СОДЕРЖАНИЕ

[Введение 4](#_Toc191006787)

[1. Общая часть 6](#_Toc191006788)

[1.1. Цель разработки 6](#_Toc191006789)

[1.2. Средства разработки 6](#_Toc191006790)

[2. Специальная часть 9](#_Toc191006791)

[2.1. Постановка задачи 9](#_Toc191006792)

[2.1.1. Подробные требования к проекту 9](#_Toc191006793)

[2.2. Внешняя спецификация 10](#_Toc191006794)

[2.2.1. Описание задачи 10](#_Toc191006795)

[2.2.2. Входные и выходные данные 11](#_Toc191006796)

[2.2.3. Методы 11](#_Toc191006797)

[2.2.4. Технологии программирования 13](#_Toc191006798)

[2.3. Проектирование 13](#_Toc191006799)

[2.3.1. Схема данных 13](#_Toc191006800)

[2.3.2. Словарь данных 14](#_Toc191006801)

[2.3.3. Схема архитектуры 17](#_Toc191006802)

[2.3.4. Диаграмма потоков данных 18](#_Toc191006803)

[2.3.5. Сценарий использования 22](#_Toc191006804)

[2.4. Результат работы программы 22](#_Toc191006805)

[3. Технологическая часть 23](#_Toc191006806)

[3.1. Инструментальная среда разработки 23](#_Toc191006807)

[Заключение 25](#_Toc191006808)

[Список используемых материалов 26](#_Toc191006809)

[Приложение А. Текст программы 1](#_Toc191006810)

[Приложение Б. Сценарий и результаты тестовых испытаний 1](#_Toc191006811)

[Приложение В. Руководство пользователя 1](#_Toc191006812)

[Приложение Г. Скрипт базы данных 1](#_Toc191006813)

# Введение

В связи с современной тенденцией и развитием современных технологий, которые уже используются, практические, во всех сферах жизнедеятельности человека, то один из самых важных аспектов развития человека не мог пройти мимо данную тенденцию стороной. Система образования всегда развивалась в ногу со временем, предоставляя все больше возможностей для получения знаний и их закрепления, поэтому появление систем онлайн-образований было только делом времени. Данный формат обучения может предоставить студентам или же обучающимся возможность обучаться в том формате и темпе, который будет удобен для них, изучать те темы, которые интересны и могут быть полезны по их мнению. Система онлайн-образования также помогает и преподавателям, что дает возможность вести прямой контакт со студентами на расстоянии, выдавать лекционный материал и проводить оценку знаний. Составление, выдача и проверка тестов отнимает много времени и сил, таким образом и появилась идея по реализации веб-приложения «Система онлайн-образования», которое сможет предоставить студентам возможность обучаться в удобном формате, а преподавателям автоматизировать процесс выдачи и проверки тестов.

Так как подразумевается система онлайн-образования, то предстоит реализовать клиент-серверное веб-приложение с использованием, в качестве передачи данных из базы данных в веб-приложение – API.

Целью курсовой работы является повысить знания в области разработки веб-приложений на фреймворке «Django», а также по реализации API.

Задачей курсовой работы является оптимизация процесса обучения и автоматизация деятельности преподавателя по выдаче тестов и проверке их для оценивания знаний студентов.

Для определения подробностей предметной области, необходимо провести ее анализ с помощью построения бизнес-процессов, выполняющихся без использования информационной системы, определением функциональных возможностей каждых из ролей, по средству по строение диаграммы прецендентов и определением поток данных, пока происходит выполнение бизнес-процесса.

# 1. Общая часть

## 1.1. Цель разработки

Целью разработки веб-приложения «Информационная система контроля освоения знаний и практических умений обучающихся (на примере ФГБОУ ВО РЭУ им. Г.В. Плеханова)» является предоставления возможности удобной организации взаимодействия студентов и преподавателя в процессе обучения. Проект позволит оптимизировать учебный процесс и автоматизировать проведение тестовых срезов по изученным темам.

## 1.2. Средства разработки

Для проектирования, разработки и тестирования веб-приложения «Информационная система контроля освоения знаний и практических умений обучающихся (на примере ФГБОУ ВО РЭУ им. Г.В. Плеханова)» были использованы программные средства, представленные в Таблице 1.

Таблица -Программные средства разработки веб-приложения

| № | Тип средства | Название средства | Назначение |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 1 | Операционная система | Microsoft Windows 11 | Организация взаимодействия программ и пользователя |
| 2 | Инструментальные средства разработки программного обеспечения | Microsoft Visual Studio 2022 17.9.6 | Разработка API |
| 3 | Инструментальные средства разработки программного обеспечения | Microsoft Visual Studio Code | Разработка веб-приложения |
| 4 | СУБД | SQL Server Management Studio 18 | Разработка базы данных |
| 5 | Браузер | Google Chrome ver 111.0.5563.65 | Просмотр API, просмотр веб приложения |

В Таблице 2 представлены минимальные и рекомендованные системные требования, на базе которых возможно комфортное использование реализуемого веб-приложения.

Таблица - Минимальные и рекомендованные системные требования

| № | Тип оборудования | Наименование оборудования |
| --- | --- | --- |
| 1 | 2 | 3 |
| Минимальные системные требования | | |
| Персональный компьютер | | |
| 1 | Операционная система | Microsoft Windows 7, macOS 10.12, Linux Ubuntu 16.04 |
| 2 | Браузер | Google Chrome 80+, Mozilla Firefox 78+, Microsoft Edge 80+, Safari 12+ |
| 3 | Процессор | Процессор с 2 физическими ядрами с тактовой частотой 1,5 ГГц |
| 4 | Оперативная память | 4 Гб |
| 5 | Видеокарта | Интегрированная с поддержкой WebGL |
| 6 | Свободное место на накопителе | 1Гб |
| 7 | Пропускная способность интернет-соединения | 3 Мбит/с |
| 8 | Размер экрана | 14,0 |
| 9 | Разрешение экрана | 1366x768 |
| Рекомендованные системные требования | | |
| Персональный компьютер | | |
| 1 | Операционная система | Windows 10/11, macOS 12+, Linux (Ubuntu 22.04+) |
| 2 | Браузер | Google Chrome (Последней версии), Mozilla Firefox (Последней версии), Samsung Internet (Последней версии), Safari (Последней версии) |
| 3 | Процессор | Процессор с 4 физическими ядрами с тактовой частотой 2,5 ГГц |
| 4 | Оперативная память | 8 Гб+ |
| 5 | Видеокарта | Дискретная или интегрированная, поддержка WebGL 2.0 |
| 6 | Свободное место на накопителе | 1Гб и больше |
| 7 | Пропускная способность интернет-соединения | 25 Мбит/с |
| 8 | Размер экрана | 14,0 |
| 9 | Разрешение экрана | 1920x1080 |

В качестве средств вычислительной техники при разработке и для использования веб-приложения использовался ноутбук Asus Zenbook 14 UM433IQ, в Таблице 3 представлены подробные характеристики.

Таблица - Подробные характеристики средства вычислительной техники

| № | Тип средства | Название средства |
| --- | --- | --- |
| 1 | 2 | 3 |
| Для разработки и использования | | |
| Ноутбук Asus Zenbook 14 UM433IQ | | |
| 1 | Размер экрана | 14 |
| 2 | Разрешение экрана | 1920x1080 |
| 3 | Линейка процессора | AMD Ryzen 7 4700U with Radeon Graphics |
| 4 | Количество ядер процессора | 8 |
| 5 | Оперативная память | 16 Гб |
| 6 | Видеокарта | NVIDIA GeForce MX350 |
| 7 | Конфигурация накопителей | SSD |
| 8 | Общий объем всех накопителей | 1000 |

# 2. Специальная часть

## 2.1. Постановка задачи

На рисунке 1 представлен основной бизнес-процесс «Контролирование освоения знаний и практических умений обучающихся» модели To-Be. Ниже расположены задачи Веб-приложения (основные бизнесс-процессы)

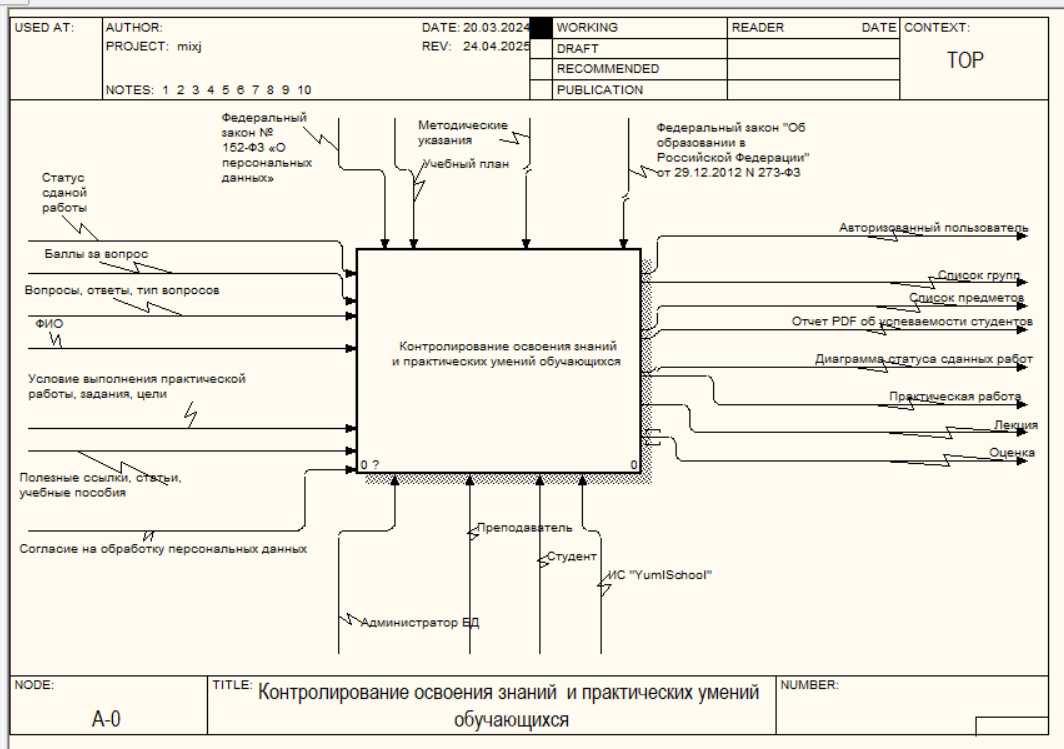


Рисунок - Основной бизнес-процесс (To-Be)

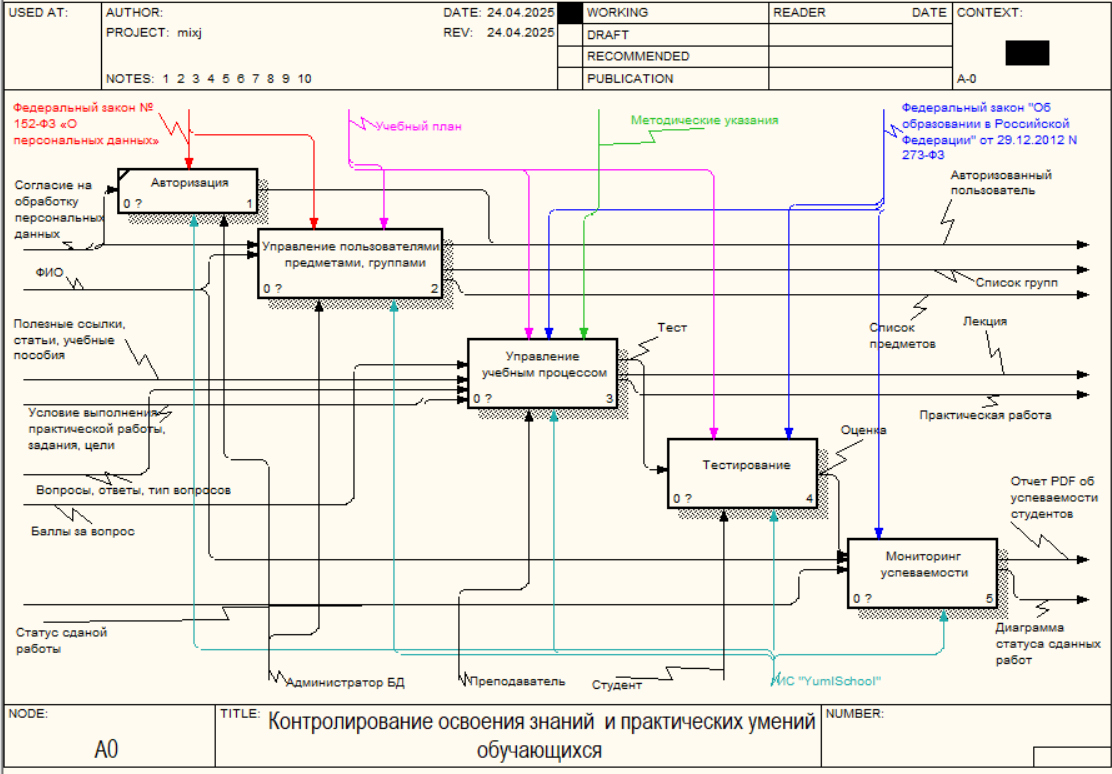


Рисунок - Декомпозиция основного процесса

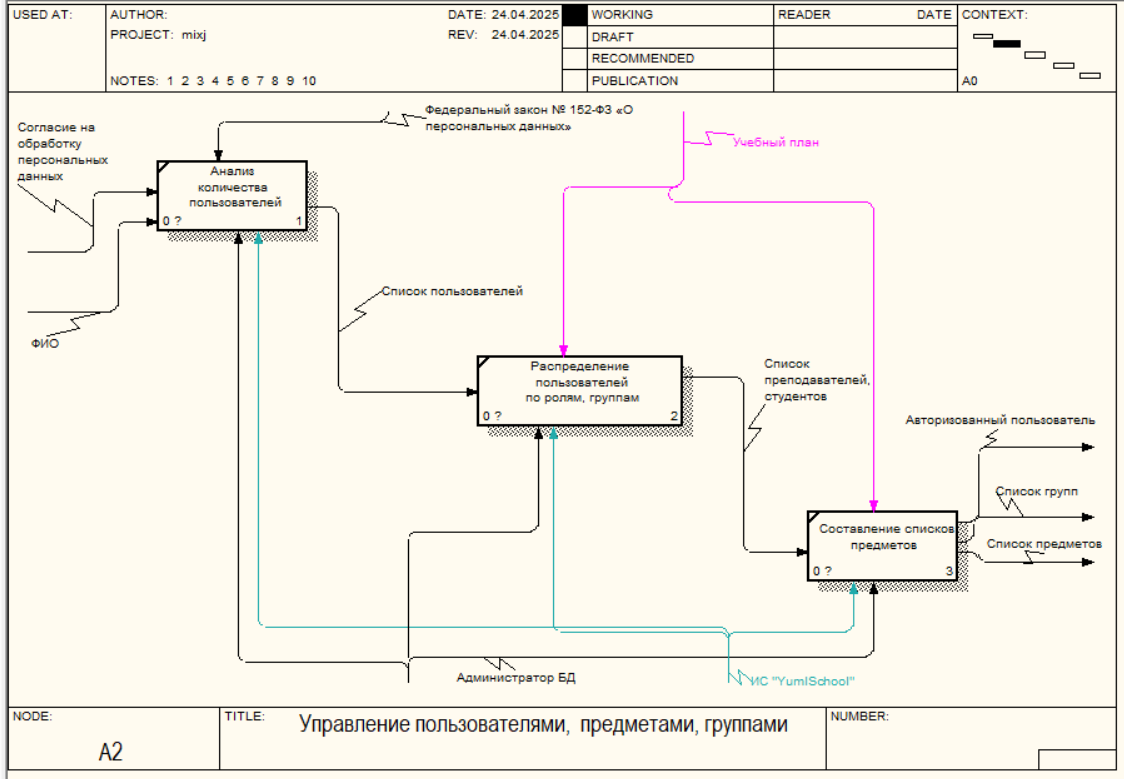


Рисунок - Декомпозиция управления пользователями, предметами, группам

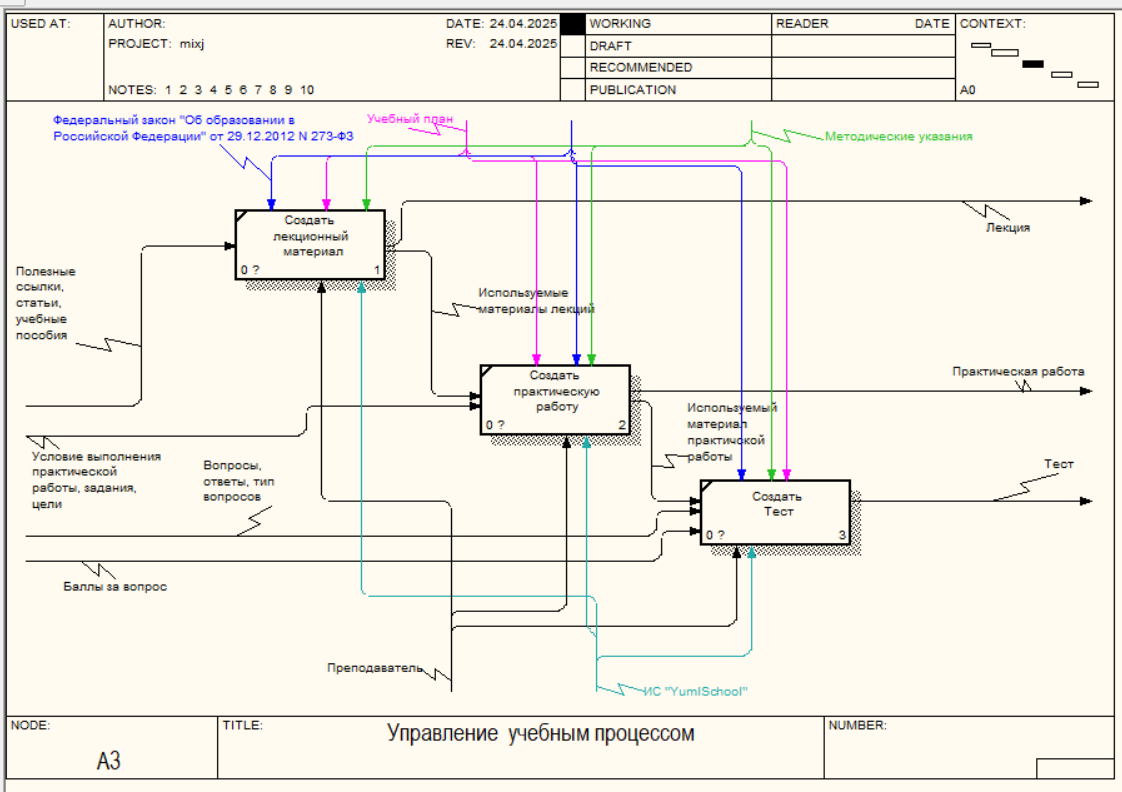


Рисунок - Декомпозиция управления учебным процессом

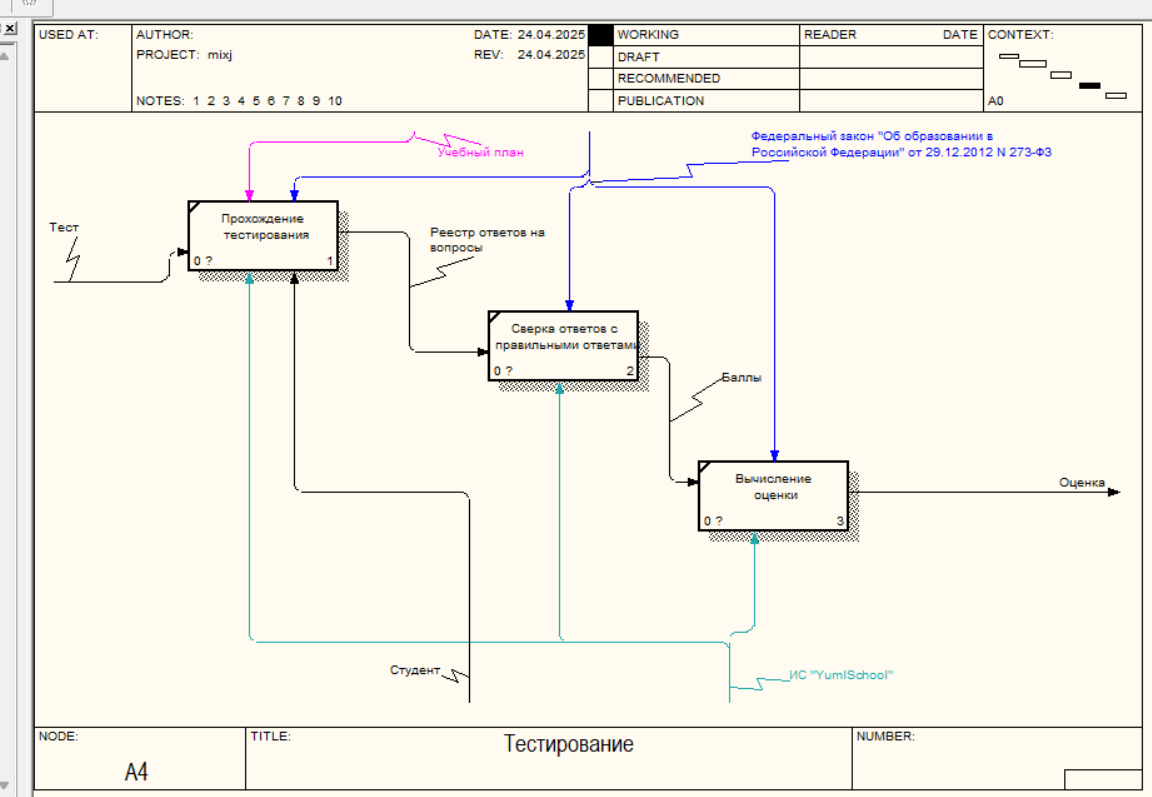


Рисунок - Декомпозиция тестирования

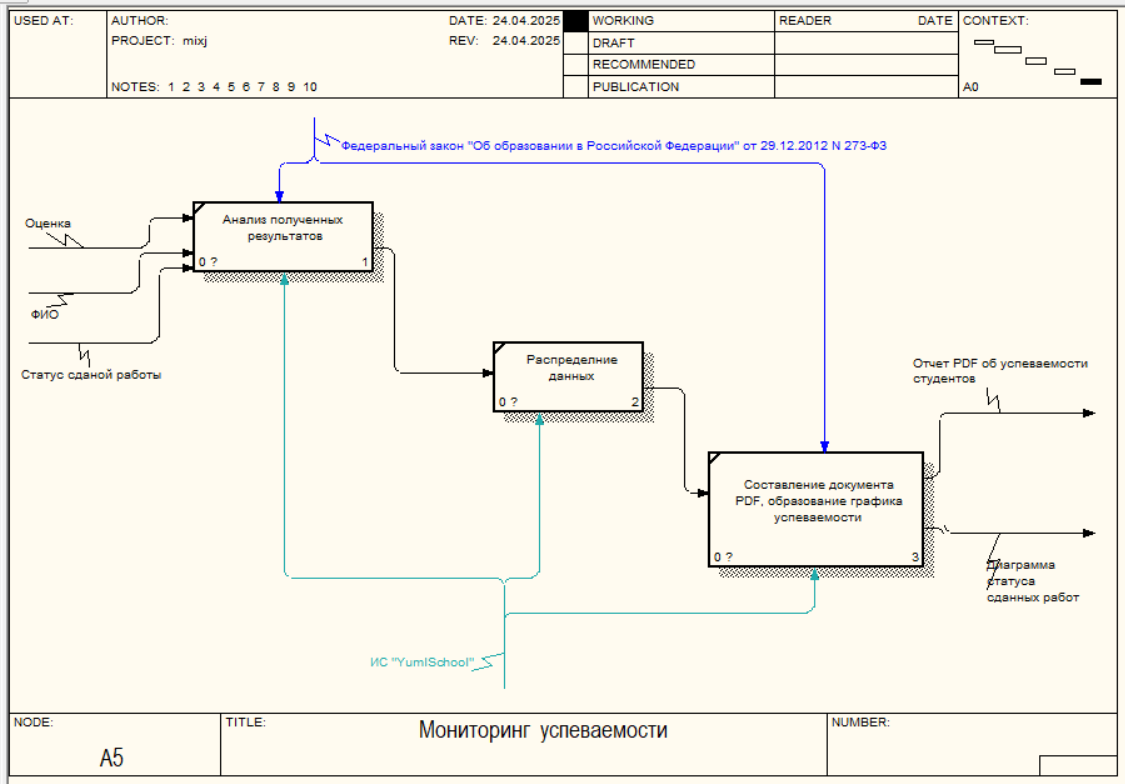


Рисунок - Декомпозиция мониторинга успеваемости

### 2.1.1. Входные данные

Входные данные Веб-приложения представлены в следующем виде:

* ФИО, почта, логин и пароль
* Данные о практической работе (Название, требования, ссылка на файл), лекции (Название, ссылка на файл), тесте (Название, вопросы, ответы, баллы)
* Данные о статусе сданной работы (Статус, оценка)
* Данные о предметах (Название, группа, преподаватель)

### 2.1.2. Выходные данные

Выходные данные Веб-приложения представлены в следующем виде:

* Данные о практической работе, тесте (Оценка, баллы)
* Данные об выгруженной отчетности и диаграммах (Отчет об успеваемости формата PDF, диаграмма статусов работы)
* Данные о пользователях (Список групп, предметов, пользователей)

### 2.1.3. Подробные требования к проекту

Подробные требования к разработке Web-приложения «YumlSchool» представлены в Приложении А. Техническое задание.

## 2.2. Внешняя спецификация

## 2.2.1. Описание задачи

Разработать Web-приложение для контроля освоений знаний и практических умений обучающихся.

Данное программное решение должно состоять из реляционной базы данных под управлением СУБД Microsoft SQL Server Management Studio.

Приложение должно иметь возможность отображения различной информации, хранящейся в базе данных, с помощью технологии API. Помимо отображения данных должно быть предоставлено пользователю возможности добавления, изменения и удаления различной информации.

В приложении предусмотрены три роли:

* Администратор БД отвечает за администрирование системы.
* Преподавателю доступны управление тестами, лекциями, практическими работами и их проверка.
* Студенту доступно взаимодействие с практическими работами, прохождение тестирования, просмотр лекций.

Описание перечня пользователей с их функциями изображено на рисунке 7.

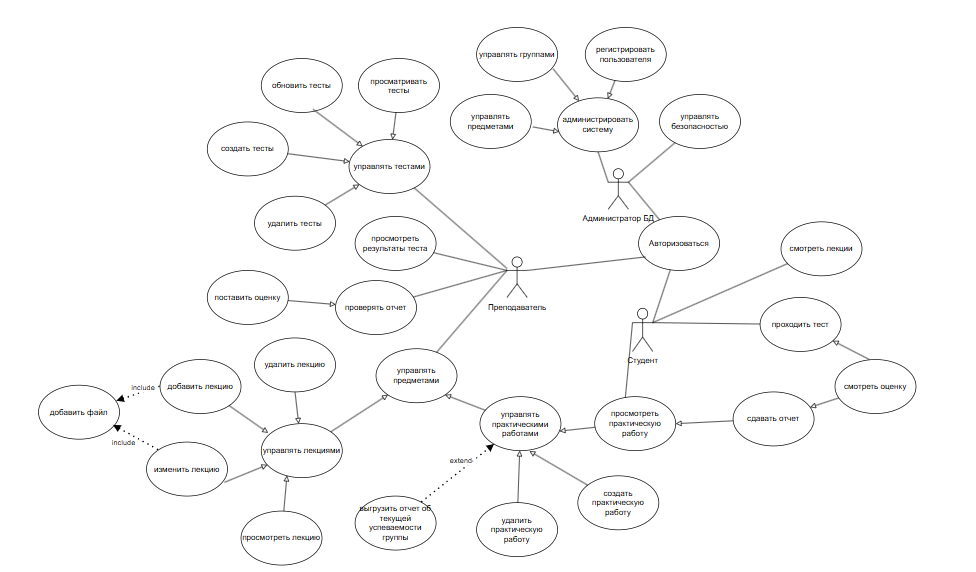


Рисунок - Диаграмма прецендентов

### 2.2.2. Входные и выходные данные

В таблице 4 представлены входные данные.

Таблица 4. Входные данные

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Имя | Тип | | Ограничение | | | Формат ввода | Описание | |
| 1 | 2 | | 3 | | | 4 | 5 | |
| Персональные данные | | | | | | | | |
| Login | | Строка | | [A-Za-z0-9@]  {5, 100} | Поле ввода | | | Логин пользователя |
| Password | | Строка | | [A-Za-z0-9@]  {5, 100} | Поле ввода | | | Пароль пользователя |
| First\_Name | | Строка | | [А-Яа-я]{2, 45} | Поле ввода | | | Имя пользователя |
| Second\_Name | | Строка | | [А-Яа-я]{2, 45} | Поле ввода | | | Фамилия пользователя |
| Middle\_Name | | Строка | | [А-Яа-я]{2, 45} | Поле ввода | | | Отчество пользователя |
| Данные лекции | | | | | | | | |
| Name\_Lection | | Строка | | [A-Za-z0-9@]  {5, 100} | Поле ввода | | | Название лекции |
| Description\_lection | | Строка | | [A-Za-z0-9@]  {5, 100} | Поле ввода | | | Описание лекции |
| URL\_file\_lection | | Строка | | Нет  ограничений | Поле ввода | | | Ссылка на файл лекции |
| Данные практической работы | | | | | | | | |
| Name\_ PracWork | | Строка | | [A-Za-z0-9@]  {5, 100} | Поле ввода | | | Название практической работы |
| Description\_ Name\_ PracWork | | Строка | | [A-Za-z0-9@]  {5, 100} | Поле ввода | | | Описание практической работы |
| URL\_file\_ Name\_ PracWork | | Строка | | Нет  ограничений | Поле ввода | | | Ссылка на файл практической работы |
| Данные о тесте | | | | | | | | |
| Name\_ Test | | Строка | | [A-Za-z0-9@]  {5, 100} | Поле ввода | | | Название теста |
| Description\_ Name\_ Test | | Строка | | [A-Za-z0-9@]  {5, 100} | Поле ввода | | | Описание теста |
| Question | | Строка | | [A-Za-z0-9@]  {5, 100} | Поле ввода | | | Вопрос |
| Answer | | Строка | | [A-Za-z0-9@]  {5, 100} | Поле ввода | | | Ответ |

В таблице 5 представлены выходные данные.

Таблица 5. Выходные данные

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Тип | Ограничение | Описание |
| 1 | 2 | 3 | 4 |
| Выходные данные | | | |
| Персональные данные | | | |
| First\_Name | Строка | [А-Яа-я]{2, 45} | Имя пользователя |
| Second\_Name | Строка | [А-Яа-я]{2, 45} | Фамилия  пользоваться |
|  |  |  |  |

### 2.2.3. Методы

В ходе написания программы были использованы следующие методы разработки программного обеспечения:

При разработке API использовалась методология ООП, а именно абстракция, в виде классов моделей с характеристиками объекта, которым обладает он же в БД, инкапсуляция, в виде методов, позволяющих скрыть подробности характеристик объекта, которые возвращают значения и наследование, каждый контроллер наследуется от родительского класса «ControllerBase».

Также использовалась методология REST API – это архитектурный стиль взаимодействия между клиентов и сервером по протоколу HTTP. REST API использует принципы REST:

• Клиент-серверная архитектура – клиент делает запрос на сервер, сервер обрабатывает запросы и отправляет клиенту ответ

• Отсутствие состояния – сервер не хранит состояние клиента между запросами;

• Единообразие интерфейса – ресурсы представлены в виде URL, для всех ресурсов используется одни и те же HTTP-методы (GET, POST, PUT, DELETE).

Для разработки веб-приложения использовался паттерн MVC (Model-View-Controller), но в рамках фреймворка «Django» он именуется как MTV (Model-Template-View). В данном паттерне подразумевается, что модели (Model) используются для работы с базой данных, описывает структуру данных и логику их обработки; шаблон (Template) отвечает за отображение данных пользователю – это файлы HTML, в которых можно динамически выводить данные, переданные из представлений; представления (View) прослойка между моделями и шаблонами, которая позволяет управлять логикой обработки запросов, т.е. представления запрашивают данные из моделей, после чего передают их в шаблоны для вывода на страницу. Схема паттерна MTV представлена на Рисунке 8.



Рисунок - Схема паттерна MTV

### 2.2.4. Технологии программирования

Для разработки веб-приложения, использовалась технология программирования – Django. Представляющую свою реализацию в виде паттерна MTV (подробнее в п. 2.2.3).

Для разработки веб-API, использовалась технология программирования – ASP.NET Core WEB-API, с использованием EntityFramework.

Для разработки базы данных, использовалась технология программирования – Microsoft SQL Server Management.

2.2.5. Тесты

1. По формальности тестирования.

Тестирование по тестам – тестирование по предварительно написанным тест-кейсам.

2. По исполнению кода.

Динамическое тестирование - во время тестирования код исполняется.

3. По уровню тестирования.

Системное – проверка работы всей системы на соответствие заявленным требованиям к программному продукту.

4. По целям.

Функциональное тестирование направлено на проверку того, какие функции ПО реализованы, и того, насколько верно они реализованы.

5. По степени автоматизации.

Ручное – без использования дополнительных программных средств.

6. По позитивности сценария.

Позитивным – проверка ПО на соответствие ожидаемому поведению. Негативным – проверяет, будет ли ПО работать в случае, когда поведение пользователя отличается от ожидаемого.

7. По знанию системы.

Тестирование «белого ящика» – тестирование программного продукта с доступом к коду.

Тестирование «черного ящика» – тестирование без доступа к коду продукта.

8. По разработке тестовых испытаний.

На основе требований – требование было определено до начала тестирования.

Контроль целостности, описывающих ситуации и реакции приложения на выполнения функций представлен в таблице

Таблица 4 - Тест возможности регистрации

| № | Действие  (входные данные) | Ожидаемый  результат | Фактический  результат | Статус теста (пройден/не пройден) |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| 1 | Логин: vasin3485  Пароль: vasim4ek | Успешная  авторизация | Успешная  авторизация | Тест  пройден |
| 2 | Логин: vasin3485  Пароль: 123 | Авторизация не прошла. Над полями ввода отображается надпись «Неверный пароль» | Авторизация не прошла. Над полями ввода отображается надпись «Неверный пароль» | Тест  пройден |
| 3 | Логин: vasin34851  Пароль: vasim4ek | Авторизация не прошла. Над полями ввода отображается надпись «Такого логина не существует» | Авторизация не прошла. Над полями ввода отображается надпись «Такого логина не существует» | Тест  пройден |
| 4 | Логин: пустота  Пароль: пустота | Авторизация не прошла. Над полем ввода логина и пароля отображается сообщение «Заполните это поле.» | Авторизация не прошла. Над полем ввода логина и пароля отображается сообщение «Заполните это поле.» | Тест  пройден |
| 5 | Логин: vasin3485  Пароль: пустота | Авторизация не прошла. Над полем ввода пароля отображается сообщение «Заполните это поле» | Авторизация не прошла. Над полем ввода пароля отображается сообщение «Заполните это поле» | Тест  пройден |
| 6 | Логин: пустота  Пароль: vasim4ek | Авторизация не прошла. Над полем ввода логина отображается сообщение «Заполните это поле.» | Авторизация не прошла. Над полем ввода логина отображается сообщение «Заполните это поле.» | Тест  Пройден |

## 2.3.Проектирование

### 2.3.1. Схема архитектуры приложения

На Рисунке 9 представлена схема «Клиент-серверной» архитектуры веб-приложения.

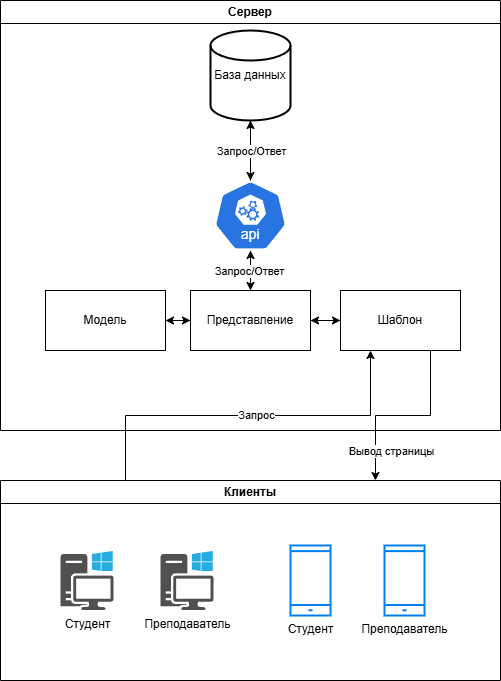


Рисунок - Схема архитектуры веб-приложения

### 2.3.2. Логическая схема данных

Разработанная логическая модель базы данных, представленная на Рисунке 10, послужила основой для реализации логики манипуляции данными в проектируемой базе данных.

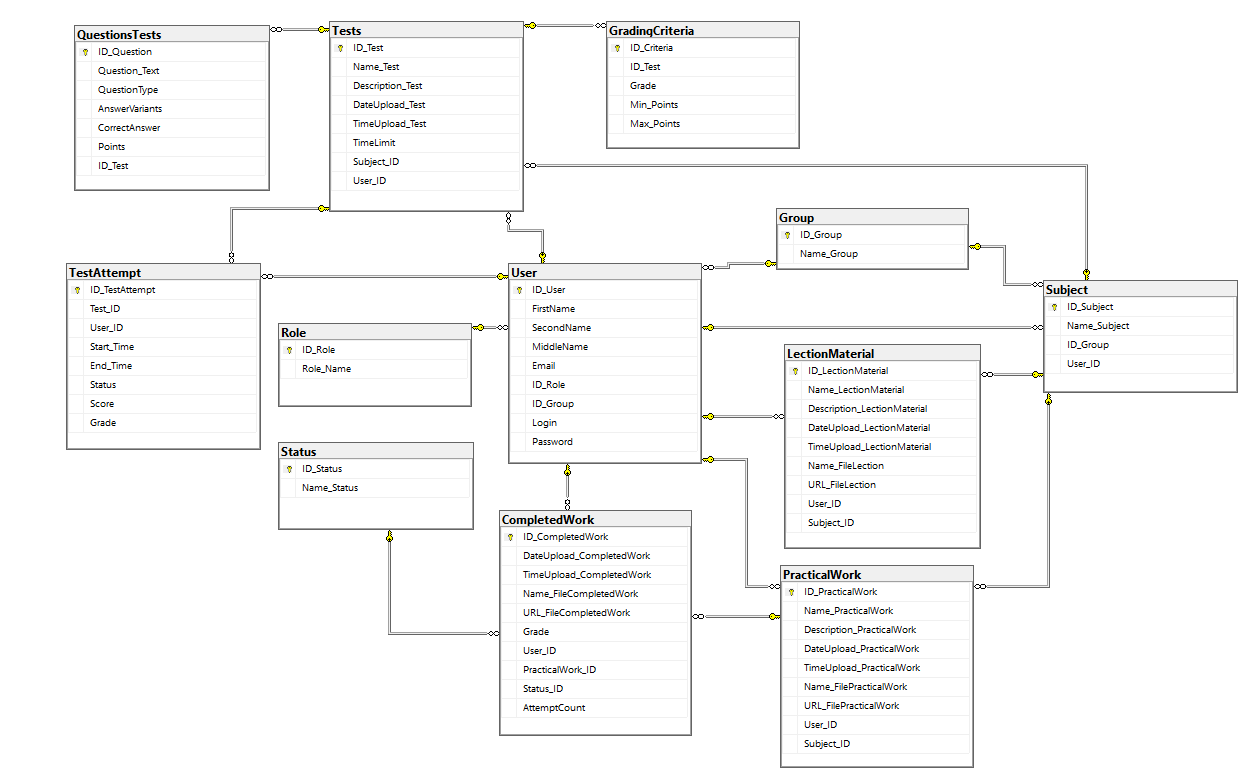


Рисунок - Логическая схема данных

На Рисунке 11 представлена физическая схема данных.

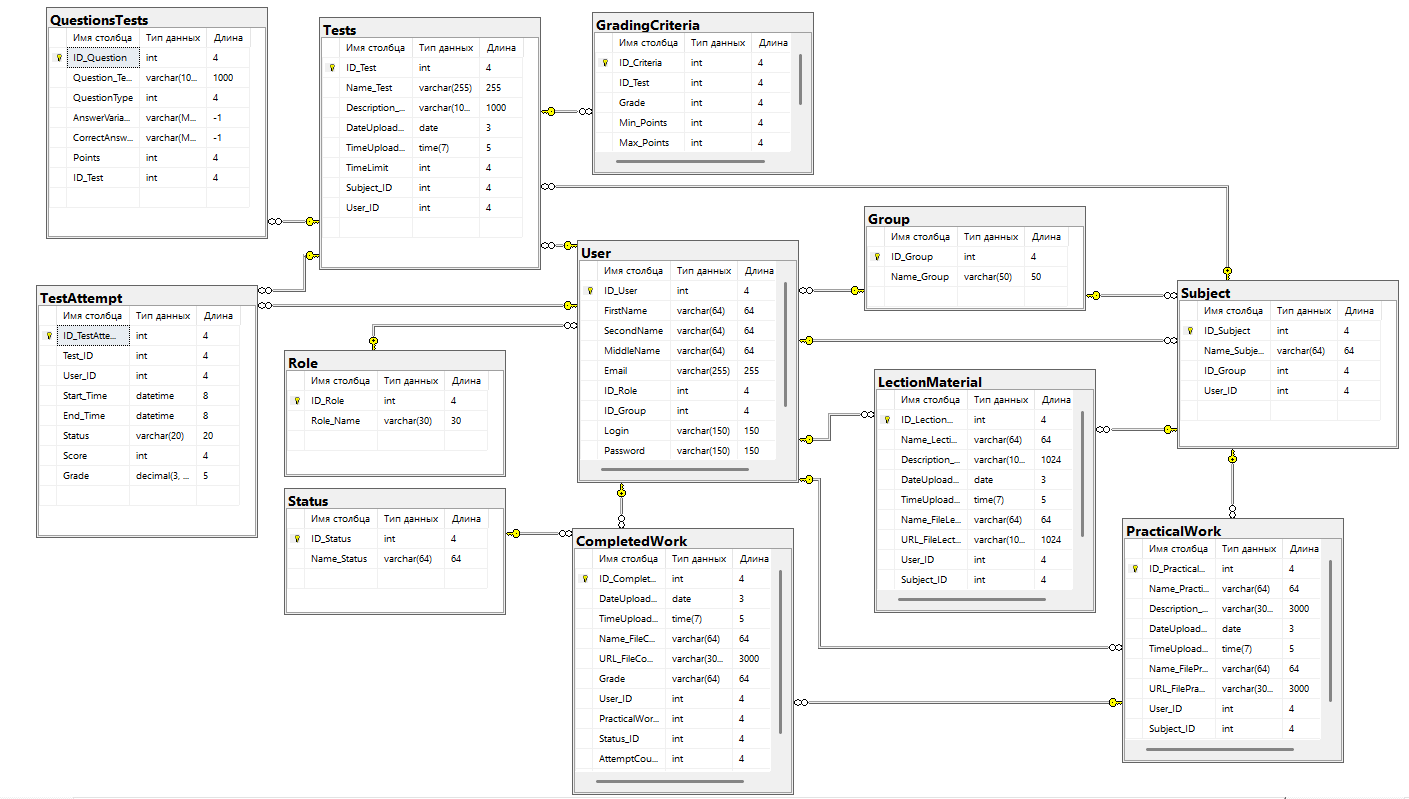


Рисунок - Физическая схема данных

В проектируемой базе данных было выделено 12 сущностей (таблиц), каждая из которых содержит набор атрибутов (полей) для записи информации. В таблице 6 представлен словарь данных реализуемой базы данных для данного мобильного приложения.

Таблица 1. Словарь

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ключ | Поле | Тип данных | Обязательность заполнения | Описание |
| 1 | 2 | 3 | 4 | 5 |
| PK | ID\_TestAttempt | Int | Not Null | Идентификатор попытки теста |
| FK | Test\_ID | int | Not Null | Код теста |
| FK | User\_ID | int | Not Null | Код пользователя |
|  | Start\_Time | datetime | Not Null | Начало времени |
|  | End\_Time | datetime | Не обязательно | Конец времени |
|  | Status | varchar(20) | Not Null | Статус |
|  | Score | int | Not Null | Выполненная работа |
|  | Grade | decimal(3, 1) | Not Null | Оценка за тест |
| PK | ID\_Criteria | int | Not Null | Идентификатор критерия теста |
| FK | ID\_Test | int | Not Null | Код теста |
|  | Grade | int | Not Null | Оценка критерия |
|  | Min\_Points | int | Not Null | Минимальное количество баллов |
|  | Max\_Points | int | Not Null | Максимальное количество баллов |
| PK | ID\_Test | int | Not Null | Идентификатор теста |
|  | Name\_Test | varchar(255) | Not Null | Название |
|  | Description\_Test | varchar(1000) | Не обязательно | Описание |
|  | DateUpload\_Test | date | Не обязательно | Дата выдачи |
|  | TimeUpload\_Test | time(7) | Не обязательно | Время выдачи |
|  | TimeLimit | int | Не обязательно | Лимит теста |
| FK | Subject\_ID | int | Не обязательно | Код предмета |
| FK | User\_ID | int | Не обязательно | Код пользователя |
| PK | ID\_LectionMaterial | int | Not Null | Идентификатор лекции |
|  | Name\_LectionMaterial | varchar(64) | Not Null | Название лекции |
|  | Description\_LectionMaterial | varchar(1024) | Not Null | Описание |
|  | DateUpload\_LectionMaterial | date | Не обязательно | Дата выдачи |
|  | TimeUpload\_LectionMaterial | time(7) | Не обязательно | Время выдачи |
|  | Name\_FileLection | varchar(64) | Not Null | Название для файла лекции |
|  | URL\_FileLection | varchar(1024) | Not Null | URL лекции |
| FK | User\_ID | int | Не обязательно | Код пользователя |
| FK | Subject\_ID | int | Не обязательно | Код предмета |
| PK | ID\_Subject | int | Not Null | Идентификатор предмета |
|  | Name\_Subject | varchar(64) | Not Null | Название предмета |
| FK | ID\_Group | int | Not Null | Код группы |
| FK | User\_ID | int | Not Null | Код пользователя |
| PK | ID\_Group | int | Not Null | Идентификатор группы |
|  | Name\_Group | varchar(50) | Not Null | Название группы |
| PK | ID\_Question | int | Not Null | Идентификатор вопроса в тесте |
|  | Question\_Text | varchar(1000) | Not Null | Текст вопроса |
|  | QuestionType | int | Not Null | Тип вопроса |
|  | AnswerVariants | varchar(MAX) | Not Null | Вариант ответа |
|  | CorrectAnswer | varchar(MAX) | Not Null | Правильный ответ |
|  | Points | int | Не обязательно | Баллы за тест |
| FK | ID\_Test | int | Не обязательно | Код теста |
| PK | ID\_User | int | Not Null | Идентификатор пользователя |
|  | FirstName | varchar(64) | Not Null | Имя |
|  | SecondName | varchar(64) | Not Null | Фамилия |
|  | MiddleName | varchar(64) | Не обязательно | Отчество |
|  | Email | varchar(255) | Not Null | Почта |
| FK | ID\_Role | int | Not Null | Код роли |
| FK | ID\_Group | int | Not Null | Код группы |
|  | Login | varchar(150) | Not Null | Логин |
|  | Password | varchar(150) | Not Null | Пароль |
| PK | ID\_Group | int | Not Null | Идентификатор группы |
|  | Name\_Group | varchar(50) | Not Null | Название группы |
| PK | ID\_Role | int | Not Null | Идентификатор роли |
|  | Role\_Name | varchar(30) | Not Null | Название роли |
| PK | ID\_Status | int | Not Null | Идентификатор статуса |
|  | Name\_Status | varchar(64) | Not Null | Название статуса |
| PK | ID\_CompletedWork | int | Not Null | Идентификатор выполненной работы |
|  | DateUpload\_CompletedWork | date | Не обязательно | Дата сдачи |
|  | TimeUpload\_CompletedWork | time(7) | Не обязательно | Время сдачи |
|  | Name\_FileCompletedWork | varchar(64) | Not Null | Название файл |
|  | URL\_FileCompletedWork | varchar(3000) | Not Null | URL файла |
|  | Grade | varchar(64) | Not Null | Оценка |
| FK | User\_ID | int | Не обязательно | Код пользователя |
| FK | PracticalWork\_ID | int | Не обязательно | Код практической работы |
| FK | Status\_ID | int | Не обязательно | Статус работы |
|  | AttemptCount | int | Не обязательно | Количество попыток |
| PK | ID\_PracticalWork | int | Not Null | Идентификатор практической работы |
|  | Name\_PracticalWork | varchar(64) | Not Null | Название практической |
|  | Description\_PracticalWork | varchar(3000) | Not Null | Описание |
|  | DateUpload\_PracticalWork | date | Не обязательно | Дата выдачи |
|  | TimeUpload\_PracticalWork | time(7) | Не обязательно | Время выдачи |
|  | Name\_FilePracticalWork | varchar(64) | Not Null | Название файла |
|  | URL\_FilePracticalWork | varchar(3000) | Not Null | URL файла |
| FK | User\_ID | int | Не обязательно | Код пользователя |
| FK | Subject\_ID | int | Не обязательно | Код предмета |

### 2.3.3. Диаграмма классов

На рисунке 12 представлена диаграмма классов Web-приложения. На ней графически изображены классы, которые состоят из полей, методов и свойств.

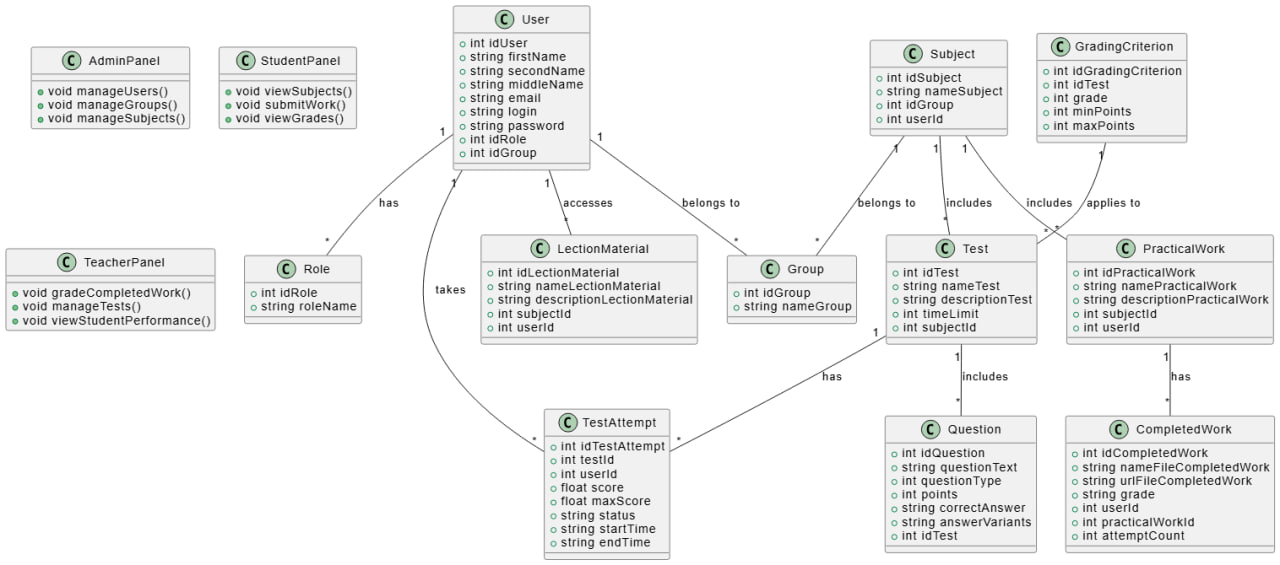


Рисунок 12 - Диаграмма классов

### 2.3.4. Функциональная схема

На рисунке 13 изображена функциональная схема

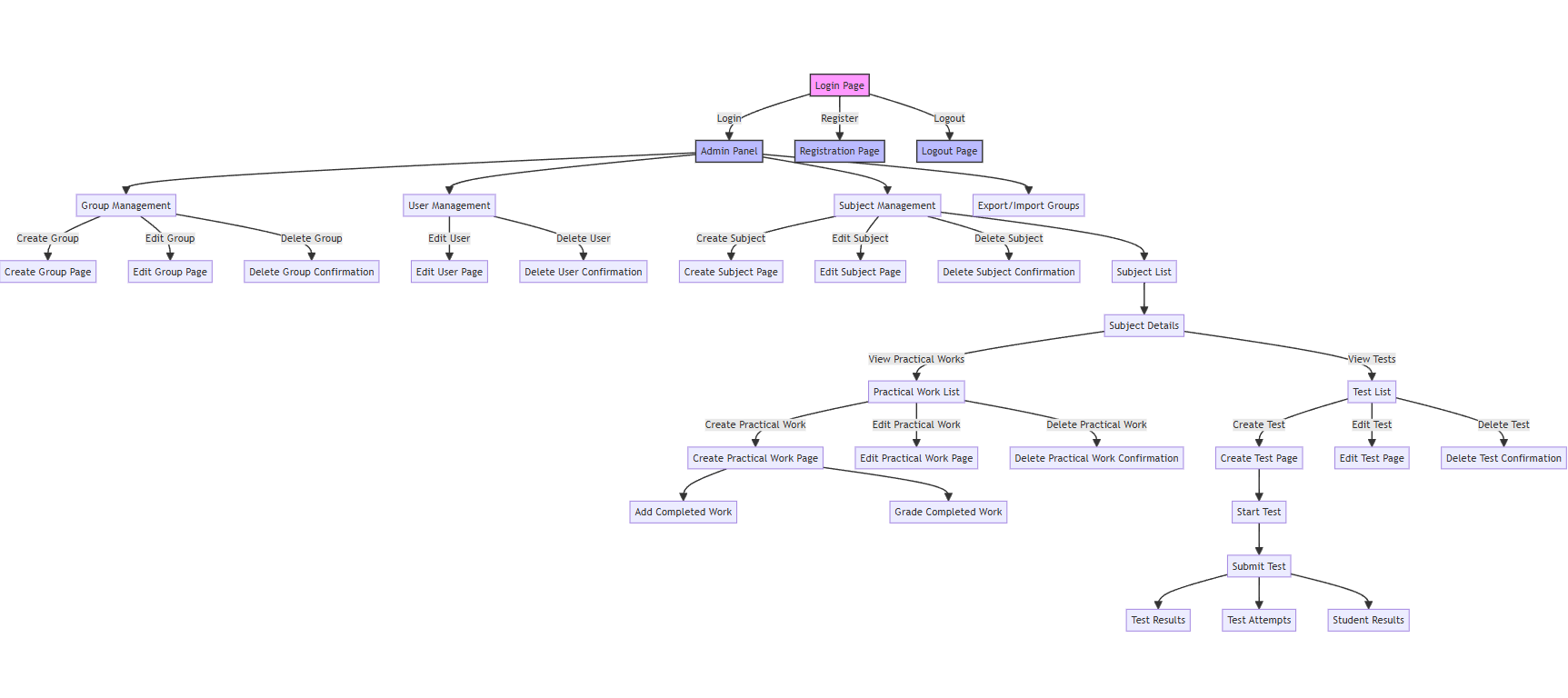


Рисунок 13 - Функциональная схема

## 2.4. Результат работы программы

На рисунке 14 видны элементы основного окна для студента, здесь видны:

- Данные о сессии пользователя, его фамилия и имя, кнопка «Выход» - для выхода из системы и возврата на страницу авторизации.

- Данные о предметах пользователя в которых он состоит. Здесь видны название предмета, группа, преподаватель предмета. По нажатию на предмет происходит переход на страницу предмета.

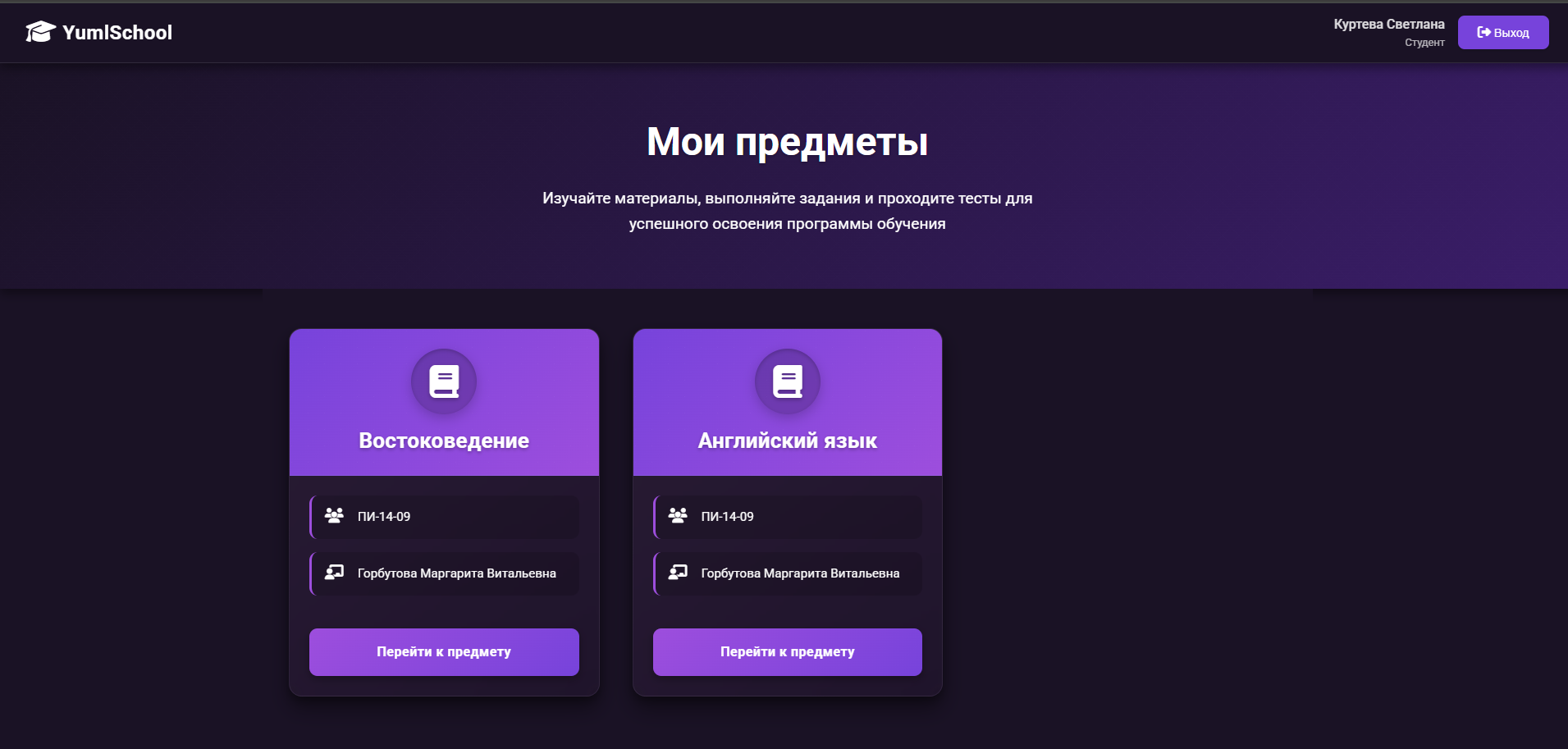


Рисунок - Элементы основного окна у студента

На рисунке 15 изображена авторизация пользователей

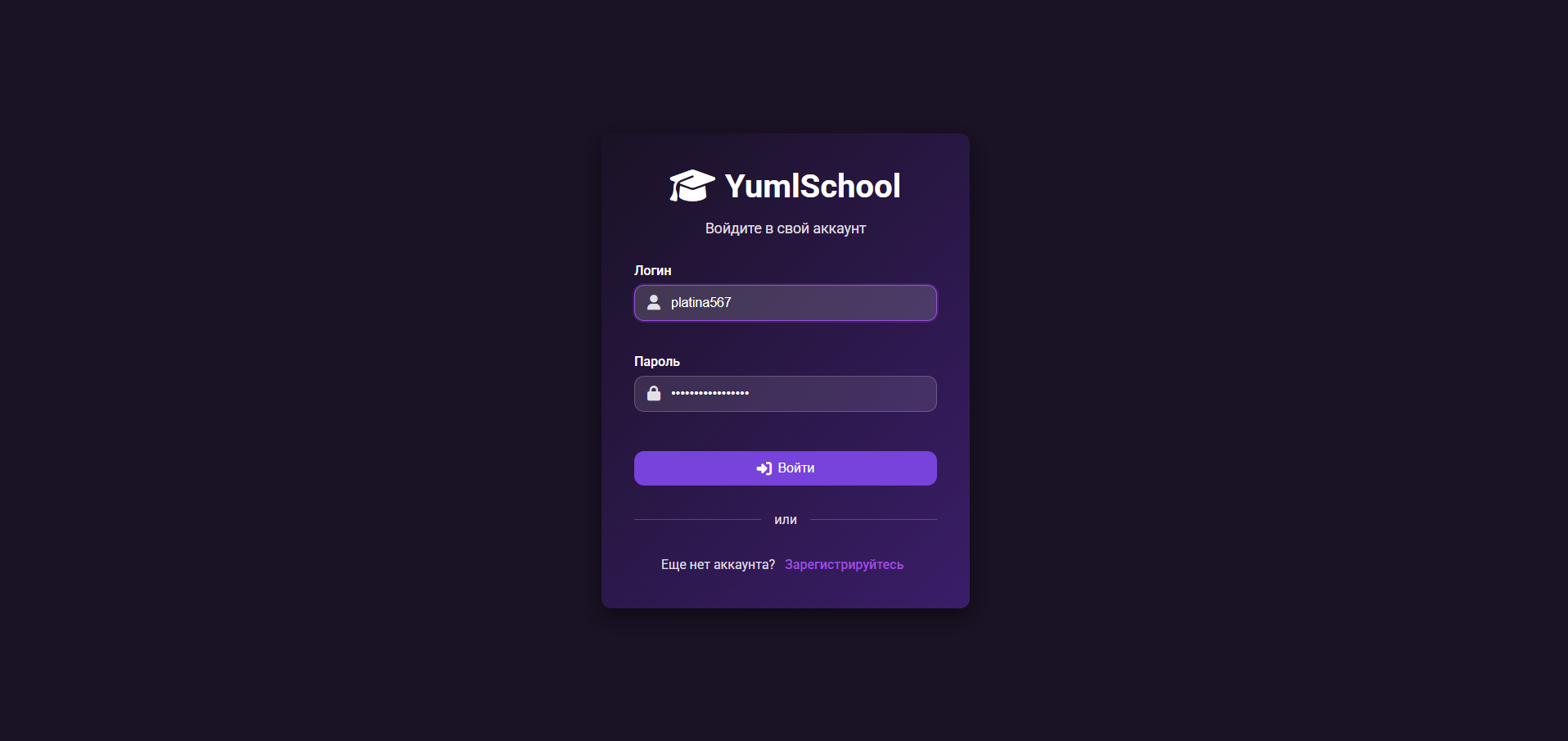


Рисунок – Авторизация

На рисунке 16 изображена страница результатов тестирования

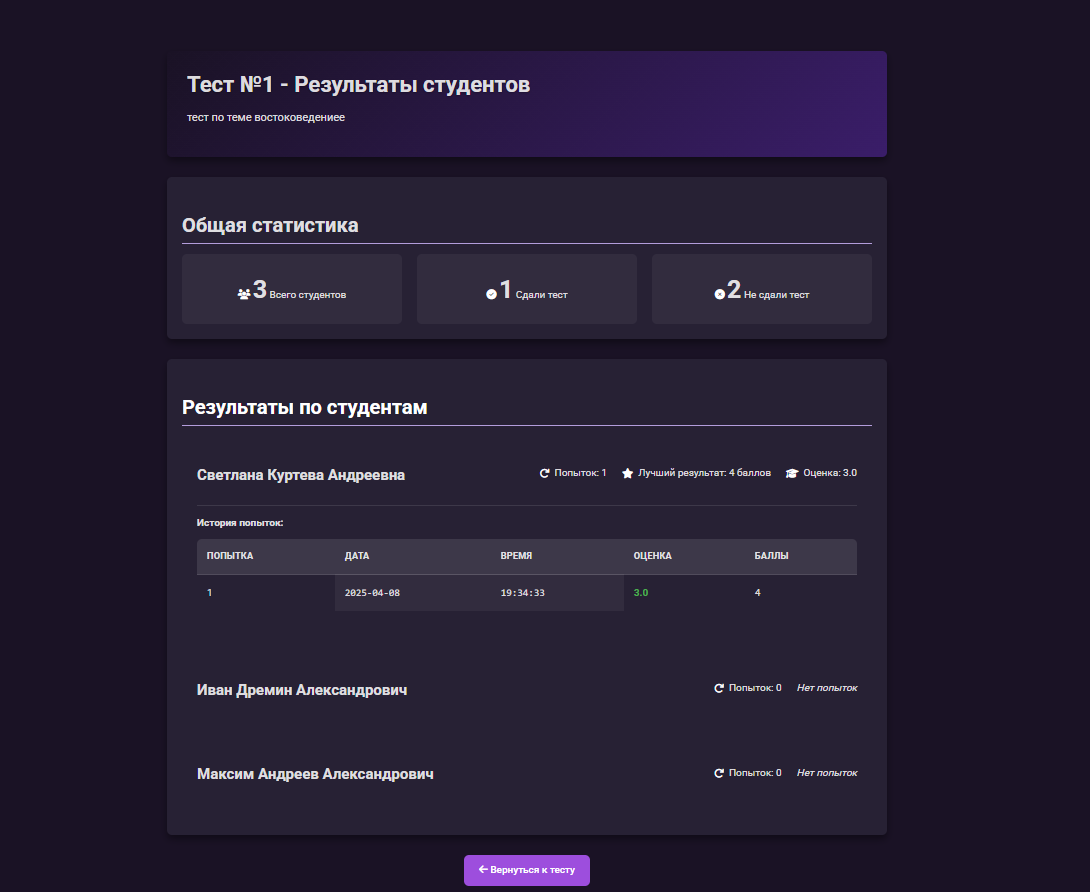


Рисунок - Результаты тестирования

# 3. ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ

## 3.1. Инструментальная среда разработки

Для написания базы данных была выбрана СУБД – SSMS. Выбор обусловлен тем, что разработка серверной части подразумевается на языке программирования «C#», с использованием технологии «ASP.NET Core».

Для написания Веб-API был выбран язык программирования «C#» и интеграционная среда разработки «Visual Studio 2022» и технология программирования – Веб-API ASP.Net Core. Выбор был сделан в пользу данного инструмента потому, что Visual Studio имеет непосредственно напрямую подключаться к базе данных СУБД SSMS. Также не маловажным фактом было наличие обширного количества сторонних библиотек, добавление проект которых, происходит по средству использования пакетов NuGet. Фреймворк, использованный при разработке API, был EntityFramework, позволяющий автоматически, создать модели, контроллеры и контекст базы данных, на основе подключения к самой базе данных, с последующей возможность вручную изменять, добавлять или же удалять модели или контроллеры, добавляя необходимые эндпоинты. Также EntityFramework имеет множество функций, позволяющих выполнять LINQ-запросы с List в асинхронном потоке, что значительно ускоряет выполнения и отправки ответа назад клиенту.

В качестве редактора программного кода, для верстки страниц был выбран «Visual Studio Code 1.78.2». Главным фактором в выборе данного редактора было наличие маркетплейса с внушительным количеством расширений, которые позволяют упростить и сделать процесс разработки комфортней и продуктивней. Второстепенным фактором является возможность работы с файлами любого формата.

Для написания веб-приложения «Система онлайн-образования» был выбран язык программирования «Python», текстовый редактор «Microsoft Visual Studio Code» и технология программирования – «Django», данный выбор обусловлен тем, что «Django» предоставляет всё необходимое для разработки веб-приложения изначально, что позволяет использовать уже готовые и протестированные функции вместо реализации их с нуля. Структура проекта «Django» позволяет реализовывать разбиение логики на приложения – это означает, что проект можно без усилий масштабировать, добавляя в него новый функционал. «Django» автоматически предоставляет защиту для веб-приложения от таких критических уязвимостей как: XSS и CSRF-атак, и SQL-инъекций.

# Заключение

Главной задачей данной курсовой работы было: оптимизировать учебный процесс и автоматизировать деятельность преподавателя по выдаче тестов и их проверки для оценивания знаний студентов, по средству реализации веб-приложения «Системы онлайн-образования» с возможностью управления курсами: создавать их, изменять, приглашать студентов, и материалами: загружать материалы по темам в курсы со студентами; проведением тестирования: создание тестирований, добавление в них вопросов, с возможностью указания вариантов ответа и выбором правильного для того, чтобы автоматизировать их проверку и подведения оценки знаний.

В результате выполнения курсовой работы, были получены навыки по работе с передачей файлов между клиентом и сервером, с возможностью реализации функции скачивания это файла с веб-приложения на устройство клиента. Были получены навыки по реализации WEB-API с использованием фреймворка «ASP.NET Core».

В результате выполнения курсовой работы была разработана система онлайн-образования, которая закрывала проблематику, описанную в задаче курсовой работы, была достигнута поставленная цель по повышению навыков реализации веб-приложения с использованием фреймворка «Django», реализации API с использованием фреймворка «ASP.NET Core», реализации валидации навыков на стороне сервера базы данных. Преимуществами системы можно выделить: . Минусами системы можно выделить: .

# Список используемых материалов

1. Django введение - Изучение веб-разработки - MDN Web Docs, URL - <https://developer.mozilla.org/ru/docs/Learn_web_development/Extensions/Server-side/Django/Introduction> (Дата обращения: 23.09.2024)
2. Руководство по созданию веб-API с помощью ASP.NET Core, URL - <https://learn.microsoft.com/ru-ru/aspnet/core/tutorials/first-web-api?view=aspnetcore-9.0> (Дата обращения: 24.09.2024)
3. ASP.NET Core и C# | Полное руководство – Metanit, URL - <https://metanit.com/sharp/aspnet6/> (Дата обращения: 24.09.2024)
4. Руководство. Создание минимального API с помощью ASP.NET Core, URL - <https://learn.microsoft.com/ru-ru/aspnet/core/tutorials/min-web-api?view=aspnetcore-9.0> (Дата обращения: 24.09.2024)
5. Введение в Django: создание веб-приложений на Python – Skypro, URL - <https://sky.pro/wiki/python/vvedenie-v-django-sozdanie-veb-prilozhenij-na-python/> (Дата обращения: 23.09.2024)
6. Общие сведения об Entity Framework, URL - <https://learn.microsoft.com/ru-ru/dotnet/framework/data/adonet/ef/overview> (Дата обращения: 16.10.2024)
7. Официальная документация Django, URL - <https://docs.djangoproject.com/ru/stable/> (Дата обращения: 24.10.2024)
8. Документация по ASP.NET Core, URL - <https://learn.microsoft.com/ru-ru/aspnet/core/?view=aspnetcore-9.0> (Дата обращения: 24.10.2024)
9. Документация по Entity Framework, URL - <https://learn.microsoft.com/ru-ru/ef/> (Дата обращения: 16.10.2024)
10. HTML: HyperText Markup Language MDN, URL - <https://developer.mozilla.org/ru/docs/Web/HTML> (Дата обращения: 12.10.2024)
11. CSS: Каскадные таблицы стилей MDN, URL - <https://developer.mozilla.org/ru/docs/Web/CSS> (Дата обращения: 12.10.2024)
12. Интеграция Django с внешними API, URL - <https://sky.pro/wiki/python/integraciya-django-s-vneshnimi-api/> (Дата обращения: 20.11.2024)
13. Понимание архитектуры Django: паттерн MTV, URL - <https://dev.to/vincenttommi/2-understanding-djangos-architecture-the-mtv-pattern-1gl> (Дата обращения: 27.10.2024)
14. Архитектура в Django проектах, URL - <https://habr.com/ru/companies/vivid_money/articles/544856/> (Дата обращения: 27.10.2024)
15. Основы Django: Архитектура Model-View-Template (MVT), URL - <https://angelogentileiii.medium.com/basics-of-django-model-view-template-mvt-architecture-8585aecffbf6> (Дата обращения: 27.10.2024)
16. Архитектура REST, URL - <https://habr.com/ru/articles/38730/> (Дата обращения: 24.09.2024)
17. REST API: принципы, применение, URL - <https://gb.ru/blog/rest-api/> (Дата обращения: 25.09.2024)
18. Что такое RESTful API, URL - <https://aws.amazon.com/ru/what-is/restful-api/> (Дата обращения: 25.09.2024)
19. Что такое RESTful на самом деле, URL - <https://habr.com/ru/companies/hexlet/articles/274675/> (Дата обращения: 26.09.2024)
20. Профессиональная разработка технической документации – ГОСТ 34, URL - [https://www.swrit.ru/gost-34.html](https://www.swrit.ru/gost-34.html%20) (Дата обращения 25.11.2024)