.

2.3. Проектирование

2.3.1. Схема архитектуры приложения

Приложение отправляет запросы на сервер, где находится база данных, с помощью API. Запросы и ответы осуществляются по определенному протоколу и формату данных, которые заранее согласовываются между приложением и сервером.

Приложение может отправлять запросы на чтение, изменение или удаление данных из базы данных через API. Соответственно, сервер обрабатывает запросы и отправляет обратно ответы с нужной информацией в формате, понятном приложению. Это позволяет приложению получать и обрабатывать данные из базы данных без необходимости обращения к ней напрямую.

Ниже на рисунке 13 представлена схема связи базы данных и приложения через API.

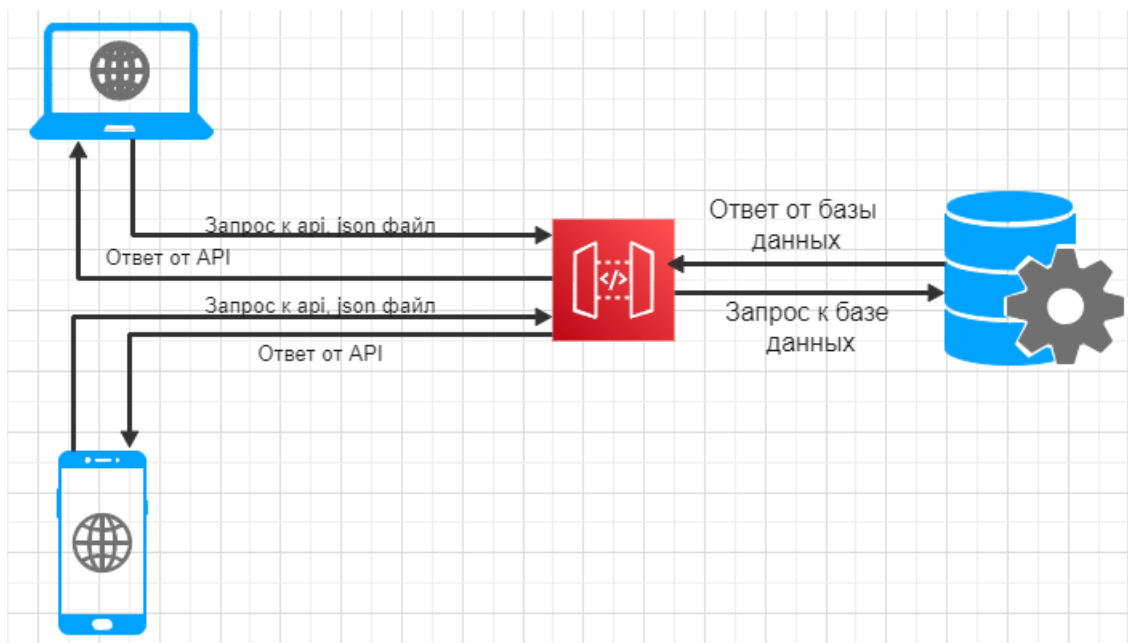


Рисунок 11 - Схема архитектуры приложения

2.3.2. Логическая схема данных

Ниже на рисунке 14 представлена логическая схема данных, разработанная под приложение.

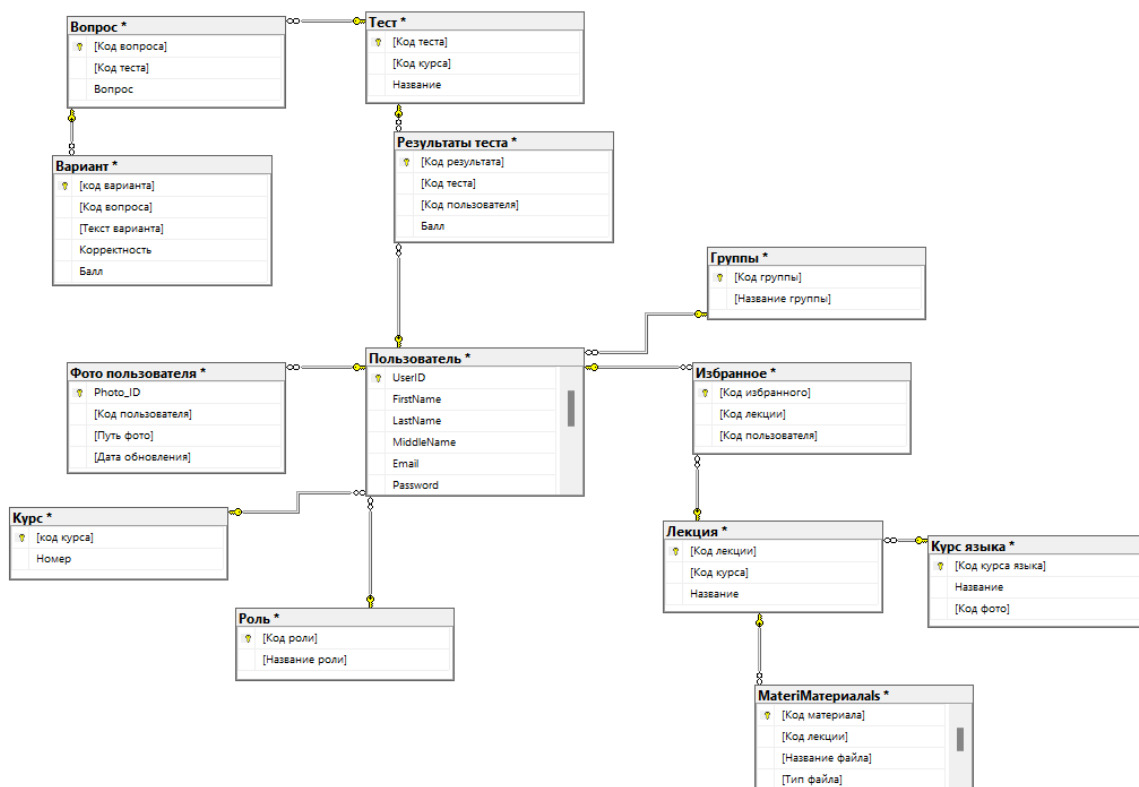


Рисунок 12 - Логическая схема базы данных

2.3.3. Физическая схема данных

В физической схеме данных показано какие типы данных имеют поля в таблицах. Также на ней стрелочками показано какие таблицы имеют связь между собой.

Ниже на рисунке 15 представлена физическая схема базы данных

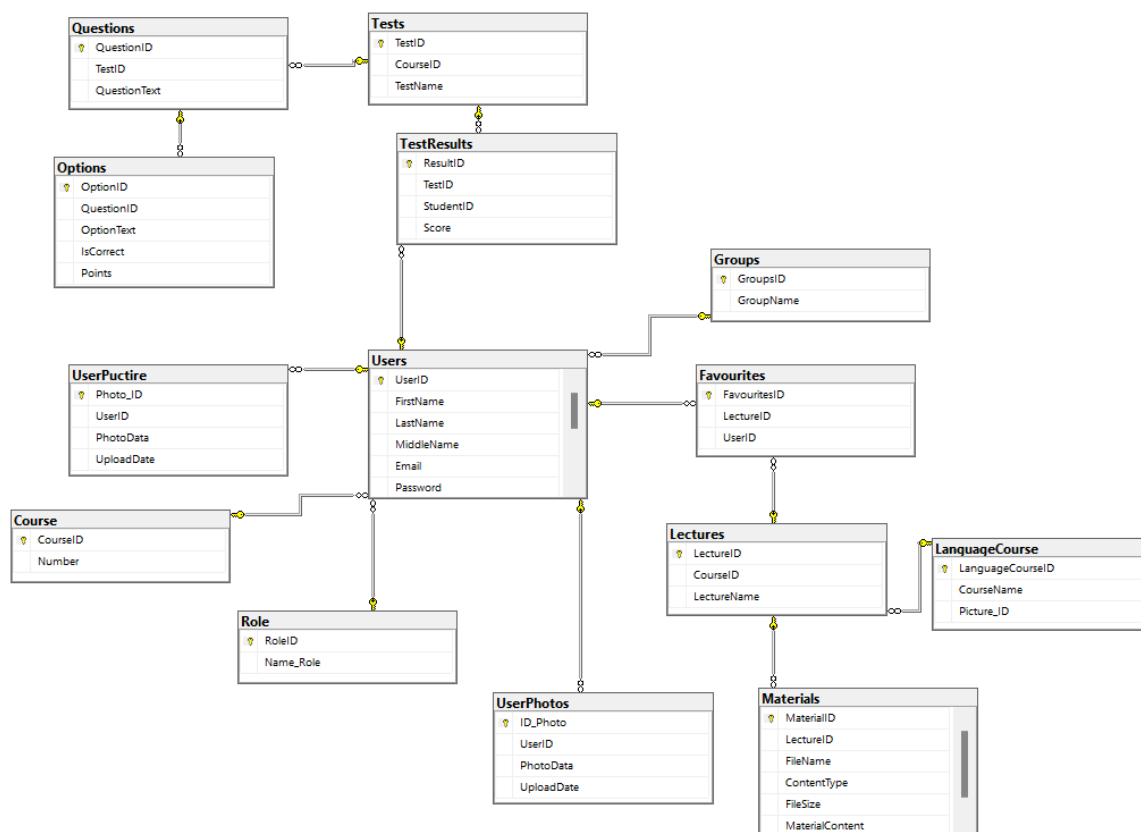


Рисунок 13 - Физическая схема базы данных

Словарь данных представлен в таблице ниже.

Таблица 6 - Словарь данных

Ключ	Наименование	Тип данных	Обязательность заполнения	Описание
GroupsID	Идентификатор группы	INT	Да	Уникальный идентификатор группы
GroupName	Название группы	VARCHAR(200)	Да	Название учебной группы
CourseID	Идентификатор курса	INT	Да	Уникальный идентификатор курса
Number	Номер	INT	Да	Номер курса

ID_Picture	Идентификатор изображения	INT	Да	Уникальный идентификатор изображения
PhotoData	Бинарные данные изображения	VARBINARY(max)	Да	Двоичные данные изображения
UploadDate	Дата загрузки изображения	DATETIME	Да	Дата и время загрузки изображения
RoleID	Идентификатор роли	INT	Да	Уникальный идентификатор роли
Name_Role	Наименование роли	VARCHAR(50)	Да	Название роли пользователя
UserID	Идентификатор пользователя	INT	Да	Уникальный идентификатор пользователя
FirstName	Имя	VARCHAR(100)	Да	Имя пользователя
LastName	Фамилия	VARCHAR(100)	Да	Фамилия пользователя
MiddleName	Отчество	VARCHAR(100)	Нет	Отчество пользователя
Email	Электронная почта	VARCHAR(255)	Да	Электронная почта пользователя
Password	Пароль	VARCHAR(255)	Да	Пароль пользователя
GroupsID	Идентификатор группы	INT	Нет	Ссылка на группу пользователя
CourseID	Идентификатор курса	INT	Нет	Ссылка на курс пользователя
RoleID	Идентификатор роли	INT	Нет	Ссылка на роль пользователя
Salt	Соль	VARCHAR(256)	Нет	Дополнительный уровень защиты пароля
FcmToken	Токен FCM	NVARCHAR(255)	Нет	Firebase Cloud Messaging токен
LanguageCourseID	Идентификатор курса языка	INT	Да	Уникальный идентификатор курса языка

CourseName	Название курса	VARCHAR(200)	Да	Название курса языка
Picture_ID	Идентификатор изображения	INT	Нет	Ссылка на изображение курса
LectureID	Идентификатор лекции	INT	Да	Уникальный идентификатор лекции
CourseID	Идентификатор курса	INT	Да	Ссылка на курс языка
LectureName	Название лекции	VARCHAR(200)	Да	Название лекции
FavouritesID	Идентификатор избранного	INT	Да	Уникальный идентификатор записи в избранном
LectureID	Идентификатор лекции	INT	Да	Ссылка на лекцию, добавленную в избранное
UserID	Идентификатор пользователя	INT	Да	Ссылка на пользователя, добавившего лекцию в избранное
MaterialID	Идентификатор материала	INT	Да	Уникальный идентификатор материала
LectureID	Идентификатор лекции	INT	Да	Ссылка на лекцию
FileName	Имя файла	NVARCHAR(255)	Да	Имя файла с материалом
ContentType	Тип содержимого	NVARCHAR(100)	Да	MIME-тип содержимого файла
FileSize	Размер файла	BIGINT	Да	Размер файла в байтах
MaterialContent	Содержимое материала	VARBINARY(MAX)	Да	Двоичные данные материала
TestID	Идентификатор теста	INT	Да	Уникальный идентификатор теста
CourseID	Идентификатор курса	INT	Да	Ссылка на курс, к которому принадлежит тест
TestName	Название теста	VARCHAR(100)	Да	Название теста

QuestionID	Идентификатор вопроса	INT	Да	Уникальный идентификатор вопроса
TestID	Идентификатор теста	INT	Да	Ссылка на тест
QuestionText	Текст вопроса	VARCHAR(MAX)	Да	Текст вопроса
OptionID	Идентификатор опции	INT	Да	Уникальный идентификатор опции
QuestionID	Идентификатор вопроса	INT	Да	Ссылка на вопрос
OptionText	Текст опции	VARCHAR(MAX)	Да	Текст опции
IsCorrect	Правильность ответа	BIT	Да	Является ли опция правильным ответом
Points	Количество баллов	INT	Да	Количество баллов за выбор этой опции
ResultID	Идентификатор результата	INT	Да	Уникальный идентификатор результата теста
TestID	Идентификатор теста	INT	Да	Ссылка на тест
StudentID	Идентификатор студента	INT	Да	Ссылка на пользователя, проходящего тест
Score	Количество баллов	INT	Да	Количество баллов, полученных студентом
Photo_ID	Идентификатор фотографии	INT	Да	Уникальный идентификатор фотографии пользователя
UserID	Идентификатор пользователя	INT	Да	Ссылка на пользователя
PhotoData	Бинарные данные изображения	VARBINARY(MAX)	Да	Двоичные данные фотографии
UploadDate	Дата загрузки изображения	DATETIME	Да	Дата и время загрузки изображения

2.3.4. Структурная схема

В данной структурной схеме, также известной как диаграмма классов, представлена графическая модель, показывающая связи, зависимости и отношения между классами в программном проекте. Классы, как объекты программирования, могут иметь свойства, методы и поля. Диаграмма классов обычно используется для представления сложных систем и облегчения понимания связей между классами, что упрощает разработку и сопровождение кода в будущем.

Диаграмма классов приложения:

На рисунке 16 представлена схема приложения для персонала.

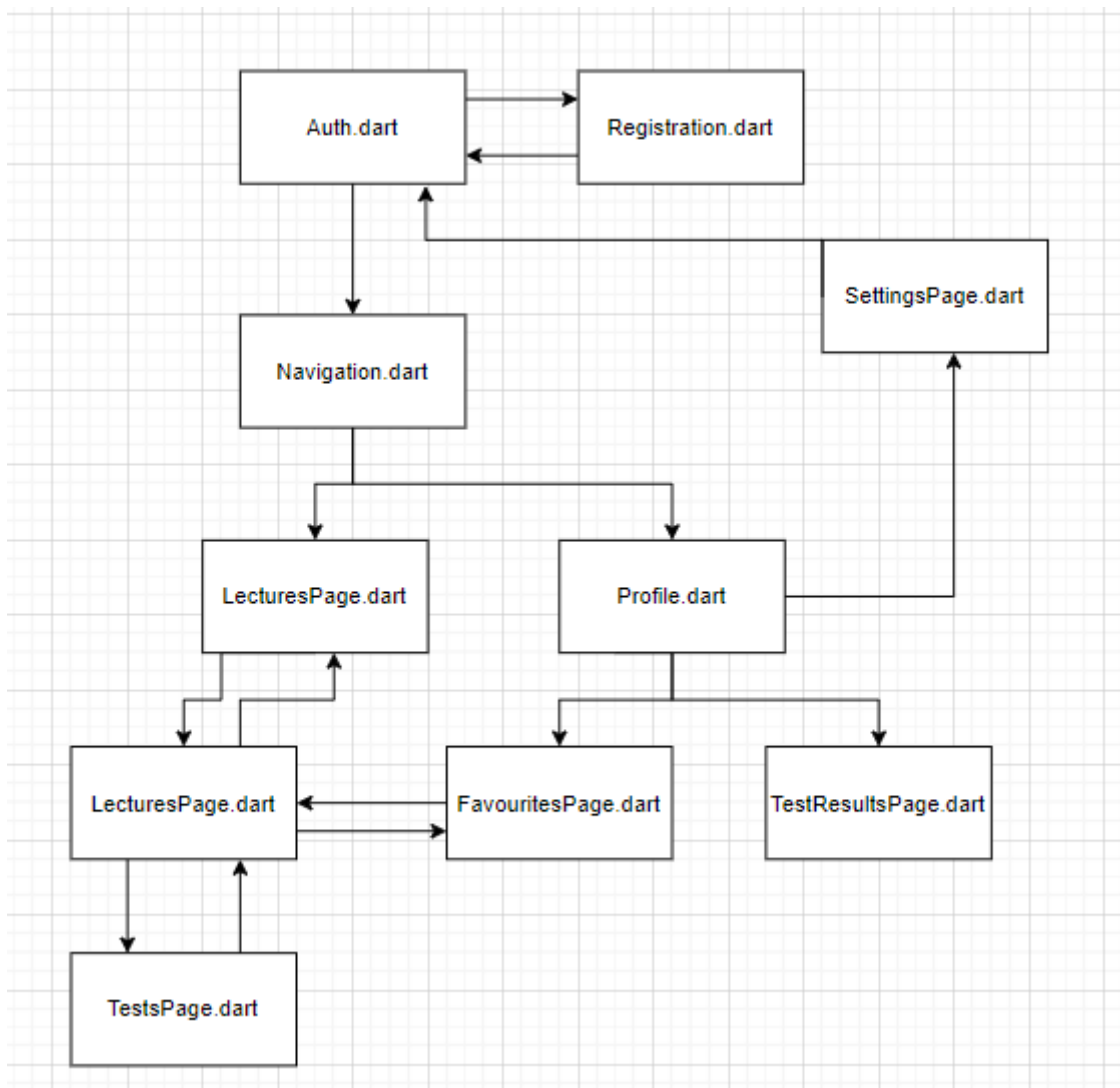


Рисунок 14 - Структурная схема приложения персонала.

2.3.5. Функциональная схема

На функциональной схеме представлены функции, находящиеся на окнах приложений. Данная схема позволяет легче проследить связи между компонентами приложения и оценивать работу системы в целом.

На рисунке 17 представлена функциональная схема приложение для персонала.

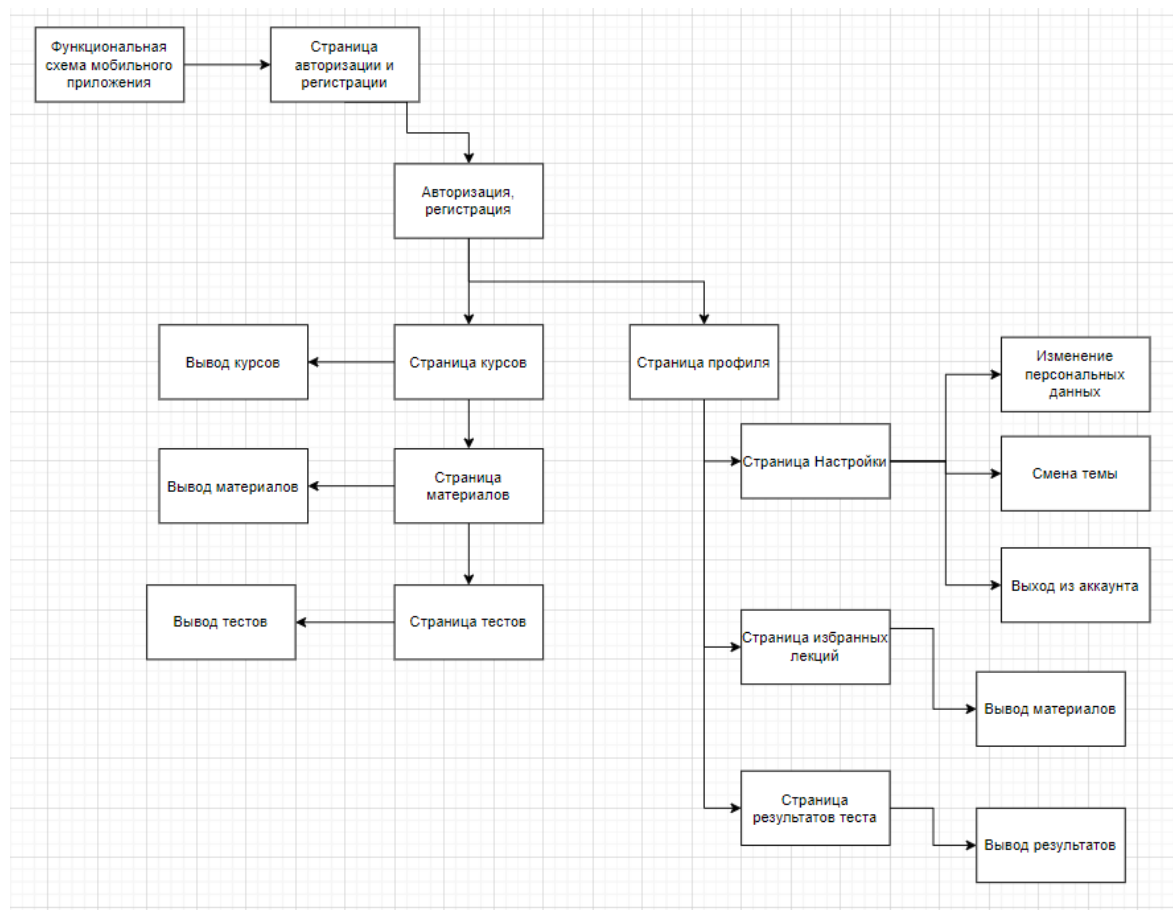


Рисунок 15 - Функциональная схема приложения.

2.3.6. Схема тестирования

Схема тестирования показана в приложении В Сценарий и результаты тестовых испытаний.

2.3.7. Схема пользовательского интерфейса

Ниже представлена схема пользовательского интерфейса, которая показывает, как осуществляется переход между окнами приложений.

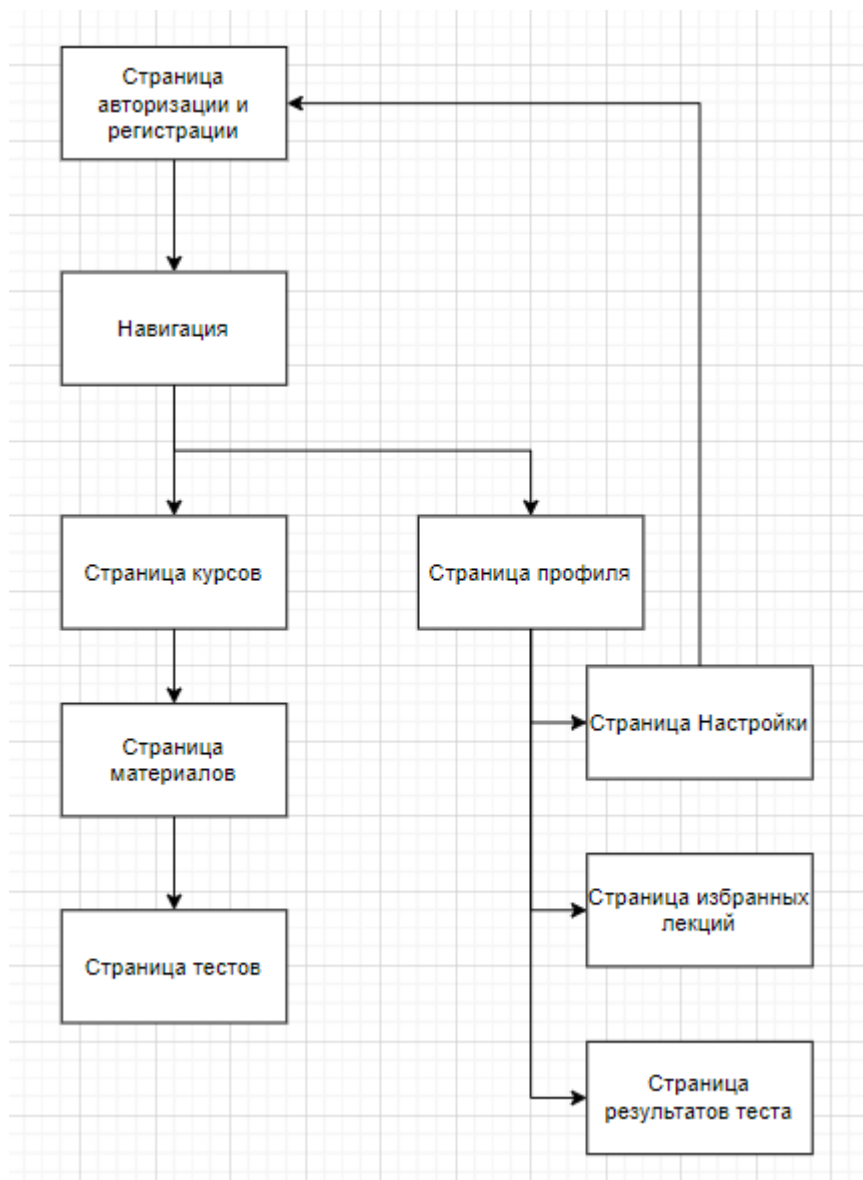


Рисунок 16 - Схема пользовательского интерфейса

2.4. Результат работы программы

В результате выполнения поставленной задачи, было разработано решение, которое является мобильным приложением

На рисунках 19-20 представлен результат работы программы. Больше информации о результате работы программы описан в приложении Г «Руководство пользователя».

Ниже представлены фотографии приложения для пользователя:

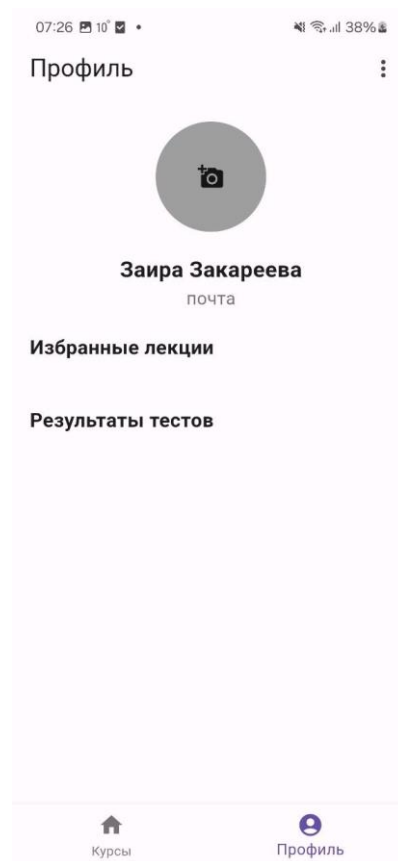


Рисунок 17 – Профиль



Рисунок 18 – Страница курсов

3. ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ

3.1. Инструментальные средства

MsSQL была выбрана в качестве основы для базы данных из-за её удобного ядра и использования SQL Server Management Studio для разработки. Postgres, хоть и бесплатная, и открытая, обладает богатым функционалом, но настройка и управление могут быть сложными. MySQL, также бесплатная и популярная, уступает MSSQL и Postgres в определенных возможностях, таких как поддержка транзакций, полнотекстовый поиск и работа с большими объемами данных. Именно поэтому MS SQL Server была предпочтительнее при выборе ядра для этой базы данных.

Были выбраны инструменты разработки, такие как IntelliJ IDEA для написания мобильного приложения, а также Microsoft Visual Studio 2022 для создания API и WPF приложения. IntelliJ IDEA была выбрана в качестве оптимальной среды разработки приложений под Android на языке Dart. Microsoft Visual Studio 2022, с фреймворком ASP.NET и архитектурой MVC, использовалась для удобства разработки API. Архитектура MVC предоставляет значительные преимущества, включая возможность генерировать стандартные запросы к API. Кроме того, Visual Studio выбрана благодаря возможности создавать приложения для операционной системы Windows.

3.2. Отладка программы

Отладка приложения и API была осуществлена с помощью встроенных средств отладки в IntelliJ IDEA и в IDE Visual Studio. Отладка использовалась при включенном приложении и приостанавливалась, доходя до точки остановки.

В ходе работы над проектом было распространено несколько ошибок. Первая ошибка была проблемой с пользователями, происходила

неверная инициализация, из-за чего пользователь был пустым и его персональные данные тоже.



Рисунок 19 - Ошибка инициализации

Второй распространенной проблемой было не правильное обращение к методу, к методу, который вызывал ошибку «404». Ошибка исправляется верным заполнением данных и перезапуском запроса.

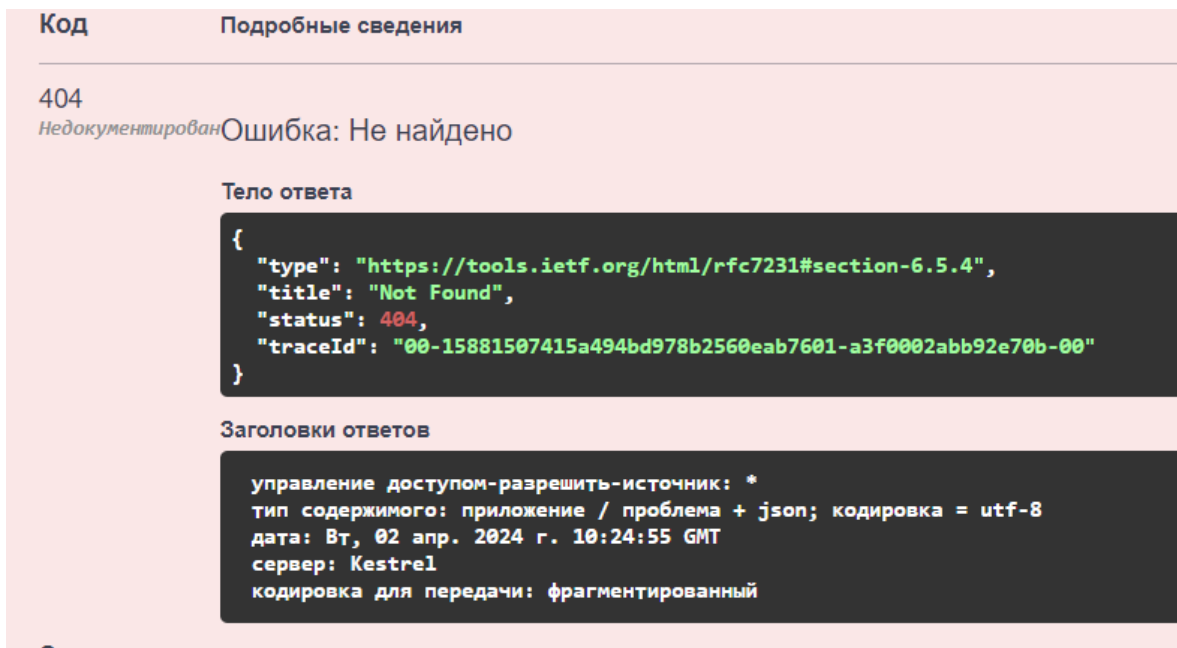


Рисунок 20 - Ошибка не правильного обращения к методу

3.3. Защитное программирование

Механизм `try catch` в программе и в API помогает обнаруживать ошибки во время выполнения кода и предпринимать соответствующие действия для их обработки, что повышает надежность и безопасность системы.

В случае возникновения ошибок, механизм `try_catch` позволяет изолировать проблемный участок кода и обработать ошибку локально, не прерывая выполнение всего скрипта или приложения, что улучшает производительность и снижает риски сбоев.

Ниже представлен пример его использования:

```
Future<void> _deleteLecture(int lectureId) async {
  try {
    final response = await Dio().delete('$api/Lectures/$lectureId');

    if (response.statusCode == 204) {
      ScaffoldMessenger.of(context).showSnackBar(
        const SnackBar(content: Text('Лекция успешно удалена!')),
      );
      fetchLectures().then((updatedLectures) {
        setState(() {
          // Обновление списка лекций
        });
      });
    } else {
      throw Exception('Ошибка при удалении лекции');
    }
  } catch (e) {
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text('Ошибка при удалении лекции: $e')),
    );
  }
}
```

Рисунок 21 - Блок try catch

Так же для безопасности был использован токен, который генерируется при авторизации пользователя. В токене шифруется номер учетной записи пользователя и его роль. После чего у каждого запроса в API есть проверка на то какие роли имеют к ней доступ.

```

public class JwtTokenService
{
    private readonly IConfiguration _configuration;

    Ссылка 0
    public JwtTokenService(IConfiguration configuration)
    {
        _configuration = configuration;
    }

    Ссылка 2
    public string GenerateToken(User user)
    {
        var securityKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(_configuration["Jwt:Key"]));
        var credentials = new SigningCredentials(securityKey, SecurityAlgorithms.HmacSha256);

        var claims = new[]
        {
            new Claim(JwtRegisteredClaimNames.Sub, user.Email),
            new Claim("id", user.UserId.ToString()),
            new Claim(JwtRegisteredClaimNames.Jti, Guid.NewGuid().ToString()),
            // Добавьте другие claims здесь, если необходимо
        };

        var token = new JwtSecurityToken(
            issuer: _configuration["Jwt:Issuer"],
            audience: _configuration["Jwt:Audience"],
            claims: claims,
            expires: DateTime.Now.AddMinutes(120),
            signingCredentials: credentials
        );

        return new JwtSecurityTokenHandler().WriteToken(token);
    }
}

```

Рисунок 22 - Методы для работы с токеном.

3.4. Характеристики программы

Разработанное мобильное приложение может запускаться на операционной системе Android и IOS. Основным критерием считается версия ОС Android, на котором запускается разработанный ресурс, версия должна быть 8.0 или выше.

Приложения администратора и персонала могут быть запущены на операционных системах Windows 10 и выше.

Характеристики программы представлены в приложении Б «Текст программы» в таблице 1 «Модули».

ЗАКЛЮЧЕНИЕ

Заключение проекта разработки мобильного приложения для изучения языков программирования подчеркивает его значимость и актуальность в современном мире. Это приложение не только открывает перед пользователями широкие возможности для изучения и практического применения знаний в области программирования, но и предоставляет инструменты для постоянного развития и самосовершенствования. Интеграция системы тестирования и создание дружелюбного пользовательского интерфейса делают процесс обучения не только эффективным, но и мотивирующим.

Осознавая важность постоянного обучения и адаптации к новым технологиям, приложение предлагает решение, которое помогает пользователям не только следовать за текущими трендами в мире программирования, но и активно участвовать в этом процессе, формируя собственные знания и навыки. Разработка такого приложения является вкладом в будущее информационных технологий, где каждый желающий сможет найти для себя путь в мире программирования, независимо от предыдущего опыта и уровня подготовки.

В заключение, данный проект несет в себе большой потенциал для образовательной сферы, предоставляя удобный, интуитивно понятный и всесторонний инструмент для изучения языков программирования. Это приложение способно вдохновить новое поколение программистов, исследователей и разработчиков, обеспечивая им основу для достижения новых высот в развитии технологий и воплощения инновационных идей в жизнь.

СПИСОК ИСПОЛЬЗУЕМЫХ МАТЕРИАЛОВ

1. Microsoft Docs - SQL Server Documentation: <https://docs.microsoft.com/en-us/sql/> (Дата использования: 01.10.23)
2. Flutter Documentation: <https://flutter.dev/docs> (Дата использования: 02.10.23)
3. Dart Documentation: <https://dart.dev/guides> (Дата использования: 03.10.23)
4. ASP.NET Documentation: <https://docs.microsoft.com/en-us/aspnet/> (Дата использования: 04.10.23)
5. Microsoft SQL Server Official Website: <https://www.microsoft.com/en-us/sql-server/> (Дата использования: 05.10.23)
6. C# Programming Guide: <https://docs.microsoft.com/en-us/dotnet/csharp/> (Дата использования: 06.10.23)
7. Flutter and Dart Package Repository - pub.dev: <https://pub.dev/> (Дата использования: 07.10.23)
8. ASP.NET Core Documentation: <https://docs.microsoft.com/en-us/aspnet/core/> (Дата использования: 08.10.23)
9. SQL Server Central - Community and Resources: <https://www.sqlservercentral.com/> (Дата использования: 09.10.23)
10. Flutter YouTube Channel: <https://www.youtube.com/c/FlutterDev> (Дата использования: 10.10.23)
11. Dart GitHub Repository: <https://github.com/dart-lang> (Дата использования: 11.10.23)
12. C# GitHub Repository: <https://github.com/dotnet/csharplang> (Дата использования: 12.10.23)
13. SQL Server Blog - Official Microsoft Blog for SQL Server: <https://techcommunity.microsoft.com/t5/sql-server-blog/bg-p/SQLServer> (Дата использования: 13.10.23)
14. Flutter Cookbook: <https://flutter.dev/docs/cookbook> (Дата использования: 14.10.23)
15. .NET Blog - Official Blog for the .NET development community: <https://devblogs.microsoft.com/dotnet/> (Дата использования: 15.10.23)
16. Flutter GitHub Repository: <https://github.com/flutter/flutter> (Дата использования: 16.10.23)
17. DartPad - Online Dart Editor: <https://dartpad.dev/> (Дата использования: 17.10.23)
18. SQL Server Books Online: <https://technet.microsoft.com/en-us/sqlserver/bb671430.aspx> (Дата использования: 18.10.23)
19. ASP.NET Community Forums: <https://forums.asp.net/> (Дата использования: 19.10.23)
20. C# Corner - Online Community for Software Developers: <https://www.c-sharpcorner.com/> (Дата использования: 20.10.23)

21. FlutterFire - Firebase Integration for Flutter: <https://firebase.flutter.dev/> (Дата использования: 21.10.23)
22. .NET Foundation - Supporting Open Source .NET Projects: <https://dotnetfoundation.org/> (Дата использования: 22.10.23)
23. Dart SDK GitHub Repository: <https://github.com/dart-lang/sdk> (Дата использования: 23.10.23)
24. SQL Server Performance Tuning Tips: <https://www.mssqltips.com/> (Дата использования: 24.10.23)
25. Xamarin - Cross-platform App Development with .NET: <https://dotnet.microsoft.com/apps/xamarin> (Дата использования: 25.10.23)
26. Entity Framework Documentation: <https://docs.microsoft.com/en-us/ef/> (Дата использования: 26.10.23)
27. Flutter Medium Publication: <https://medium.com/flutter> (Дата использования: 27.10.23)
28. SQL Server Query Optimization Techniques: <https://www.red-gate.com/simple-talk/sql/performance/sql-query-optimization-techniques-in-sql-server/> (Дата использования: 28.10.23)
29. Blazor Documentation - Building Interactive Web Applications with .NET: <https://docs.microsoft.com/en-us/aspnet/core/blazor/> (Дата использования: 29.10.23)
30. Stack Overflow - Community-driven Question and Answer Platform: <https://stackoverflow.com/> (Дата использования: 30.10.23)

Проверка на уникальность авторского текста курсового проекта

На тему: Разработка мобильного приложения для изучения языков программирования

Закареевой Заире Руслановне
Студент (-ка) 4 курса группы П50-11-21

по специальности 09.02.07 «Информационные системы и программирование»
квалификация: Программист

Уникальность текста проверялась на сайте www.antiplagiat.ru. Процент оригинальности содержимого пояснительной записки составил 96,51%. Результат представлен на Рисунке 1.

Отчет о проверке

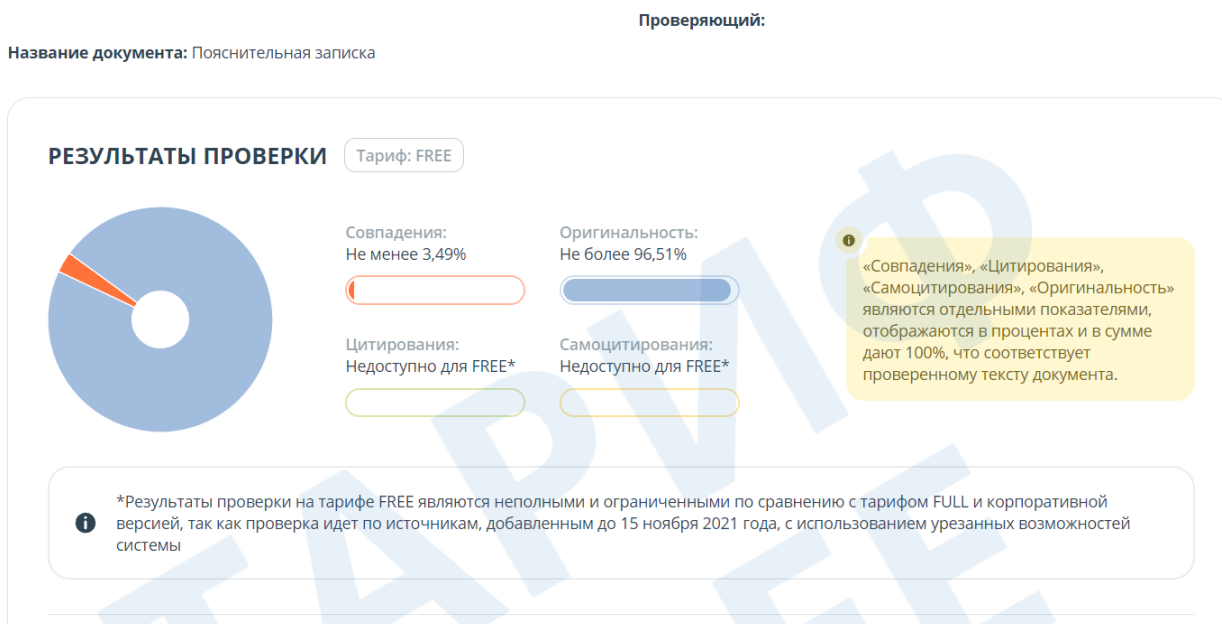


Рисунок 1 - Проверка на уникальность текста пояснительной записки