

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего
образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет
вычислительной техники

Кафедра
«Вычислительная техника»

Направление подготовки: 09.03.01 «Информатика и вычислительная техника»

Профиль подготовки: «Программное обеспечение средств вычислительной
техники и автоматизированных систем»

БАКАЛАВРСКАЯ РАБОТА

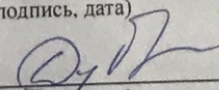
на тему

СИСТЕМА КОНТРОЛЯ УСПЕВАЕМОСТИ
УЧАЩИХСЯ

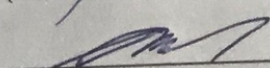
Студент

 Анохин Андрей Александрович
(подпись, дата) (ФИО полностью)

Руководитель

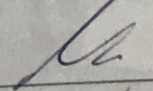
 А.В. Дубравин
(подпись, дата) (фамилия, инициалы)

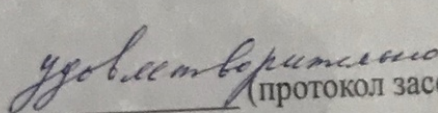
Нормоконтролёр

 А.С. Бычков
(подпись, дата) (фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от . 2023 г. №)

Заведующий кафедрой

 М.А. Митрохин
(подпись) (фамилия, инициалы)

Работа защищена с отметкой  удовлетворительно протокол заседания ГЭК от

27.06.2023г. № 32)

Секретарь ГЭК

 Е.И. Гурин
(подпись) (фамилия, инициалы)

Пенза 2023

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет
вычислительной техники

Кафедра
«Вычислительная техника»

«Утверждаю»
Заведующий кафедрой
М.А. Митрохин
«26» 04 2023г.

ЗАДАНИЕ
НА БАКАЛАВРСКУЮ РАБОТУ

1. Студент Анохин Андрей Александрович гр. 19ВВП1 факультета вычислительной техники.

2. Тема работы «Система контроля успеваемости учащихся»

Тема утверждена приказом ПГУ № 458/0 от "26" 04 2023 г.

3. Руководитель бакалаврской работы к.т.н., доцент Дубравин А.В.

4. Задание на работу (назначение разработки, исходные данные и т.п.)

Разработать приложение «Система контроля успеваемости учащихся», выполняющее следующие функции:

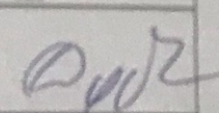
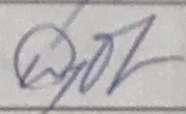
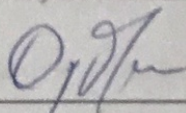
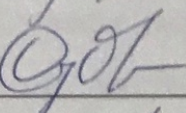
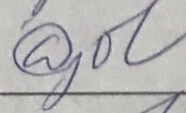
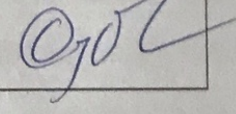
- авторизация и регистрация разных типов пользователей: преподаватель, ученик;
- оценка преподавателем учебной успеваемости учащихся;
- контроля учениками и преподавателями учебной успеваемости учащихся;

Приложение должно быть спроектировано по принципу клиент-серверной архитектуры. Клиентская часть программы должна предоставлять интерфейс для работы с приложением в виде веб – сайта в браузере. Серверная часть должна предоставлять возможность обработки и хранения в базе данных пользовательских данных, таких как оценки по различным предметам, актуальные напоминания и т. д.

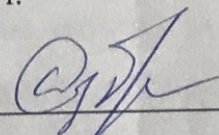
5. Перечень подлежащих разработке вопросов

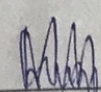
Требуется произвести анализ предметных областей: изучить аналоги существующих систем контроля успеваемости учащихся, изучить способы проектирования и разработки собственного клиент-серверного приложения с применением REST API методологии. Также необходимо изучить функционал фреймворка Django и языка программирования Python.

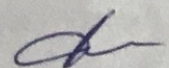
6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объ- ем рабо- ты	Срок выполне- ния	Подпись руководите- ля
1.	Изучение предметной области	5 0	26.04.2023 - 30.04.2023	
2.	Анализ требований	1 0	31.04.2023 - 02.05.2023	
3.	Выбор технологий, применяемых при проектировании и разработке	5 0	03.05.2023 - 07.05.2023	
4.	Описание алгоритмов	4 0	08.05.2023 - 11.05.2023	
5.	Разработка приложения «Система контроля успеваемости учащихся»	1 5 0	12.05.2023 - 29.05.2023	
6.	Оформление пояснительной записки	2 4	30.05.2023 - 31.05.2023	

Дата выдачи задания "26" апреля 2023 г.

Руководитель бакалаврской работы  Дубравин А.В.
(подпись, дата)

Задание к исполнению принял студент  Анохин А.А.
(подпись, дата)

Работу к защите допустить
Декан факультета  Фионова Л.Р.
(подпись, дата)
27.06.2023

Реферат

Пояснительная записка выпускной квалификационной работы
содержит 140 страниц формата А4, 24 рисунка, 9 источников, 2 приложения
WEB - ПРИЛОЖЕНИЕ, REST API, PYTHON, DJANGO, SQLite

Выбранным объектом разработки является клиент-серверное WEB-приложение, которое позволяет как ученикам, так и преподавателям легко вести контроля успеваемости учащихся.

Цель работы – разработать систему контроля успеваемости учащихся. Приложение должно обладать следующим функционалом: ученик может просматривать свои оценки, различные напоминания от учителей, загруженные домашние задания, список учителей, а также имеет возможность обновить свой профиль и поменять пароль. Учитель имеет возможность загружать домашние задания для всех учеников, загружать напоминания, просматривать список всех студентов, просматривать студентов своей группы, а также выставлять им оценки.

В ходе разработки данного *WEB*-приложения был проведен анализ конкурентоспособности проекта, потребностей конечных пользователей приложения, т.е. учеников и преподавателей, выбор инструментов для разработки *WEB*-приложения, раздел БЖД.

Результатом работы над вышеописанными пунктами стало клиент-серверное *WEB*-приложение для контроля успеваемости учащихся.

					ПГУ1.090301.01.001 ПЗ			
Изм.	Лист	№ докум.	Подпись	Дата	Система контроля успеваемости учащихся Пояснительная записка	Лит.	Лист	Листов
Решено		Алексеев А.А.		2019			2	142
Получено		Дубравин А.В.				ФВТ, каф. ВТ гр. 1988П1		
Н. Контроль		Белов А.С.						
Решено		Михайлов В.А.						

Содержание

Перечень сокращений и обозначений.....	4
Введение.....	5
1 Система контроля успеваемости учащихся.....	7
1.1 Анализ предметной области.....	7
1.1.1 Анализ аналогов.....	8
1.1.2 Анализ стека технологий	10
1.1.3 Выбор СУБД.....	14
1.1.4 Выбор среды разработки	17
1.2 Проектирование и разработка.....	19
1.2.1 Реализация серверной части	20
1.2.2 Реализация БД	24
1.2.3 Разработка клиентской части.....	28
1.3 Анализ результатов	30
1.3.1 Тестирование приложения	31
1.3.2 Установка и проверка работоспособности <i>Python</i>	35
1.3.3 – Установка и проверка работоспособности DBrowser.	36
2 Охрана труда.....	38
2.1 Характеристики потенциальных опасностей и вредностей,которые могут возникнуть при работе с ЭВМ.....	39
2.2 Санитарно-гигиеническая характеристика производства	40
Заключение	41
Список используемых источников.....	42
Приложение А – UML - диаграммы.....	43
Приложение Б – Листинг программы	48
Приложение В - Презентация	142

Перечень сокращений и обозначений

В настоящей работе применяют следующие сокращения и обозначения

БД – база данных

СУБД – система управления базами данных

Фреймворк – программная платформа для разработки программного обеспечения

API (Application Programming Interface) – программный интерфейс приложения

CRUD – акроним, обозначающий четыре базовые функции, используемые при работе с базами данных: создание (англ. create), чтение (read), модификация (update), удаление (delete)

HTTP – протокол передачи гипертекста

MVT (Model-View-Template) – «Модель-Представление-Шаблон» или «Модель-Вид-Шаблон»

URL – унифицированный способ идентификации ресурсов

				<i>ПГУ1.090301.01.001 ПЗ</i>	Лист
					4
Изм.	Лист	№ докум	Подпись Дата		

Введение

На сегодняшний день информационные системы стали неотъемлемой частью нашей жизни. Они выполняют огромное количество различных функций и позволяют в несколько кликов совершить огромное количество действий: заказать себе еду, такси, записать на прием к парикмахеру, купить себе билет на поезд или авиабилет, легко и быстро найти информацию в поисковых сетях. Описанные выше примеры – это примеры всего лишь из частной жизни, но информационные системы сегодня используются также в продажах, финансовой сфере, производстве, научной деятельности и т.д. Все это подчеркивает огромную их значимость в нашей жизни.

Современные информационные системы, которые мы сегодня используем, состоят в основном из двух частей: клиентская часть и серверная. Клиентская часть должна предоставлять для пользователя удобный юзер - френдли интерфейс, в котором он сможет быстро сориентироваться и выполнить необходимые задачи. Серверной же части отводится роль обработки запросов, поступающих с клиентской части и выполнение в связи с этим каких-либо действий: отправку запросов к другим сервисам, запросы к базе данных или просто некая манипуляция с данными и возвращение их назад в новом виде. Цель информационных систем, построенных по данному принципу, состоит в том, чтобы создать для пользователя доступное и быстрое решение каких-либо его проблем. Клиент-серверные приложения чаще всего можно открыть с любого устройства через браузер, а у многих из компаний, работающих по данному технологическому принципу, есть и мобильные приложения, построенные на той же основе.

Цели данного проекта:

– анализ и изучение различных современных инструментов разработки веб-приложений;

				ПГУ1.090301.01.001 ПЗ	Лист
					5
Изм.	Лист	№ докум	Подпись Дата		

– выбор на основе анализа необходимых инструментов – основного стека приложения;

– готовая к работе система контроля успеваемости учащихся.

Для достижения поставленных целей следует:

– выявить потребности потенциальных пользователей приложения;

– проанализировать существующие на рынке аналоги;

– провести анализ современных инструментов разработки клиент-серверных приложений;

– выбрать стек технологий, наиболее удовлетворяющий поставленной задаче;

– продумать структуру и основные модули, из которых будет состоять приложение;

– продумать структуру базы данных для данного приложения;

– написать кодовую базу приложения;

– провести различные виды тестирования данного приложения.

				ПГУ1.090301.01.001 ПЗ	Лист
					6
Изм.	Лист	№ докум	Подпись Дата		

1 Система контроля успеваемости учащихся

1.1 Анализ предметной области

Информационные системы все больше автоматизируют различные процессы в окружающем нас мире. Автоматизация помогает людям отслеживать различные показатели в финансовой сфере, медицине, спорте и т. д. На основе сбора данных показателей можно оперативно отслеживать динамику их изменения (в положительную или отрицательную сторону), а также вырабатывать общую стратегию управления каким-либо ресурсом, будь то система контроля показателей АЭС или система контроля жизнедеятельности человека и его активность за день.

Системы контроля способны хранить и эффективно обрабатывать огромное количество данных, будь это данные, конкретно введенные пользователем или полученные с каких-либо устройств вывода, например, на производстве. Также они, чаще всего, обладают удобным для человека интерфейсом, что позволяет активно изучать полученные показатели пользователям данной системы и делать на их основе какие-либо выводы. Помимо этого, они дают возможность добавлять данные, изменять их и удалять, однако часто в таких системах стоят различного рода ограничения для того, чтобы разделять пользователей системы по группам и давать различным группам различные права, тем самым ограничивая данные от использования и изменения. Благодаря всем вышеописанным функциям системы контроля на сегодняшний день активно используются повсеместно, в т.ч. и в учебном процессе. С помощью них у учеников появляется удобный источник доступа к различным заданиям, материалам и т.д., а также к своим оценкам, а у учителей и родителей, в свою очередь, появляется возможность прозрачно отслеживать учебный процесс каждого отдельно взятого студента.

				ПГУ1.090301.01.001 ПЗ	Лист
					7
Изм.	Лист	№ докум	Подпись Дата		

1.1.1 Анализ аналогов

На сегодняшний день существует несколько аналогов проектов, над которым проводилась работа в рамках данной дипломной работы. Рассмотрим некоторые из них.

Сервис “Госуслуги Пензенская область” предоставляет возможность доступа к сервису “Электронная школа”. Изображение данного сервиса с официального сайта представлено на рисунке 1.1.

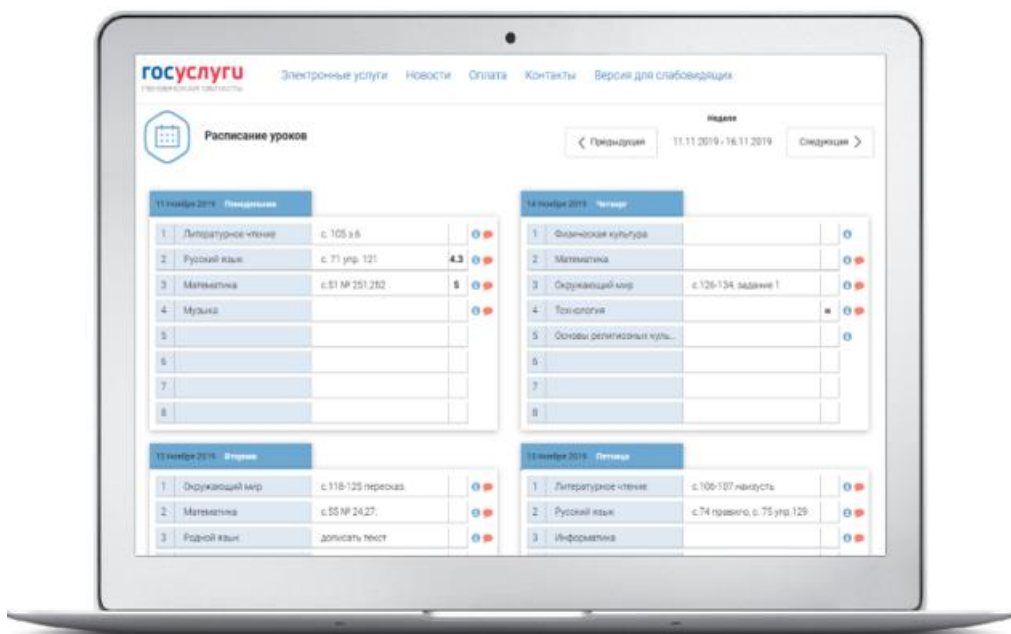


Рисунок 1.1 – Сервис “Электронная школа”

У данного сервиса можно выделить ряд преимуществ и недостатков.

Преимущества данного сервиса:

- большое разнообразие различных функций: можно просматривать оценки ученика, используя различным фильтрам по датам, просматривать средний балл, просматривать домашние задания, а также результаты ОГЭ и ЕГЭ;
- красивый современный понятный интерфейс, имеющий версию даже для слабовидящих людей;
- можно также использовать сервис для работы с картой школьного питания.

Недостатки приложения:

- для использования приложения нужно обязательно иметь аккаунт на “Госуслугах”, что может подходить далеко не всем пользователям по различным причинам;

- нельзя напрямую связаться с учителем через сервис.

Еще одним приложением для контроля успеваемости учащихся является сайт “Дневник.ру”, который позволяет также просматривать и контролировать различные части учебного процесса. Снимок экрана данного сайта представлен на рисунке 1.2.

№	Предметы	Оценки	Отсутствия	Пропуски	Средний балл	1 семестр
			Всего	По болезни		Итог
1	Астрономия	5 5 4	0	0	4.66	
2	Геометрич. задачи	1 2 3 4 5	0	0		
3	Иностранный язык	4 5 5 4 5 4 4 5 4 5	0	0	4.57	
4	Информатика	4 4 5 5 4	0	0	4.4	
5	История	5 4 5 4 1	0	2	4.5	
6	Литература	4 5 4 4 5 5 3 4	0	0	4	
7	Математика	4 5 4 5 1 4 5 5 4 1	0	4	4.5	
8	ОБЖ	4 1 5 5	0	1	4.66	
9	Рус. язык	5 4 4	0	0	4.33	
10	Русская словесность	4	0	0	4	
11	Физика	4 4 5 1 4 5 5 4 1 1	0	3	4.5	
12	Физкультура	4 5 4 5 4 5 4 5 5 4	0	0	4.54	
13	Химия	4	0	0	4	
14	ЭК (алгебра)		0	0		
15	ЭК (физика)	5 1 4 5 5	0	1	4.75	
16	ЭК (Черчение)	3 3 4	0	0	3.33	
17	ЗУП (обществознание)		0	0		

Рисунок 1.2 – Сайт “Дневник.ру”

Выделим также преимущества и недостатки данного сервиса.

Преимущества:

- огромный функционал: приложение автоматизирует большую часть учебного процесса, от просмотра оценок до внеклассных занятий и автоматизации зачисления в образовательные организации;

- приложение предоставляет различные полезные услуги широкому кругу лиц: от учителей, предоставляя удобные инструменты для онлайн-образования, до госорганов, предоставляя им различную статистику по школам, ученикам и так далее.

Недостатки:

- для того, чтобы пользоваться приложением, необходима платная подписка;
- сайт перегружен различным функционалом, при этом имеет немного устаревший интерфейс, что снижает удобство его использования для многих пользователей.

1.1.2 Анализ стека технологий

Так как клиент-серверная архитектура занимает сегодня лидирующие позиции в мире программирования, то для написания подобного рода приложений существует большое количество высокоуровневых языков программирования, таких как *Java*, *C++*, *C#*, *GO*, *Python* и другие.

Java – язык, известный своей переносимостью. Девиз *Java* – “Write once, run anywhere” (напишите один раз, запустите где угодно). Он часто используется для web-разработки, как язык, отвечающий за серверную часть приложения. Одно из основных преимуществ *Java* это то, что код, написанный на этом языке, может быть легко перенесен на другую операционную систему, поскольку он компилируется в промежуточный байт-код, который обрабатывается *JVM* (java virtual machine). Также этот язык объектно-ориентированный и имеет надежную систему управления памятью и сборщик мусора.

C++ – это язык программирования, способный решать самый широкий круг задач. На этом языке можно написать что угодно, от разработки драйверов и работой с различными платами до клиент-серверных приложений и компьютерных игр. Говоря про *C++* в контексте клиент-серверных приложений, можно заметить его высокую эффективность, которая возможна благодаря прямому управлению памятью в этом языке, поэтому данный язык использует для высокопроизводительных приложений, таких как высоконагруженный веб-сервисы и базы данных, однако как раз из-за большого контроля над системными ресурсами разработка на *C++* крайне сложна, в ней гораздо больше вероятность какой-либо ошибки, а

				ПГУ1.090301.01.001 ПЗ	Лист
					10
Изм.	Лист	№ докум	Подпись Дата		

также разработка обычно ведется явно дольше по сравнению с другими языками программирования.

C# – это объектно-ориентированный язык программирования от компании *Microsoft*. Данный язык, как и упомянутая ранее *Java*, поддерживает сборку мусора, что упрощает написание кода на нем и уменьшает риск возникновения утечки памяти. Также язык интегрирован в систему *Microsoft.NET*, ориентированную в том числе и на *Web* – разработку, что позволяет эффективно его использовать в данной сфере.

Golang – язык, созданный в 2007 году в компании *Google*, призванный решить проблемы, связанные с разработкой сетевых и серверных приложений. Язык легок и эффективен, быстро работает и компилируется, а его синтаксис довольно лаконичен, что позволяет его быстро освоить и писать на нем различные программы. Также язык имеет довольно широкий набор обширных библиотек для работы с сетью, которые берут на себя часть задач по взаимодействию с пользователем. Язык набирает достаточно сильную популярность с каждым годом за счет перечисленных выше конкурентных преимуществ.

Python – интерпретируемый язык программирования, созданный в 1991 году. Является одним из наиболее популярных языков программирования. Свою популярность он приобрел за счет простоты в создании веб-приложений. Это обусловлено его лаконичным синтаксисом и удобным структурам данных, что позволяло быстро создавать серверные приложения и работать с базами данных [1].

PHP – язык был создан в 1994 году и довольно быстро обрел широкую популярность и славу языка для разработки серверных приложений, и на сегодняшний день язык все еще сохраняет лидирующие позиции в данной сфере. К преимуществам языка можно отнести то, что на нем очень просто создавать веб-приложения и обрабатывать данные на веб-стороне. *PHP* имеет

встроенную поддержку всех необходимых для сервера функций: работы с базами данных, с сетевыми протоколами, с изображениями и т. д.

NodeJs – серверный язык программирования, разработанный в 2009 году и быстро ставший популярным для создания высокоэффективных и масштабируемых сетевых приложений. Одним из основных преимуществ *Node.js* является его асинхронная модель выполнения, которая позволяет обрабатывать большое количество запросов без блокировки потоков. Это позволяет создавать масштабируемые приложения, которые могут обслуживать огромное количество пользователей. *Node.js* также имеет множество модулей и библиотек, которые облегчают создание серверных приложений. Одним из самых популярных модулей является *Express.js*, который упрощает создание веб-приложений и *API*.

Учитывая все преимущества и недостатки представленных языков программирования – был выбран язык *Python*.

В качестве фреймворка для проекта – был выбран *Django Framework*.

Django Framework – бесплатный и открытый фреймворк для веб-приложений, написанных на языке *Python* [2]. У данного фреймворка существует ряд преимуществ, делающих его приоритетнее других фреймворков на *Python*. Он имеет довольно обширное количество встроенных функций и библиотек, которые упрощают создание веб-страниц. Он обладает довольно высокой скоростью за встроенной асинхронности. Помимо этого, *Django* обладает довольно высоким уровнем защиты от различных атак: он имеет встроенную защиту от таких атак, как *XSS* и *CSRF*, что позволяет создать безопасные и надежные веб-приложения [3].

Django Framework работает по принципу модель-представление-шаблон (*MVT*). Этот принцип разделяет логику приложения на отдельные части, что позволяет упростить разработку и поддержку кода [4].

Слой модели определяет структуру данных, отвечает за работу с базой данных и определяет доступ к данным. В *Django* модель определяется

				ПГУ1.090301.01.001 ПЗ	Лист
					12
Изм.	Лист	№ докум	Подпись Дата		

классом, который наследуется от класса *Model*. Создавая модель, мы определяем поля, которые будут храниться в базе данных и типы данных для этих полей. В дальнейшем мы можем использовать модель для выполнения запросов к базе данных и получения результатов.

Слой представления отвечает за логику приложения и определяет, какие именно данные будут переданы в итоговый шаблон. Слой является промежуточным между слоем модели и слоем шаблона. В Django представление реализуется классом, наследующимся от класса *View*. В представлении определяется метод *HTTP (GET, POST, DELETE, PUT)*, который будет использоваться, обрабатываются данные, полученные из запроса, а также осуществляется передача данных в шаблон.

Слой шаблон отвечает за отображение полученных данных на странице, их визуализацию и взаимодействие с пользователем. Шаблон написан на специальном языке шаблонов *Django*, который позволяет использовать условия, циклы и другие функции для создания динамического контента. В шаблоны можно передавать примеры из представления и выполнять логические операции для отображения контента пользователю.

Взаимодействие между слоями *MVT* происходит следующим образом: при запросе страницы пользователем, *Django* связывает *URL* с соответствующим представлением, через модель выполняет запрос к базе данных и получает нужные данные. Затем *Django* передает данные в шаблон и возвращает сформированную страницу пользователю. Схематично данная архитектура представлена на рисунке 1.3. Также к еще одному из явных преимуществ данной архитектуры можно отнести то, что в этой архитектуре у нас присутствует легкая взаимозаменяемость. Так как мы привязываемся к абстрактным понятиям, а не к конкретным реализациям конкретных классов с запросами к БД, например, мы можем легко и просто манипулировать архитектурой нашей программы.

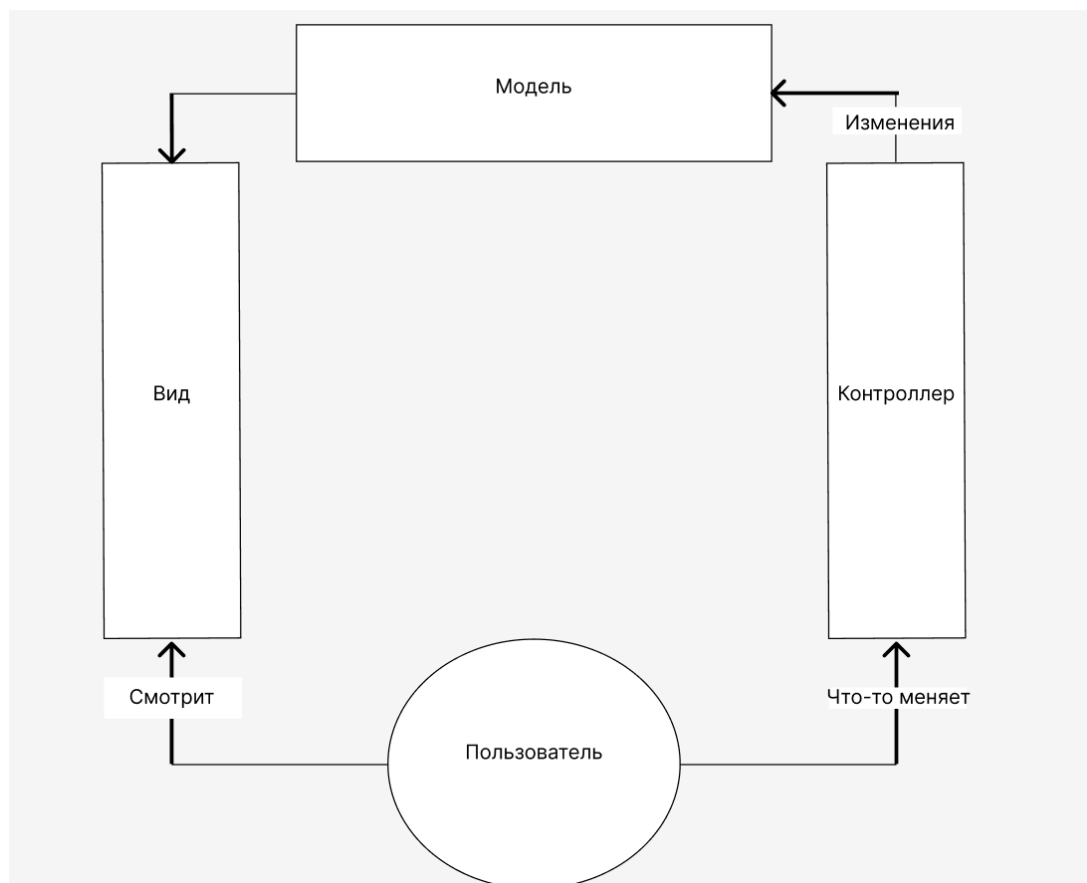


Рисунок 1.3 – Схема архитектуры *MVT*

1.1.3 Выбор СУБД

В данное время существует огромное множество систем управления базами данных. Их подразделяют на два основных вида: реляционные (*SQL*) и нереляционные (*NoSQL*). Основное отличие данных СУБД в способе хранения данных.

Выбор СУБД – это ответственный шаг для проекта. Разные СУБД обладают разным набором качеств и решают разные проблемы, поэтому их так много на сегодняшний день. Ключевые факторы, которые влияют на выбор СУБД: производительность, масштабируемость, безопасность и доступность. На сегодняшний день популярные реляционные базы данных (*SQL*), но также набирают свою популярность и нереляционные базы данных (*NoSQL*). Рассмотрим данные виды баз данных.

Реляционные базы данных – это один из видов баз данных, позволяющий организовать хранение данных в виде таблиц с установлением отношений между этими таблицами. Каждая таблица состоит из строк и столбцов, в столбцах таблицы находятся атрибуты, а строки обозначают конкретные экземпляры данных.

Связь в таких таблицах устанавливается с помощью внешних ключей – атрибут таблицы, который связывает таблицы между собой и представляет собой ссылку на значение таблицы в первичном ключе другой таблицы. Можно выстраивать различные параметры управления такими ключами, например, настроить работу с ними так, чтобы при удалении записи, на которую ссылается внешний ключ, все записи, ссылка которых ведет на эту запись, будут автоматически удалены из базы данных.

Свое название реляционные базы данных берут от математической теории отношений, а поскольку связи между таблицами в базах данных тоже называются отношениями, то базы данных, использующие такую организацию данных, быстро стали называться реляционными базами данных. На рисунке 1.4 представлен пример математической теории отношений.

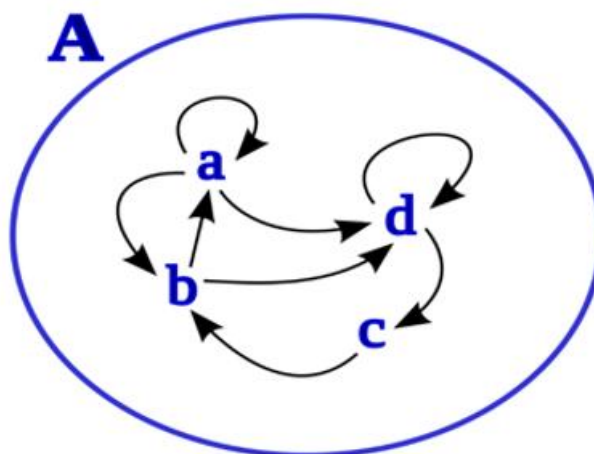


Рисунок 1.4 – Иллюстрация примера отношений над множеством А

Нереляционные базы данных – это семейство СУБД, которые не используют классическую модель организации данных в виде таблиц и связей между ними, вместо этого они предоставляют гибкую и расширяемую структуру хранения данных.

Временем, когда стали появляться первые *NoSQL* базы данных, можно считать 200-е годы. Данные базы данных были призваны решить ограничения, которые накладывают реляционные базы данных в хранении и обработки больших объемов неструктурированных данных, таких как графы, документы, ключи-значения и т.д.

Существует несколько основных типов *NoSQL* – баз данных:

- документо-ориентированные базы данных хранят неструктурированные данные в формате документов, часто используются для хранения данных веб-страниц и блогов;
- ключ-значение базы данных хранят небольшие объемы данных, используя уникальный ключ для доступа к каждой записи;
- графовые базы данных хранят данные в виде графов, позволяя легко устанавливать связи между различными элементами данных;
- колоночные базы данных организуют данные по колонкам вместо строк, что позволяет быстро получать данные как для конкретной колонки, так и для группы колонок.

На рисунке 1.5 изображен пример документоориентированной СУБД.



Рисунок 1.5 – *MongoDB* – документоориентированная СУБД

К основным преимуществам нереляционных баз данных можно отнести их гибкую структуру хранения данных, что позволяет быстро реагировать на изменения данных и динамично создавать новые схемы, высокую производительность на больших объемах данных, а также то, что они хорошо масштабируются.

Для реализации проекта была выбрана реляционная СУБД – *SQLite3*.

SQLite3 – это компактная, быстрая и универсальная СУБД, которая может использоваться как локальное хранилище данных, так и как сервер базы данных для веб-приложений.

Основные преимущества данной базы данных:

- легковесность: *SQLite3* не требует никакой специальной предварительной настройки, она может быть представлена файлом, который автоматически создается, и мы сразу сможем иметь к нему доступ, что упрощает процесс разработки и экономит время;

- безопасность: несмотря на свою простоту, *SQLite3* использует сильные алгоритмы шифрования, что гарантирует данным значительную защищенность;

- надежность: *SQLite3* предоставляет ACID-совместимые транзакции, что обеспечивает целостность базы данных и защищает из от ошибок данных;

- поддержка основных типов данных: *SQLite3* поддерживает все основные типы данных, такие как числа, текст, бинарные данные, дата и время, что делает ее удобной в использовании для всех языков программирования;

- бесплатность: *SQLite3* является бесплатной *Open source* СУБД, что делает ее доступной в использовании для любых проектов.

1.1.4 Выбор среды разработки

Для быстрой и комфортной разработки на языке Python существует множество различных сред разработки, от самых простых и скучных по

				ПГУ1.090301.01.001 ПЗ	Лист
					17
Изм.	Лист	№ докум	Подпись Дата		

функционалу до современных мощных средств [5]. К наиболее популярным средам разработки на данном языке можно отнести *Pycharm*, *VSCode*, *Spyder*, *PyDev* и *Jupyter Notebook*. Рассмотрим каждую из них немного подробнее.

Pycharm – это бесплатная универсальная и мощная *IDE* от компании *JetBrains*, которая содержит в себе довольно значительный набор функций, которые упрощают разработку, отладку, тестирование и поддержку Python-приложений. У *Pycharm* есть огромное количество конкурентных преимуществ: автодополнение, проверка типов, встроенный отладчик, инструменты анализа кода и т.д. На сегодняшний день эта среда разработки считается стандартом среди разработчиков на языке *Python*.

VSCode – бесплатный редактор кода с открытым исходным кодом, который поддерживает работу с различными языками программирования, в том числе и с *Python*. *VSCode* может быть удобным образом настроен для работы с *Python* путем установки расширений из официального хранилища. Устанавливая расширения, разработчик получает множество полезных функций, таких как подсветку синтаксиса, отображение ошибок и предупреждений, встроенный отладчик и тестирование. Основные преимущества редактора раскрываются в полной мере только после установки дополнительных расширений.

Spyder – это *IDE* для языка *Python*, которая была разработана на базе библиотеки *QT*. К основным преимуществам можно отнести просто и интуитивно понятный интерфейс, наличие графической оболочки и широкие возможности для анализа данных. К недостаткам относятся склонность данной среды к замедлению скорости работы при открытии большого числа модулей и вкладок, относительная ограниченность функционала по сравнению с другими *IDE*, т.к. *Spyder* все же ориентирован больше на работу с научными вычислениями и анализом данных, а не на *Web*-разработку или другие ниши.

PyDev – это не совсем полноценная *IDE*, а по сути бесплатное дополнение к таким *IDE* как *Eclipse*, *Aptana Studio* и другим. *PyDev* обладает рядом преимуществ, такими как открытый исходный код, мощные функции автодополнения для библиотек и фреймворков *Python*, графический интерфейс пользователя, поддержка множественного оконного режима, встроенный отладчик и легкость использования. К его недостаткам относится необходимость иметь другие *IDE* для полноценной работы, недружелюбный графический интерфейс для новых пользователей, а также повышенная требовательность к ресурсам и медленная скорость работы из-за этого, особенно на старых машинах.

Jupyter Notebook – это интерактивная среда разработки, которая представляет из себя веб-приложение, которое запускается в браузере и базируется на языках *Python*, *R* и *Julia*. К основным преимуществам ноутбука можно отнести возможность сразу в одном месте писать код, проводить анализ данных и создавать документацию, а также удобные инструменты для визуализации и обработки данных.

После анализа различных сред разработки выбор был сделан в пользу *Pycharm*.

1.2 Проектирование и разработка

Данное приложение будет представлять из себя классическое клиент-серверное приложение. Сервер будет отвечать за обработку запросов на соответствующих эндпоинтах, реализовывать бизнес-логику, а также работать с данными в виде моделей и производить соответствующие манипуляции с базой данных. Клиентская часть будет реализована на *HTML* и *CSS*, но *HTML* шаблоны будут в данном случае частью *MVT* архитектуры фреймворка *Django* [6]. Сам язык *HTML* с помощью тегов будет формировать основной костяк страницы, а затем на этих страницах будет производиться

				ПГУ1.090301.01.001 ПЗ	Лист
					19
Изм.	Лист	№ докум	Подпись Дата		

динамическая манипуляция данными с помощью специальных тегов шаблонов *Django*.

Пользователь будет обладать возможностями поиска определенной кинокартины по конкретным критериям, получения информации о фильме, а также просмотра или сохранения выбранного кинофильма.

В выбранном сервисе будет две группы пользователей: ученики и учителя. Ученики смогут просматривать свои оценки, получать напоминания от учителей, просматривать домашние задания, которые задали учителя, просматривать список учителей, а также иметь возможность обновить свой профиль и поменять пароль. Учителя имеют возможность просматривать список всех учеников, добавлять учеников в свой класс, загружать напоминания, загружать домашние задания, а также выставлять оценки.

Перед началом работы сервера производится инициализация всех модулей и компонентов, а также подключение к базе данных (и ее создание, при отсутствии).

1.2.1 Реализация серверной части

Исходная серверная часть была создана командой «*django-admin startproject studentmanager*». Для приложения были определены базовые настройки, представленные в файле *settings.py*, задающие список необходимых для работы приложения модулей, основные директории для сопутствующих файлов, путь до файла с базой данных *SQLite3* и так далее [7]. Часть из базовых настроек приложения представлены на рисунке 1.6.

```
import os

# Build paths inside the project like this: os.path.join(BASE_DIR, ...)
BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
TEMPLATE_DIR = os.path.join(BASE_DIR, 'template')
STATIC_DIR = os.path.join(BASE_DIR, 'static')
MEDIA_DIR = os.path.join(BASE_DIR, 'media')
```

Рисунок 1.6 – Базовые настройки приложения

				ПГУ1.090301.01.001 ПЗ	Лист
					20
Изм.	Лист	№ докум	Подпись Дата		

Затем был сформирован список эндпоинтов для данного приложения, обращаясь к которым будут вызываться соответствующие функции и будет происходить валидация данных и их сохранение в базу. Частичный список эндпоинтов приложения представлен на рисунке 1.7.

```
urlpatterns = [
    path('signup/', views.SignUp, name="signup"),
    path('signup/student_signup/', views.StudentSignUp, name="StudentSignUp"),
    path('signup/teacher_signup/', views.TeacherSignUp, name="TeacherSignUp"),
    path('login/', views.user_login, name="login"),
    path('logout/', views.user_logout, name="logout"),
    path('student/<int:pk>', views.StudentDetailView.as_view(), name="student_detail"),
    path('teacher/<int:pk>', views.TeacherDetailView.as_view(), name="teacher_detail"),
    path('update/student/<int:pk>', views.StudentUpdateView, name="student_update"),
    path('update/teacher/<int:pk>', views.TeacherUpdateView, name="teacher_update"),
    path('student/<int:pk>/enter_marks', views.add_marks, name="enter_marks"),
    path('student/<int:pk>/marks_list', views.student_marks_list, name="student_marks_list"),
    path('marks/<int:pk>/update', views.update_marks, name="update_marks"),
    path('student/<int:pk>/add', views.add_student.as_view(), name="add_student"),
    path('student_added/', views.student_added, name="student_added"),
    path('students_list/', views.students_list, name="students_list"),
    path('teachers_list/', views.teachers_list, name="teachers_list"),
    path('teacher/class_students_list', views.class_students_list, name="class_student_list"),
    path('student/<int:pk>/all_marks', views.StudentAllMarksList.as_view(), name="all_marks_list"),
    path('student/<int:pk>/message', views.write_message, name="write_message"),
    path('teacher/<int:pk>/messages_list', views.messages_list, name="messages_list"),
    path('teacher/write_notice', views.add_notice, name="write_notice"),
    path('student/<int:pk>/class_notice', views.class_notice, name="class_notice"),
]
```

Рисунок 1.7 – Список эндпоинтов приложения для вызова функций

Для обработки соответствующих эндпоинтов был создан файл *forms.py* со списком соответствующих функций. Стандартный цикл обработки пришедшего от клиента запроса выглядит следующим образом: мы анализируем, *POST* или *GET* запрос был прислан от пользователя (в данной работе рассматриваются только два данных стандартных вида запроса), и, в зависимости от того, какой именно запрос нам пришел от клиента, мы либо возвращаем стандартное представление, либо сохраняем данные в базу [8]. Стандартный пример такой функции представлен на рисунке 1.8. Можно отметить, что в *Django* в функции передается также параметр *request*, через который мы можем работать с пользовательскими данными, пришедшими с клиентской части.

```

## login view.
1 usage  Shubh0405
def user_login(request):
    if request.method == "POST":
        username = request.POST.get('username')
        password = request.POST.get('password')

        user = authenticate(username=username, password=password)

        if user:
            if user.is_active:
                login(request, user)
                return HttpResponseRedirect(reverse('home'))

            else:
                return HttpResponse("Account not active")

        else:
            messages.error(request, "Invalid Details")
            return redirect('classroom:login')
    else:
        return render(request, 'classroom/login.html', {})

```

Рисунок 1.8 – Пример функции для обработки запроса, поступившего от пользователя

В случае, если данные не представлены, будет возвращена ошибка 404, для этого используется специальная функция *get_object_or_404*, входящая в пакет *django.shortcuts* [9].

В целях обеспечения защиты некоторых частей сайта от непреднамеренного доступа, часть функций защищена специальной аннотацией *@login_required*. Эти функции могут вызываться только пользователем, прошедшим процесс аутентификации и залогиненным с помощью функций *authenticate()* и *login()* соответственно. Пример использования такой функции приведен на рисунке 1.9.

```

@login_required
def TeacherUpdateView(request, pk):
    profile_updated = False
    teacher = get_object_or_404(models.Teacher, pk=pk)
    if request.method == "POST":
        form = TeacherProfileUpdateForm(request.POST, instance=teacher)
        if form.is_valid():
            profile = form.save(commit=False)
            if 'teacher_profile_pic' in request.FILES:
                profile.teacher_profile_pic = request.FILES['teacher_profile_pic']
            profile.save()
            profile_updated = True
    else:
        form = TeacherProfileUpdateForm(request.POST or None, instance=teacher)
    return render(request, 'classroom/teacher_update_page.html', {'profile_updated': profile_updated, 'form': form})

```

Рисунок 1.9 – Пример функции, защищенной от пользователя с помощью аннотации `@login_required`

Также, для работы с основными формами сайта, такими как форма с данными ученика, учителя, форма для добавления оценок и т.д. был создан специальный модуль `forms.py`. На основе пакета `forms`, входящего в фреймворк `Django`, был создан набор классов с основными полями для данных форм. Когда происходит запросы на эндпоинты, мы легко можем либо отдавать на клиентскую часть объект на основе соответствующего класса, либо с помощью метода `isValid()` осуществлять валидацию полей на соответствие тем параметрам, которые мы указали при создании модели. Пример такого класса представлен на рисунке 1.10.

```

## User Login Form (Applied in both student and teacher login)
5 usages  ▴ Shubh0405
class UserForm(UserCreationForm):
    ▴ Shubh0405
    class Meta():
        model = User
        fields = ['username', 'password1', 'password2']
        widgets = {
            'username': forms.TextInput(attrs={'class': 'answer'}),
            'password1': forms.PasswordInput(attrs={'class': 'answer'}),
            'password2': forms.PasswordInput(attrs={'class': 'answer'}),
        }

```

Рисунок 1.10 – Класс для создания и валидации формы учащегося

1.2.2 Реализация БД

Object-Relational Mapping (ORM) - это способ связывания баз данных и объектов в *Python*-коде. В фреймворке *Django* имеется встроенный *ORM*, который позволяет управлять записями в базе данных. Также, модели, заданные в *Python* коде, определяют и отношения между самими таблицами [10].

В разрабатываемом приложении явно выделяются 5 групп моделей в базы данных, а также связи между ними, реализованные по принципу “Многие – ко – многим”: ученики, преподаватели, оценки учеников, домашние задания и напоминания, создаваемые преподавателями для учеников, сообщения учителя от учеников.

Поля группы “Преподаватели” и “Студенты” представлены в таблице 1.1:

Таблица 1.1 – Описание полей таблицы “Classroom_teacher”

№	Имя поля	Тип	Примечание
1	teacher_id	int	Первичный ключ, обязательный
2	name	varchar(128)	Необязательный
3	subject_name	varchar(128)	Необязательный
4	email	varchar(128)	Необязательный
5	phone	int	Необязательный
6	teacher_profile_pic	blob	Необязательный

Таблица 1.2 – Описание полей таблицы “Classroom_student”

№	Имя поля	Тип	Примечание
1	student_id	int	Первичный ключ, обязательный
2	name	varchar(128)	Необязательный
3	roll_no	int	Необязательный
4	email	varchar(128)	Необязательный
5	phone	int	Необязательный

6	student_profile_pic	blob	Необязательный
---	---------------------	------	----------------

UML - диаграмма связи между этими таблицами представлена на рисунке 1.11.



Рисунок 1.11 – *UML*-диаграмма классов учителей и учеников и связь между ними (многие ко многим)

Поля модели “Оценки учеников” показаны в таблице 1.3:

Таблица 1.3 – Описание полей таблицы “Classroom_teacher”

№	Имя поля	Тип	Примечание
1	marks_id	int	Первичный ключ, обязательный
2	teached_id		Внешний ключ
3	student_id		Внешний ключ
4	subject_name	varchar(128)	Обязательный
5	marks_obtained	int	Обязательный
6	maximum_marks	int	Обязательный

UML - диаграмма связи этого класса с классом учеников и преподавателей представлена на рисунке 1.12.

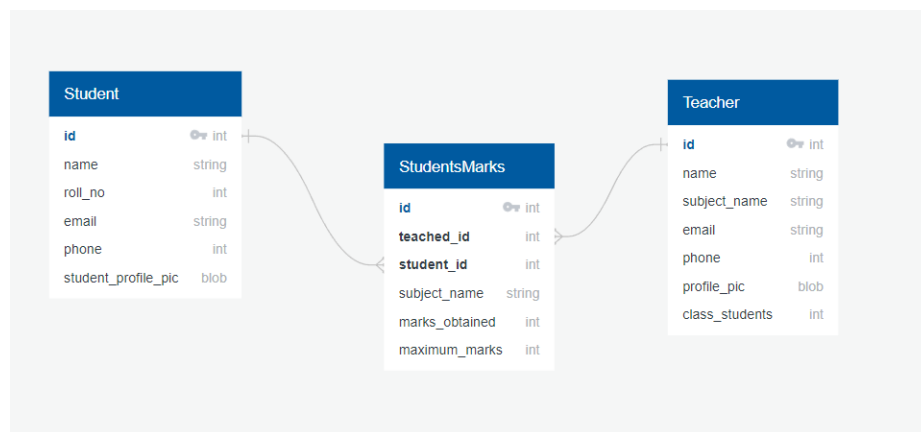


Рисунок 1.12 – UML-диаграмма класса оценок

В таблице 1.4 и 1.5 представлены поля моделей “Уведомления” и “Домашние задания”:

Таблица 1.4 – Описание полей таблицы “class_notice”

№	Имя поля	Тип	Примечание
1	id	int	Первичный ключ, обязательный
2	teached_id		Внешний ключ
3	student_id		Внешний ключ
4	created_at	datetime	Обязательный
5	message	varchar(128)	Обязательный
6	message_html	varchar(128)	Обязательный

Таблица 1.5 – Описание полей таблицы “submit_assignment”

№	Имя поля	Тип	Примечание
1	id	int	Первичный ключ, обязательный
2	teached_id		Внешний ключ
3	student_id		Внешний ключ
4	created_at	datetime	Обязательный
5	submit	blob	Обязательный

UML-диаграмма классов уведомлений и домашних заданий представлены на рисунке 1.13.

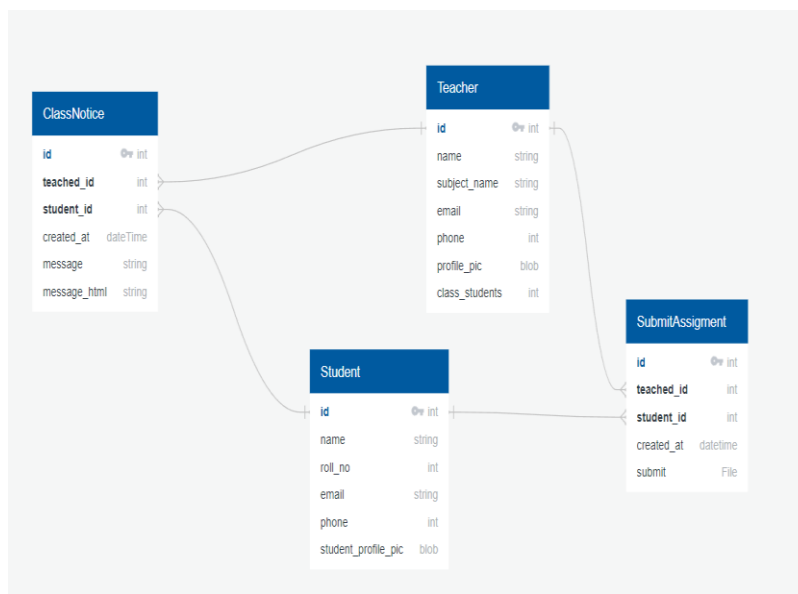


Рисунок 1.13 – *UML*-диаграмма классов уведомлений и домашних заданий

Последнюю из основных моделей в приложении составляет модель “message_to_teacher”. Данная модель отображена в соответствующей таблице: таблица 1.6.

Таблица 1.6 – Описание полей таблицы “message_to_teacher ”

№	Имя поля	Тип	Примечание
1	id	int	Первичный ключ, обязательный
2	teached_id		Внешний ключ
3	student_id		Внешний ключ
4	created_at	datetime	Обязательный
5	message	varchar(128)	Обязательный
6	message_html	varchar(128)	Обязательный

UML-диаграмма связи класса сообщений учителям с классами учеников и учителей представлена на рисунке 1.14.

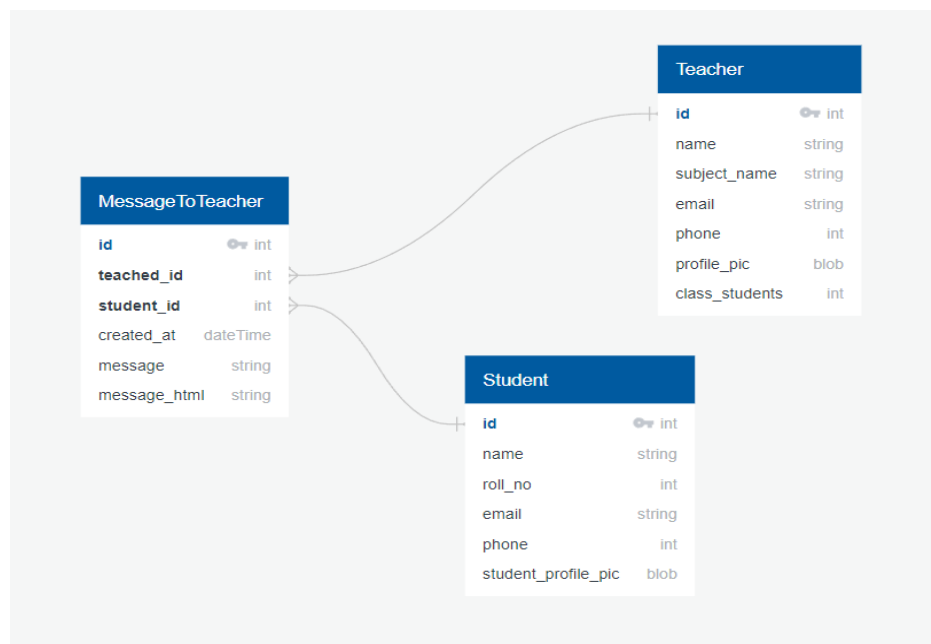


Рисунок 1.14 – *UML*-диаграмма класса сообщений учителям

1.2.3 Разработка клиентской части.

Клиентская часть данного приложения реализована как набор шаблонов *Django* фреймворка, представляющих собой стандартную *HTML* – верстку, дополненную шаблонизатором *Django* для простого расширения страниц, а также для того, чтобы отображать на клиентской части модели *Django* и использовать специальные *Django*-теги, позволяющие реализовать с помощью *Python* кода логику на стороне клиента. За внешний вид страниц отвечают подключаемые к *.html* – файлам *css* – файлы. Также в качестве фреймворка для верстки используется популярный фреймворк *Bootstrap*, позволяющий подключать в качестве удаленных модулей к страницам готовые *html* элементы. Базовой страницей, от которой наследуются, то есть берут основные базовые элементы, такие как шапка сайта, является страница *base.html*, представленная на рисунке 1.15.

```

<!DOCTYPE html>
{% load static %}
<html lang="en" dir="ltr">
<head>
  <meta charset="utf-8">
  <title>Student manger</title>
  <!-- Latest compiled and minified CSS -->
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css">

  <link rel="stylesheet" href="{% static 'css/base.css' %}">
  <link rel="stylesheet" href="{% static 'css/searchbar.css' %}">
  <link rel="stylesheet" href="{% static 'css/newbase.css' %}">
  <link rel="stylesheet" href="https://pro.fontawesome.com/releases/v5.10.0/css/all.css">
  <link href="https://fonts.googleapis.com/css2?family=Source+Sans+Pro&display=swap" rel="stylesheet">
  <link href="https://fonts.googleapis.com/css2?family=Permanent+Marker&family=Source+Sans+Pro&display=swap" rel="stylesheet">

  {% load bootstrap4 %}
  {% bootstrap_css %}
  {% bootstrap_javascript jquery='full' %}
</head>

```

Рисунок 1.16 – Код страницы *base.html*

Для того, чтобы “унаследовать” с базовой страницы часть повторяющихся *html* - стилей, используется ключевое для *Django* словосочетание `{%extends “classroom/base.html”%}`, где в кавычках указывается название страницы, от которой мы и хотим взять определенные *html* – элементы, в данном случае – *base.html*. На тех страницах, где мы хотим использовать библиотеку *bootstrap*, мы должны указать в качестве класса *html*-тега классы, предоставляемые нам этой библиотекой, например, *class = “col-md-3”*. Пример такого использования показан на рисунке 1.17.

```

<div class="col-md-11">

```

Рисунок 1.17 – Блочный тег *div* с классом *col-md-11* из библиотеки *bootstrap*

1.3 Анализ результатов

Была разработана система для контроля успеваемости на языке *Java* с применением *Spring Framework*.

Было разработано клиент-серверное приложение: система контроля успеваемости учащихся.

Учителю предоставляется следующий набор возможностей:

- загружать домашние задания;
- просматривать список заданных домашних заданий;
- просматривать список учеников;
- добавлять учеников к себе в класс;
- ставить оценки ученикам;
- писать заметки ученикам;
- просматривать свой профиль.

UML – диаграмма вариантов использования приложения учителем представлена на рисунке 1.18.



Рисунок 1.18 - UML-диаграмма вариантов использования приложения учителем

Ученику предоставляется следующий набор возможностей:

- просматривать свои оценки;
- просматривать список заданных домашних заданий;
- просматривать список напоминаний;
- просматривать список учителей;
- просматривать свой профиль.

UML – диаграмма вариантов использования приложения учеником представлена на рисунке 1.19.

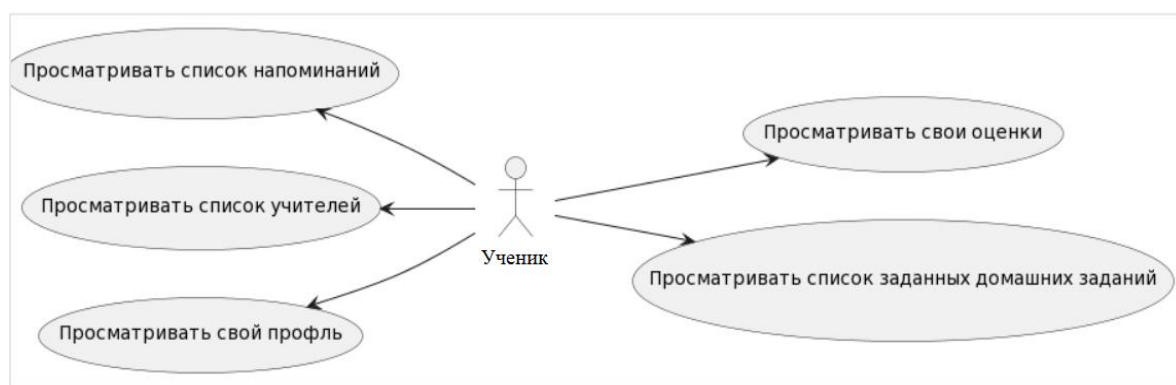


Рисунок 1.19 - *UML*-диаграмма вариантов использования приложения учеником

1.3.1 Тестирование приложения

Было проведено ручное тестирование приложения. Для начала работы следует создать в системе ученика и учителя. На рисунке 1.20 предоставлен режим регистрации в приложении. Создадим последовательно оба аккаунта.

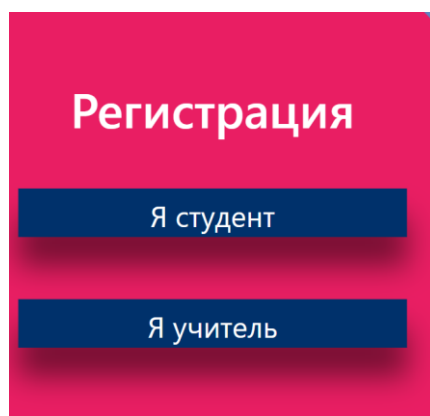


Рисунок 1.20 – Создадим в режиме регистрации аккаунт ученика и учителя

Далее зайдем под аккаунтом учителя и найдем в список ученика, которого мы до этого создали – *anton*. Результат данного пункта можем видеть на рисунке 1.21.

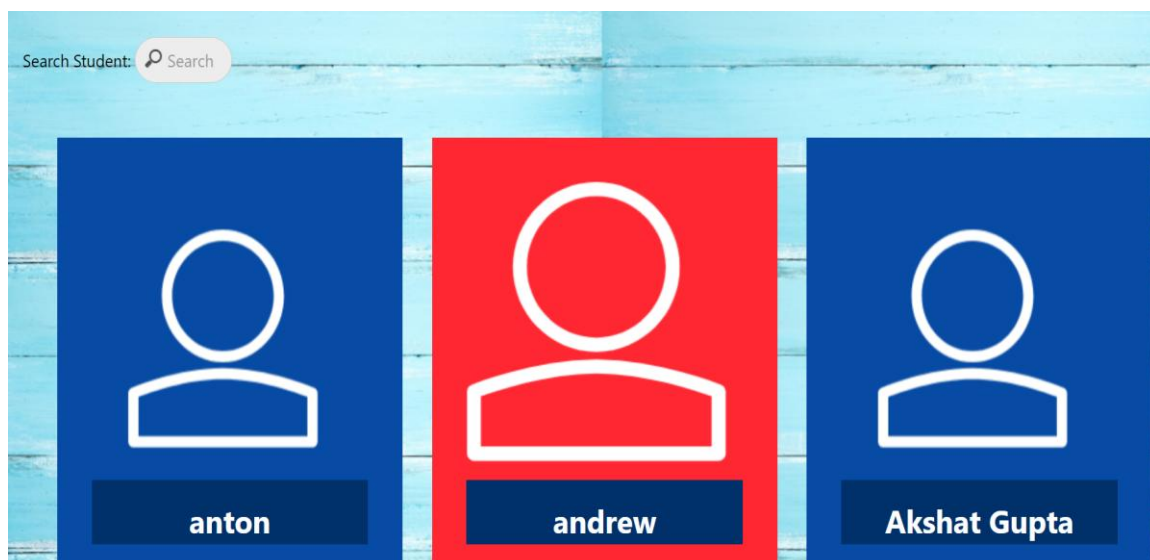


Рисунок 1.21 – Добавим себе созданного ученика по имени *anton*

Убедимся, что ученик появился в журнале класса. Результат отображен на рисунке 1.22.



Рисунок 1.22 – Ученик *anton* добавился к нам в классный журнал

Добавим данному ученику оценку 4 за предмет математика и убедимся, что оценка действительно успешно была добавлена. Результат добавления оценки отображен на рисунке 1.23.

Оценки anton				
№	Тема	Полученные оценки	Максимальный балл	
1	math	4	4	Обновить
Добавить оценки				

Рисунок 1.23 – Ученику *anton* была успешна выставлена оценка 4 по предмету математика

Теперь загрузим домашнее задание. Проверим, что домашнее задание отобразилось в списке всех домашних заданий. Результат работы данного шага отображен на рисунке 1.24.

#	Задача	Дата загрузки	Скачать	Обновить	Удалить
1	Решит пример на 14 странице, номер примера 10	June 21, 2023, 7:38 a.m.	Скачать	Обновить	Удалить

Рисунок 1.24 – Домашнее задание было успешно загружено

Теперь пошлем уведомление ученику, зайдём под его профилем, и проверим, что все пункты также успешно отображаются на стороне ученика.

На рисунке 1.25 мы можем увидеть, что оценка, выставленная преподавателем успешно отображается у ученика.

№	Предмет	Имя учителя	Полученные оценки	Максимальный балл
1	math	denis	4	4

Рисунок 1.25 – Оценка, выставленная учителем, отображается у ученика

На рисунке 1.26 видим, что заметка, написанная учителем в личном кабинете, отобразилась у ученика.



Рисунок 1.26 – Заметка, написанная учителем ученику, отобразилась у него

Последний шаг – убеждаемся, что домашнее задание успешно загружено на сервер и также отображается у ученика. Результат проверки отображен на рисунке 1.27. Таким образом, был успешно протестирован весь основной функционал приложения, который заявлялся в целях ее работы при начале ее написания. В системе контроля успеваемости учащихся успешно и без ошибок можно работать сразу под двумя пользователя: как под администратором, так и под пользователем.

#	Название задания	Загрузил	Дата загрузки	Скачать	Статус
1	Решит пример на 14 странице, номер примера 10	denis	June 21, 2023, 7:38 a.m.	Скачать	Отправить

Рисунок 1.27 – Домашнее задание успешно загружено и отображается у ученика

1.3.2 Установка и проверка работоспособности *Python*

Для того, чтобы мы могли писать *web* – приложения для языка *Python*, мы должны предварительно установить его интерпретатор. Сделать это для *Windows 11 x64* мы можем на его официальном сайте.

Установка соответствующего ПО состоит из крайне простых шагов:
перейти на официальный сайт и загрузить установщик себе на компьютер;

просмотреть настройки установки и выбрать подходящие для личного пользования, нажимая кнопку *Next* чтобы переходить к следующему шагу установки;

проверить установку *Python*, просмотрев его версию с помощью команды `python --version`.

На рисунке 1.28 представлен второй шаг инсталляции *Python* на ПК.

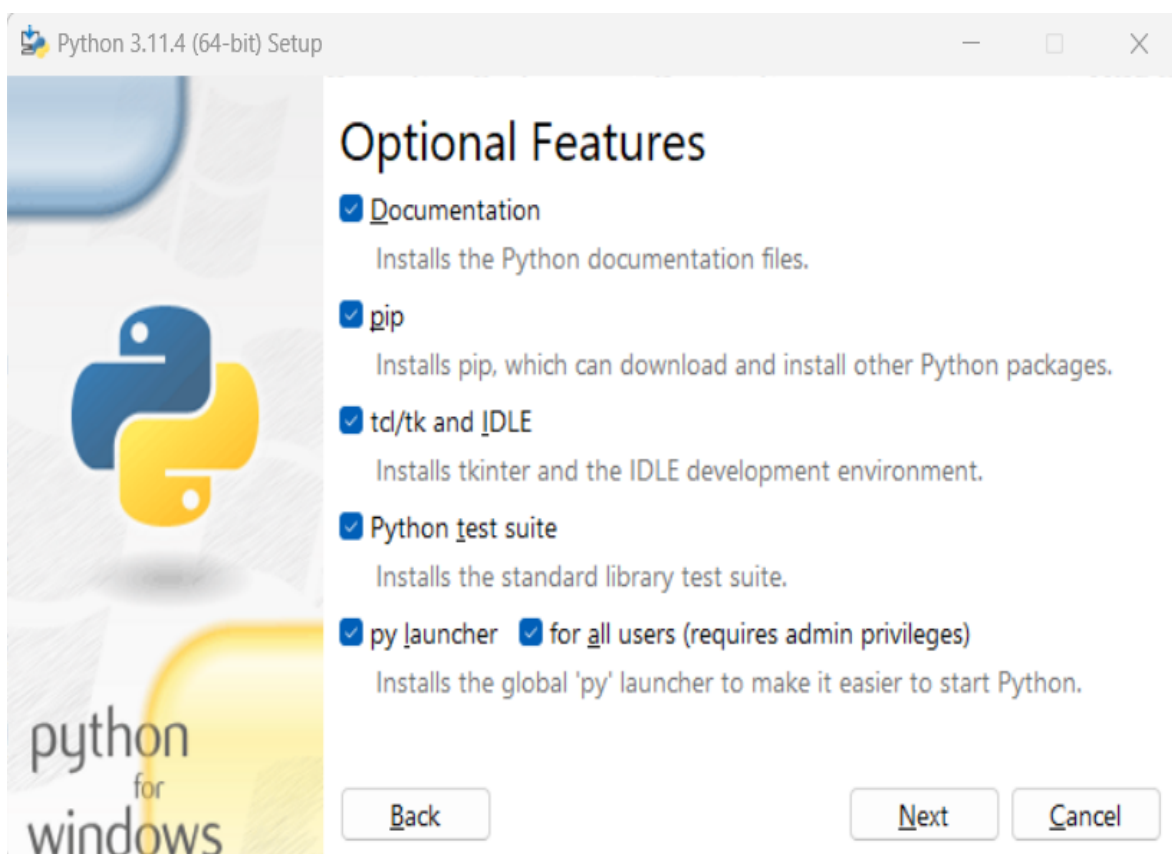
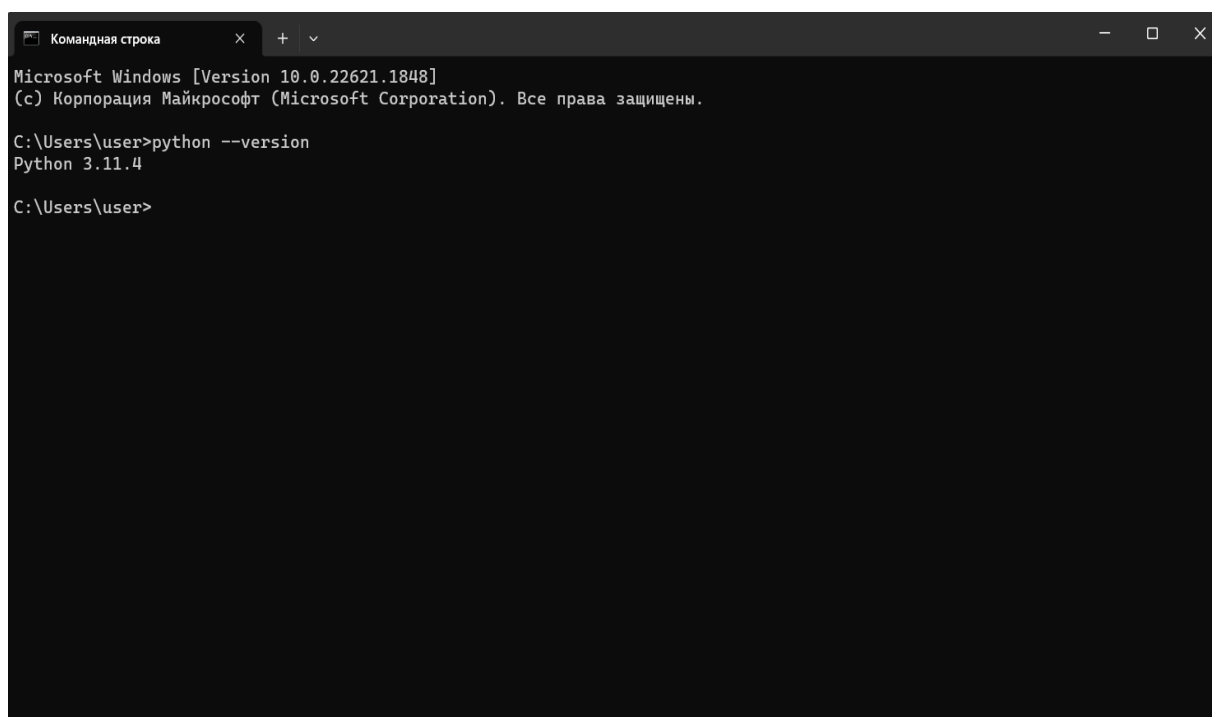


Рисунок 1.28 – Инсталляция *Python*

				ПГУ1.090301.01.001 ПЗ	Лист
					35
Изм.	Лист	№ докум	Подпись Дата		

На рисунке 1.29 проверяется успешность установки *Python* на ПК.



```
Командная строка
Microsoft Windows [Version 10.0.22621.1848]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

C:\Users\user>python --version
Python 3.11.4

C:\Users\user>
```

Рисунок 1.29 – *Python* успешно установлен на компьютер

1.3.3 – Установка и проверка работоспособности DBrowser

Для того, чтобы работать с СУБД SQLite 3 существует open source приложение, позволяющее в визуальном редакторе легко создавать, просматривать и удалять таблицы.

Данное ПО называется DB Brower for SQLite, и его также можно без каких – либо сложностей загрузить с главного сайта. Доступно приложение как для 64 – битной, так и для 32 – битной версии Windows. Установка приложения стандартна и позволяет опционально выбрать некоторые настройки приложения. На рисунке 1.30 представлено меню установщика приложения.

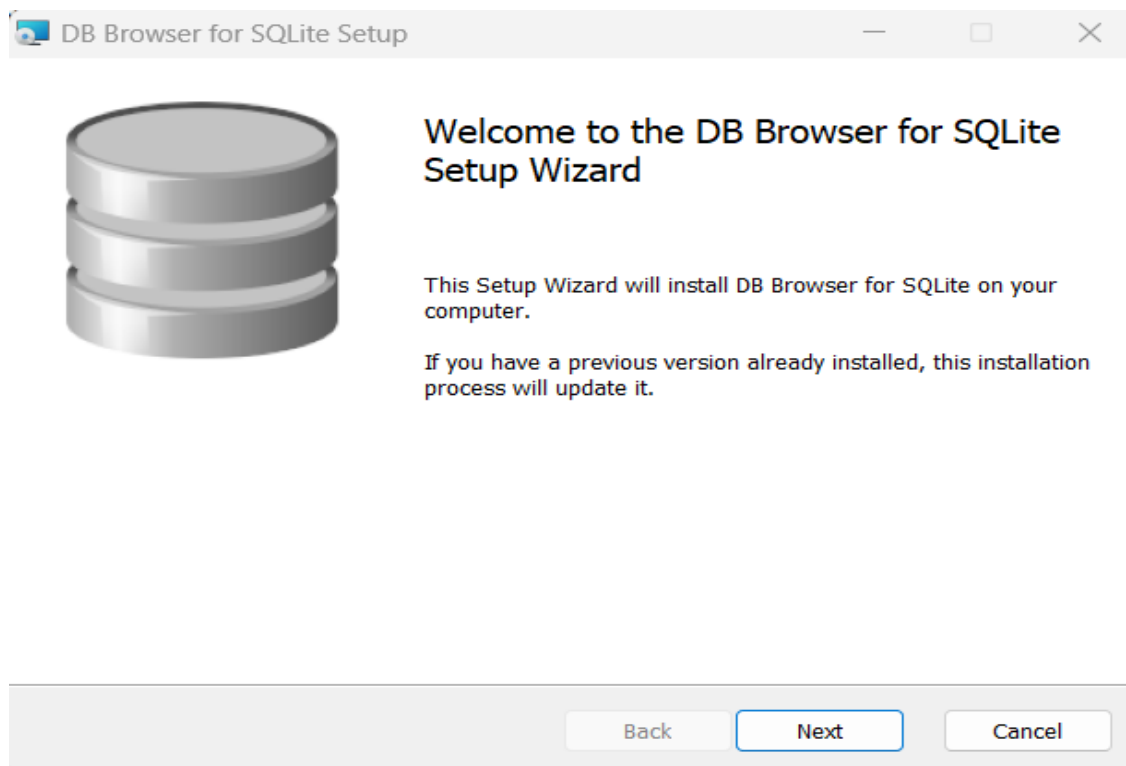


Рисунок 1. 30 – Установщик приложения

После его установки мы можем просматривать базу данных SQLite3, хранящуюся в файле db.sqlite3. На рисунке 1.31 изображена одна из таблиц базы данных, созданного для данного приложения.

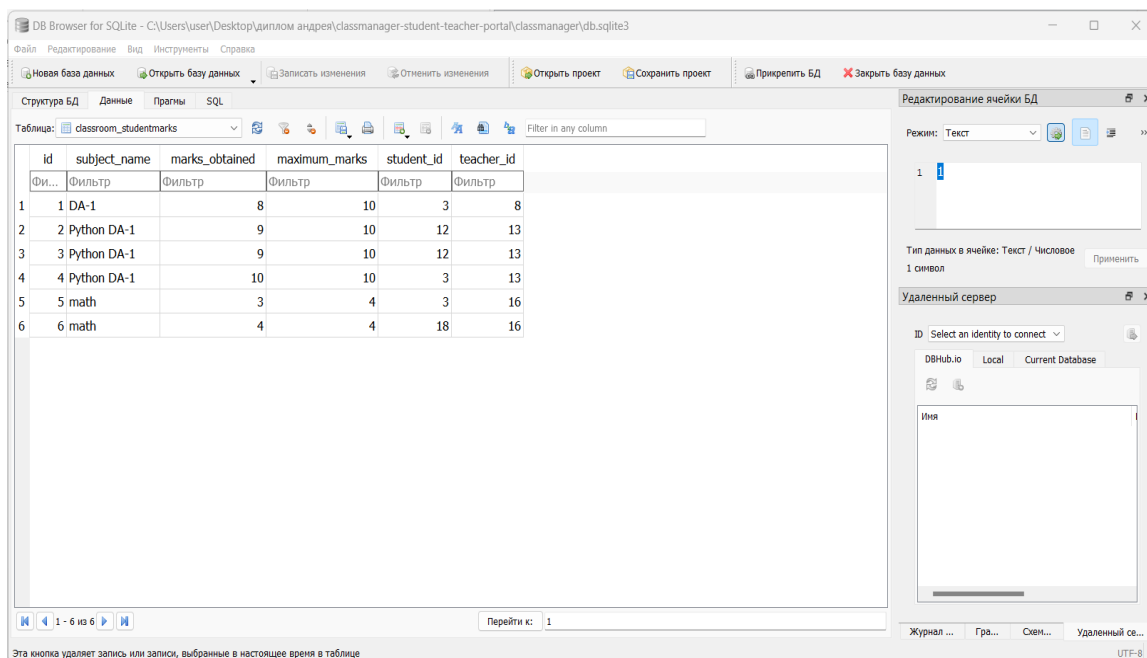


Рисунок 1.31 – Таблица в DB Browser for SQLite

2 Охрана труда

Для обеспечения безопасности жизнедеятельности и охраны труда пользователей ПЭВМ, занимающихся разработкой и эксплуатацией программного обеспечения, необходимо соблюдение правил и инструкций по безопасности на рабочем месте. Они включают в себя правила по использованию необходимой защитной экипировки, прохождение медицинских осмотров перед началом работы и регулярно в дальнейшем, а также обучение сотрудников навыкам безопасного поведения на рабочем месте.

Однако, простое соблюдение правил и инструкций может быть недостаточно для создания безопасного и здорового рабочего места. Необходима правильная организация и эргономика рабочего места, которые включают в себя установку нужной мебели, инструментов, электроники и других предметов, которые способствуют благополучию сотрудников. Кроме того, необходимо проводить регулярные проверки оборудования на безопасность и проводить ремонт при необходимости.

Также необходимо систематически анализировать и контролировать производственные процессы на предмет выявления возможных рисков и их устранения. Например, может быть необходимо обеспечить систему вентиляции для удаления вредных веществ, используемых при работе с программным обеспечением. Эти процессы должны соответствовать требованиям и стандартам безопасности.

Другими важными аспектами охраны труда являются проведение регулярных тренингов для обучения сотрудников навыкам безопасного поведения на рабочем месте и сообщение администрации об осложнениях состояния здоровья, связанных с работой.

Реализация этих мер поможет обеспечить безопасное и здоровое рабочее место, а также повысить производительность и качество производства.

				ПГУ1.090301.01.001 ПЗ	Лист
					38
Изм.	Лист	№ докум	Подпись Дата		

2.1 Характеристики потенциальных опасностей и вредностей, которые могут возникнуть при работе с ЭВМ

Также следует обратить внимание на освещение на рабочем месте, правильную настройку монитора и использование специальных очков при необходимости. Важно также следить за гигиеной рук и чистотой клавиатуры и мыши.

Кроме того, работа на компьютере может привести к снижению уровня умственной активности и концентрации, особенно при длительных периодах работы без перерывов. Для снижения риска возникновения этих проблем, необходимо проводить регулярные перерывы от работы на компьютере, включая физические упражнения и расслабляющие практики, которые помогут сбалансировать умственную нагрузку.

Работа на компьютере также может повлечь за собой повышенный риск развития синдрома карпального тоннеля, который проявляется болями и онемением в руке, вызванном повреждением среднего нерва в запястье. Для снижения риска возникновения этого синдрома необходимо использовать эргономические устройства, такие как специальные клавиатуры и мыши, а также следить за правильной позой тела при работе на компьютере.

В целом, работа на компьютере может повлечь за собой много различных рисков и проблем со здоровьем. Однако, если вы будете следить за своим здоровьем, соблюдать правильные рабочие привычки и использовать средства защиты, то риск возникновения этих проблем будет существенно снижен. Работа на компьютере может быть очень эффективной и продуктивной, если вы умеете балансировать ее с другими аспектами своей жизни и заботиться о своем физическом и психическом здоровье.

				ПГУ1.090301.01.001 ПЗ	Лист
					39
Изм.	Лист	№ докум	Подпись Дата		

2.2 Санитарно-гигиеническая характеристика производства

В сфере производства программного обеспечения (ПО) существует ряд санитарно-гигиенических норм, которые необходимо соблюдать, чтобы обеспечить безопасность и здоровье работников, а также предотвратить возникновение негативных последствий для окружающей среды.

Одним из основных аспектов санитарно-гигиенической характеристики производства ПО является обеспечение комфортных условий труда для сотрудников. Это включает в себя правильную организацию рабочих мест, установку эргономичной мебели и оборудования, а также учет потребностей в освещении и микроклимате.

Кроме того, необходимо соблюдать правильные санитарно-гигиенические условия работы, такие как проветривание помещений, регулярная уборка и дезинфекция рабочих мест, а также определение правил по утилизации продуктов питания и мусора.

Еще одним важным фактором является электробезопасность. Работа на компьютере связана с применением электроэнергии, поэтому необходимо следить за соблюдением электробезопасности и правильным подключением оборудования.

Также важно учитывать возможный вред для окружающей среды при разработке ПО, который может возникнуть при использовании определенных веществ и материалов, которые могут нанести вред окружающей среде. Поэтому необходимо соблюдать санитарно-гигиенические нормы и правила экологической защиты при выборе и использовании различных веществ и материалов.

В целом, санитарно-гигиеническая характеристика производства ПО включает в себя множество аспектов, которые необходимо учитывать для обеспечения здоровых и комфортных условий труда для сотрудников и сохранения безопасности окружающей среды.

				ПГУ1.090301.01.001 ПЗ	Лист
					40
Изм.	Лист	№ докум	Подпись Дата		

Заключение

В последние десять лет информатизация и автоматизация с каждым годом все больше проникает в различные сферы жизни человека. Благодаря им ускоряются различные процессы, повышается прозрачность ведения деятельности, повышается удобство доступа к информации.

Изучив потребности учеников и учащихся, в ходе выполнения данной выпускной квалификационной работы было решено создать сервис контроля успеваемости учащихся. Помимо непосредственно основной цели, заявленной в теме работы, было также решено добавить в этот сервис ряд дополнительных функций для того, чтобы сделать его еще полезнее для учителей, учеников и образовательного процесса в целом.

Для реализации данного программного продукта было решено выбрать наиболее подходящий стек: *Python + Django + HTML + CSS + Sqlite3*. Таким примитивные, на первый взгляд, инструменты, позволили сделать сервис достаточными простым и удобным, с интуитивно понятным интерфейсом.

В ходе выполнения работы была разработана система контроля, обладающая всем необходимым заявленным во введении функционалом, с разделением по ролям: и для учителей, и для учеников. Система строится на базе данных, хранящей информацию о пользователях, сервисах для загрузки домашних заданий, напоминаний, а также сервисе, дающем возможность послать сообщение от ученика учителю.

Система была успешно протестирована, весь заявленный функционал отработал, никаких огрехов в приложении при тестировании замечено не было.

В будущем данный сервис можно было бы улучшить, сделав возможность для преподавателя управлять сразу несколькими классами, сделать выбор типа работы при выставлении оценки, для ученика можно дать возможность считать в автоматическом режиме среднюю оценку по предмету, а также добавить отдельный личный кабинет для родителя.

				ПГУ1.090301.01.001 ПЗ	Лист
					41
Изм.	Лист	№ докум	Подпись Дата		

Список использованных источников

1. Лутц М. Изучаем *Python*; Вильямс – М., 2019. – 832 с.
2. Пинхам Э. *Django*: Основы; Сэмс – М., 2015 – 840 с.
3. Документация к фреймворку Django. [Электронный ресурс] / Режим доступа: <https://docs.djangoproject.com/en/4.2/> свободный.
4. Меле А. *Django 2* в примерах; ДМК Пресс – М., 2018 – 410 с.
5. Свейгарт Э. Большая книга проектов *Python*; Питер – М., 2022 – 432 с.
6. Бэйдер Д. Чистый *Python*. Тонкости программирования для профи; Питер – М., 2022 – 288 с.
7. Бэрри П. Изучаем программирование на *Python*; Эксмо – М., 2017 - 624 с.
8. Кольцов Д. *Python*. Полное руководство; Наука и техника – М., 2022 – 480 с.
9. Лоре А. Проектирование веб – *API*; ДМК Пресс – М., 2020 – 440 с.
10. Гарри П. *Python*. Разработка на основе тестирования; ДМК Пресс – М., 2018 – 622 с.

				<i>ПГУ1.090301.01.001 ПЗ</i>	Лист
					42
Изм.	Лист	№ докум	Подпись Дата		

Приложение А

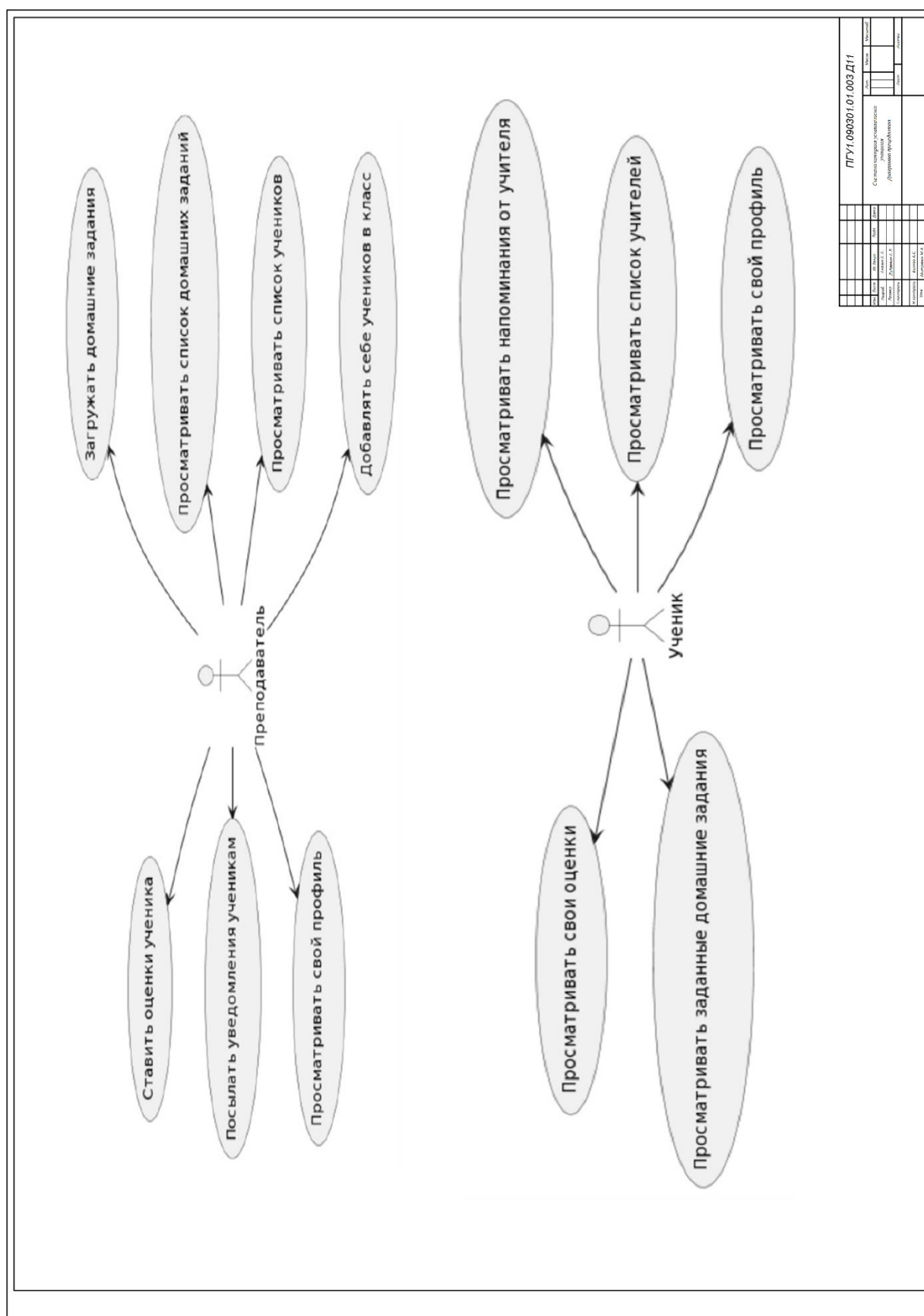
(обязательное)

Система контроля успеваемости учащихся

UML - диаграммы

				ПГУ1.090301.01.001 ПЗ	Лист
					43
Изм.	Лист	№ докум	Подпись Дата		

Приложение А.1 – Диаграмма вариантов использования



ПГУ1.090301.01.003 Д11									
№ п/п	Имя	Фамилия	Дата	Подпись	Подпись	Подпись	Подпись	Подпись	Подпись
1	Иванов	Иванов	Иванов	Иванов	Иванов	Иванов	Иванов	Иванов	Иванов
2	Петров	Петров	Петров	Петров	Петров	Петров	Петров	Петров	Петров
3	Сидоров	Сидоров	Сидоров	Сидоров	Сидоров	Сидоров	Сидоров	Сидоров	Сидоров
4	Смирнов	Смирнов	Смирнов	Смирнов	Смирнов	Смирнов	Смирнов	Смирнов	Смирнов
5	Климов	Климов	Климов	Климов	Климов	Климов	Климов	Климов	Климов
6	Куликов	Куликов	Куликов	Куликов	Куликов	Куликов	Куликов	Куликов	Куликов
7	Левин	Левин	Левин	Левин	Левин	Левин	Левин	Левин	Левин
8	Михайлов	Михайлов	Михайлов	Михайлов	Михайлов	Михайлов	Михайлов	Михайлов	Михайлов
9	Новиков	Новиков	Новиков	Новиков	Новиков	Новиков	Новиков	Новиков	Новиков
10	Осипов	Осипов	Осипов	Осипов	Осипов	Осипов	Осипов	Осипов	Осипов
11	Попов	Попов	Попов	Попов	Попов	Попов	Попов	Попов	Попов
12	Рябинин	Рябинин	Рябинин	Рябинин	Рябинин	Рябинин	Рябинин	Рябинин	Рябинин
13	Савин	Савин	Савин	Савин	Савин	Савин	Савин	Савин	Савин
14	Соловьев	Соловьев	Соловьев	Соловьев	Соловьев	Соловьев	Соловьев	Соловьев	Соловьев
15	Тихонов	Тихонов	Тихонов	Тихонов	Тихонов	Тихонов	Тихонов	Тихонов	Тихонов
16	Федотов	Федотов	Федотов	Федотов	Федотов	Федотов	Федотов	Федотов	Федотов
17	Харин	Харин	Харин	Харин	Харин	Харин	Харин	Харин	Харин
18	Цыганов	Цыганов	Цыганов	Цыганов	Цыганов	Цыганов	Цыганов	Цыганов	Цыганов
19	Чайков	Чайков	Чайков	Чайков	Чайков	Чайков	Чайков	Чайков	Чайков
20	Шаров	Шаров	Шаров	Шаров	Шаров	Шаров	Шаров	Шаров	Шаров
21	Шенников	Шенников	Шенников	Шенников	Шенников	Шенников	Шенников	Шенников	Шенников
22	Щеглов	Щеглов	Щеглов	Щеглов	Щеглов	Щеглов	Щеглов	Щеглов	Щеглов
23	Юдин	Юдин	Юдин	Юдин	Юдин	Юдин	Юдин	Юдин	Юдин
24	Якушев	Якушев	Якушев	Якушев	Якушев	Якушев	Якушев	Якушев	Якушев

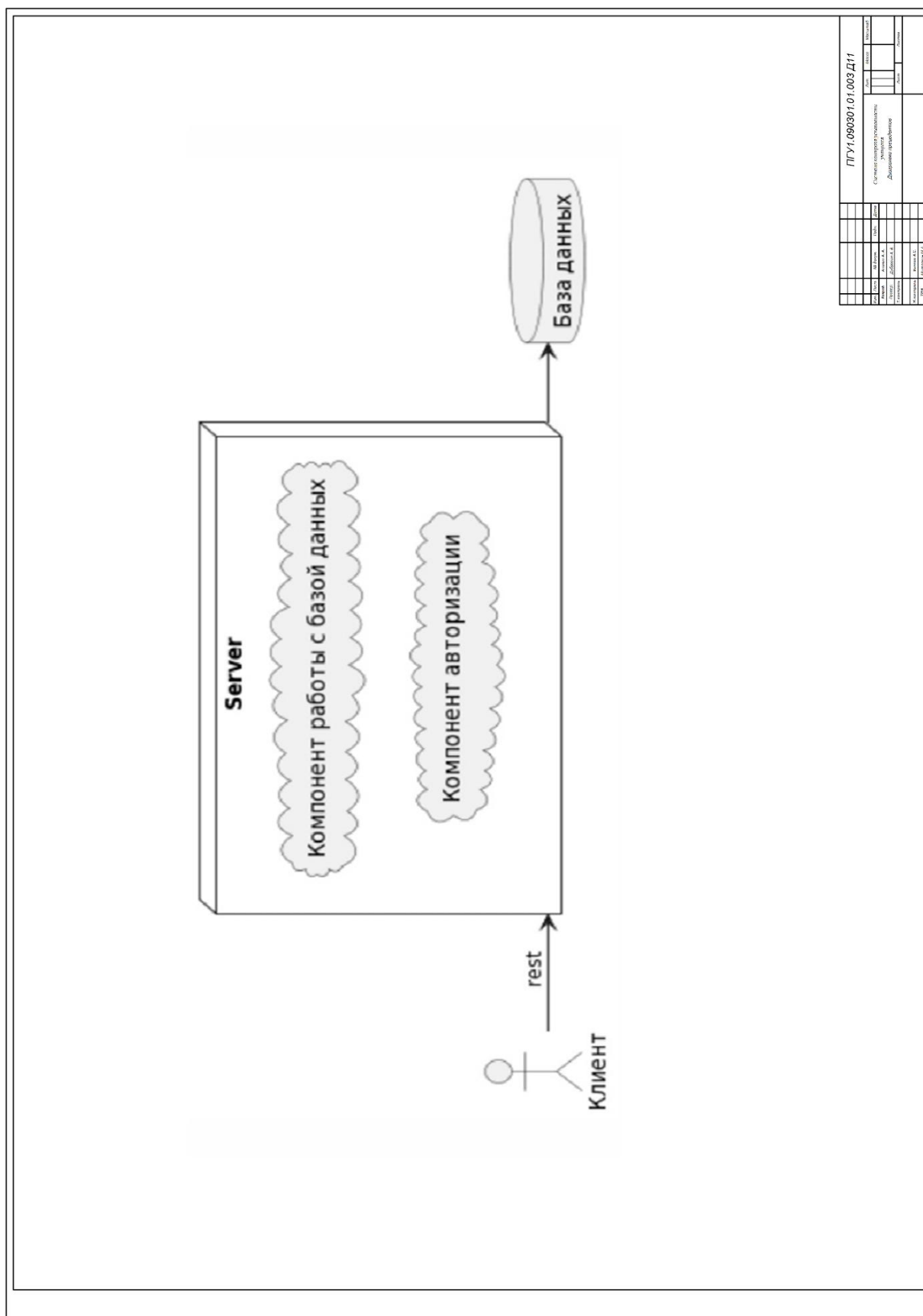
Изм.	Лист	№ докум	Подпись	Дата

ПГУ1.090301.01.001 ПЗ

Лист

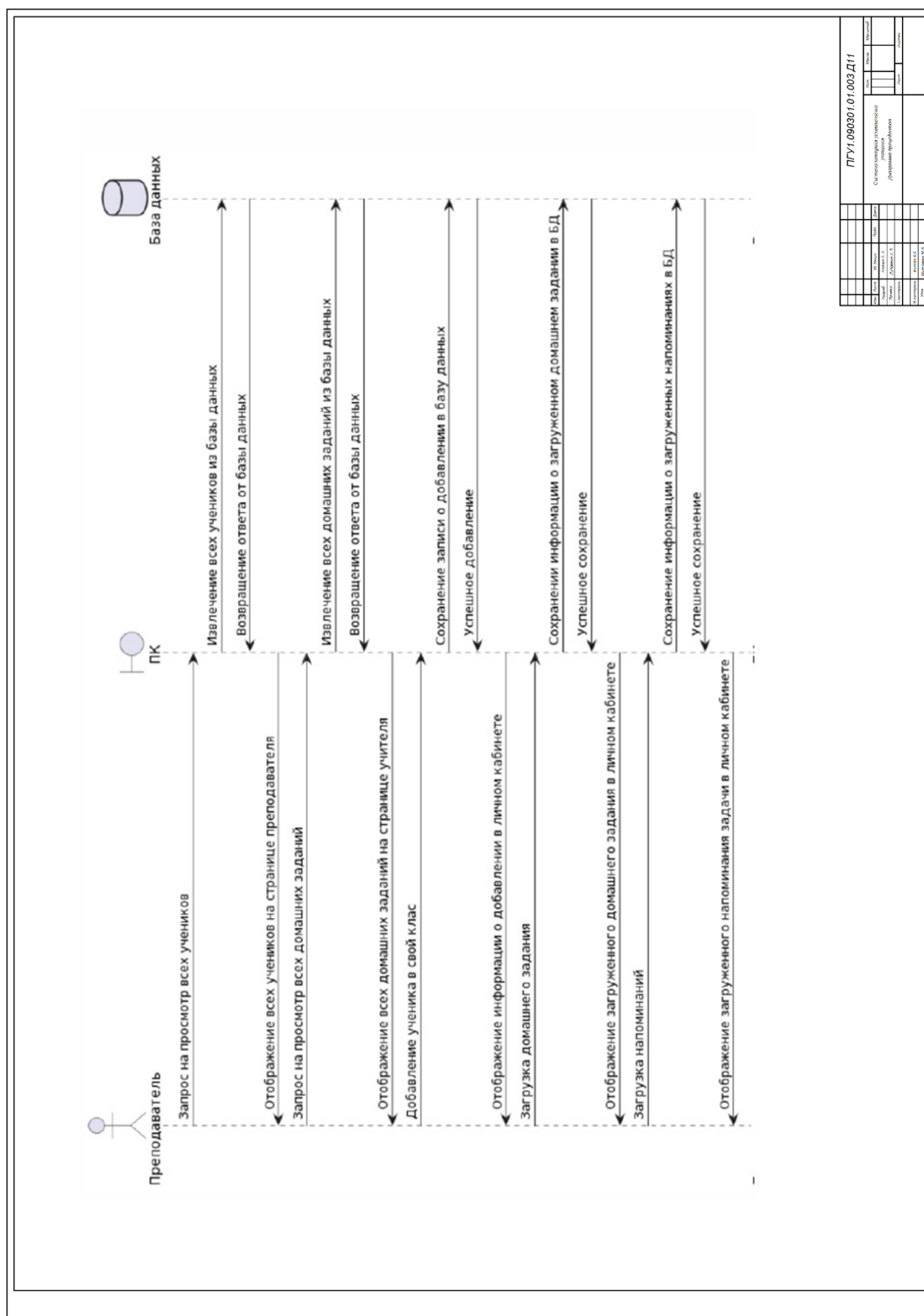
44

Приложение А.2 – Диаграмма развертывания

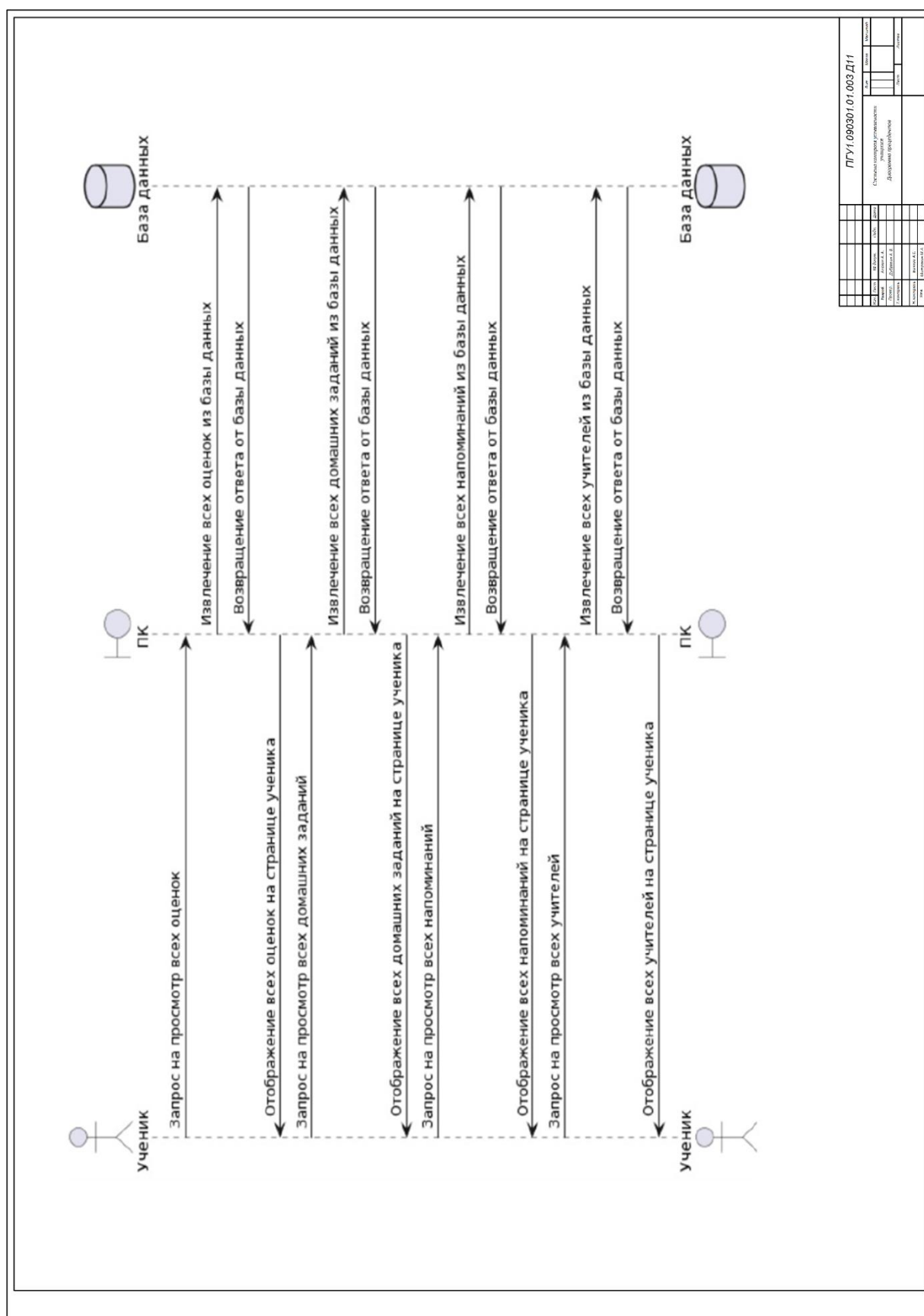


--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

Приложение А.3 – Диаграмма развертывания

[illegible]

Приложение А.4 – Диаграмма разворачивания



ПГУ1.090301.01.003 Д11		Служба контроля качества		Служба контроля качества	
Имя	Фамилия	Имя	Фамилия	Имя	Фамилия
Имя	Фамилия	Имя	Фамилия	Имя	Фамилия
Имя	Фамилия	Имя	Фамилия	Имя	Фамилия
Имя	Фамилия	Имя	Фамилия	Имя	Фамилия
Имя	Фамилия	Имя	Фамилия	Имя	Фамилия
Имя	Фамилия	Имя	Фамилия	Имя	Фамилия
Имя	Фамилия	Имя	Фамилия	Имя	Фамилия
Имя	Фамилия	Имя	Фамилия	Имя	Фамилия
Имя	Фамилия	Имя	Фамилия	Имя	Фамилия

Изм.	Лист	№ докум	Подпись	Дата

ПГУ1.090301.01.001 ПЗ

Лист

47

Приложение Б

(обязательное)

Система контроля успеваемости учащихся

Листинг программы

				ПГУ1.090301.01.001 ПЗ	Лист
					48
Изм.	Лист	№ докум	Подпись Дата		

Файл asgi.py

```
"""
ASGI config for classmanager project.

It exposes the ASGI callable as a module-level variable named
`application`.

For more information on this file, see
https://docs.djangoproject.com/en/3.0/howto/deployment/asgi/
"""

import os

from django.core.asgi import get_asgi_application

os.environ.setdefault('DJANGO_SETTINGS_MODULE',
    'classmanager.settings')

application = get_asgi_application()
```

Файл settings.py

```
"""
Django settings for studentmanager project.

Generated by 'django-admin startproject' using Django 3.0.5.

For more information on this file, see
https://docs.djangoproject.com/en/3.0/topics/settings/

For the full list of settings and their values, see
https://docs.djangoproject.com/en/3.0/ref/settings/
"""

import os

# Build paths inside the project like this:
os.path.join(BASE_DIR, ...)

BASE_DIR =
os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
```

				ПГУ1.090301.01.001 ПЗ	Лист
					49
Изм.	Лист	№ докум	Подпись Дата		

```

TEMPLATE_DIR = os.path.join(BASE_DIR, "template")
STATIC_DIR = os.path.join(BASE_DIR, 'static')
MEDIA_DIR = os.path.join(BASE_DIR, 'media')


# Quick-start development settings - unsuitable for production
# See
https://docs.djangoproject.com/en/3.0/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production
secret!

SECRET_KEY = '13x8r2se41dawv$7!a-
p_h4ni$^i^+r@le7wuadd6gmzb=$ss*'

# SECURITY WARNING: don't run with debug turned on in
production!

DEBUG = True


ALLOWED_HOSTS = []


AUTH_USER_MODEL = 'classroom.User'


# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'classroom',

```

				ПГУ1.090301.01.001 ПЗ	Лист
					50
Изм.	Лист	№ докум	Подпись Дата		

```

        'bootstrap4',
    ]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    #'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'classmanager.urls'

TEMPLATES = [
    {
        'BACKEND':
'django.template.backends.django.DjangoTemplates',
        'DIRS': [TEMPLATE_DIR,],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',

'django.contrib.messages.context_processors.messages',

            ],
        },
    ],
]

```

```

WSGI_APPLICATION = 'classmanager.wsgi.application'

# Database
# https://docs.djangoproject.com/en/3.0/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),
    }
}

# Password validation
# https://docs.djangoproject.com/en/3.0/ref/settings/#auth-
password-validators

AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME':
        'django.contrib.auth.password_validation.UserAttributeSimilarity
Validator',
    },
    {
        'NAME':
        'django.contrib.auth.password_validation.MinimumLengthValidator'
        ,
    },
    {
        'NAME':
        'django.contrib.auth.password_validation.CommonPasswordValidator
        ',
    },

```

				ПГУ1.090301.01.001 ПЗ	Лист
					52
Изм.	Лист	№ докум	Подпись Дата		

```

    },
    {
        'NAME':
'django.contrib.auth.password_validation.NumericPasswordValidato
r',
    },
]

```

```

# Internationalization
# https://docs.djangoproject.com/en/3.0/topics/i18n/

```

```
LANGUAGE_CODE = 'en-us'
```

```
TIME_ZONE = 'Asia/Kolkata'
```

```
USE_I18N = True
```

```
USE_L10N = True
```

```
USE_TZ = True
```

```

# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/3.0/howto/static-files/

```

```

STATIC_URL = '/static/'
STATICFILES_DIRS = [STATIC_DIR,]

```

```

MEDIA_ROOT = MEDIA_DIR
MEDIA_URL = '/media/'

```

				ПГУ1.090301.01.001 ПЗ	Лист
					53
Изм.	Лист	№ докум	Подпись Дата		

Файл urls.py

```
"""studentmanager URL Configuration
```

```
The `urlpatterns` list routes URLs to views. For more
information please see:
```

```
https://docs.djangoproject.com/en/3.0/topics/http/urls/
```

```
Examples:
```

```
Function views
```

```
1. Add an import:  from my_app import views
```

```
2. Add a URL to urlpatterns:  path('', views.home,
name='home')
```

```
Class-based views
```

```
1. Add an import:  from other_app.views import Home
```

```
2. Add a URL to urlpatterns:  path('', Home.as_view(),
name='home')
```

```
Including another URLconf
```

```
1. Import the include() function: from django.urls
import include, path
```

```
2. Add a URL to urlpatterns:  path('blog/',
include('blog.urls'))
```

```
"""
```

```
from django.contrib import admin
```

```
from django.urls import path,include
```

```
from . import views
```

```
from django.conf import settings
```

```
from django.conf.urls.static import static
```

```
urlpatterns = [
```

```
    path('admin/', admin.site.urls),
```

```
    path('',views.HomePage.as_view(),name="home"),
```

```
    path('accounts/', include('django.contrib.auth.urls')),
```

```
    path('classroom/',include('classroom.urls')),
```

				ПГУ1.090301.01.001 ПЗ	Лист
					54
Изм.	Лист	№ докум	Подпись Дата		

```
] + static(settings.MEDIA_URL,
document_root=settings.MEDIA_ROOT) }
}
```

Файл views.py

```
from django.views.generic import TemplateView

class HomePage(TemplateView):
    template_name = "classroom/index.html"
```

Файл wsgi.py

```
"""
WSGI config for classmanager project.

It exposes the WSGI callable as a module-level variable named
`application`.

For more information on this file, see
https://docs.djangoproject.com/en/3.0/howto/deployment/wsgi/
"""

import os

from django.core.wsgi import get_wsgi_application

os.environ.setdefault('DJANGO_SETTINGS_MODULE',
'classmanager.settings')

application = get_wsgi_application()
```

Файл admin.py

```
from django.contrib import admin

from django.contrib.auth.admin import UserAdmin

from .models import
User, Student, StudentMarks, Teacher, StudentsInClass

# Register your models here.
```

				ПГУ1.090301.01.001 ПЗ	Лист
					55
Изм.	Лист	№ докум	Подпись Дата		

```

admin.site.register(User, UserAdmin)
admin.site.register(Student)
admin.site.register(StudentMarks)
admin.site.register(Teacher)
admin.site.register(StudentsInClass)

```

Файл apps.py

```

from django.apps import AppConfig

```

```

class ClassroomConfig(AppConfig):
    name = 'classroom'

```

Файл forms.py

```

from django import forms
from django.contrib.auth.forms import UserCreationForm
from classroom.models import
User, Teacher, Student, StudentMarks, MessageToTeacher, ClassNotice, C
lassAssignment, SubmitAssignment

```

```

## User Login Form (Applied in both student and teacher login)

```

```

class UserForm(UserCreationForm):
    class Meta():
        model = User
        fields = ['username', 'password1', 'password2']
        widgets = {
            'username':
forms.TextInput(attrs={'class': 'answer'}),
            'password1':
forms.PasswordInput(attrs={'class': 'answer'}),
            'password2':
forms.PasswordInput(attrs={'class': 'answer'}),
        }

```

```

## Teacher Registration Form

```

```

class TeacherProfileForm(forms.ModelForm):
    class Meta():

```

				ПГУ1.090301.01.001 ПЗ	Лист
					56
Изм.	Лист	№ докум	Подпись Дата		

```

        model = Teacher

        fields = ['name', 'subject_name', 'phone', 'email']

        widgets = {

            'name':
forms.TextInput(attrs={'class': 'answer'}),

            'subject_name':
forms.TextInput(attrs={'class': 'answer'}),

            'phone':
forms.NumberInput(attrs={'class': 'answer'}),

            'email':
forms.EmailInput(attrs={'class': 'answer'}),

        }

## Teacher Profile Update Form
class TeacherProfileUpdateForm(forms.ModelForm):

    class Meta():

        model = Teacher

        fields =
['name', 'subject_name', 'email', 'phone', 'teacher_profile_pic']

## Student Registration Form
class StudentProfileForm(forms.ModelForm):

    class Meta():

        model = Student

        fields = ['name', 'roll_no', 'phone', 'email']

        widgets = {

            'name':
forms.TextInput(attrs={'class': 'answer'}),

            'roll_no':
forms.NumberInput(attrs={'class': 'answer'}),

            'phone':
forms.NumberInput(attrs={'class': 'answer'}),

            'email':
forms.EmailInput(attrs={'class': 'answer'}),

        }

```

```

## Student profile update form
class StudentProfileUpdateForm(forms.ModelForm):
    class Meta():
        model = Student
        fields =
['name','roll_no','email','phone','student_profile_pic']

## Form for uploading marks and also for updating it.
class MarksForm(forms.ModelForm):
    class Meta():
        model = StudentMarks
        fields =
['subject_name','marks_obtained','maximum_marks']

## Writing message to teacher
class MessageForm(forms.ModelForm):
    class Meta():
        model = MessageToTeacher
        fields = ['message']

## Writing notice in the class
class NoticeForm(forms.ModelForm):
    class Meta():
        model = ClassNotice
        fields = ['message']

## Form for uploading or updating assignment (teachers only)
class AssignmentForm(forms.ModelForm):
    class Meta():
        model = ClassAssignment
        fields = ['assignment_name','assignment']

```

```
## Form for submitting assignment (Students only)
```

```
class SubmitForm(forms.ModelForm):
```

```
    class Meta():
```

```
        model = SubmitAssignment
```

```
        fields = ['submit']
```

Файл models.py

```
from django.db import models
```

```
from django.contrib.auth.models import AbstractUser
```

```
from django.urls import reverse
```

```
from django.conf import settings
```

```
import misaka
```

```
# Create your models here.
```

```
class User(AbstractUser):
```

```
    is_student = models.BooleanField(default=False)
```

```
    is_teacher = models.BooleanField(default=False)
```

```
class Student(models.Model):
```

```
    user =
```

```
models.OneToOneField(User, on_delete=models.CASCADE, primary_key=True, related_name='Student')
```

```
    name=models.CharField(max_length=250)
```

```
    roll_no = models.CharField(max_length=50)
```

```
    email = models.EmailField(max_length=254)
```

```
    phone = models.IntegerField()
```

```
    student_profile_pic =
```

```
models.ImageField(upload_to="classroom/student_profile_pic", blank=True)
```

```
    def get_absolute_url(self):
```

				ПГУ1.090301.01.001 ПЗ	Лист
					59
Изм.	Лист	№ докум	Подпись Дата		

```

        return
reverse('classroom:student_detail',kwargs={'pk':self.pk})

def __str__(self):
    return self.name

class Meta:
    ordering = ['roll_no']

class Teacher(models.Model):
    user =
models.OneToOneField(User,on_delete=models.CASCADE,primary_key=Tr
ue,related_name='Teacher')

    name = models.CharField(max_length=250)
    subject_name = models.CharField(max_length=250)
    email = models.EmailField(max_length=254)
    phone = models.IntegerField()

    teacher_profile_pic =
models.ImageField(upload_to="classroom/teacher_profile_pic",blan
k=True)

    class_students =
models.ManyToManyField(Student,through="StudentsInClass")

    def get_absolute_url(self):
        return
reverse('classroom:teacher_detail',kwargs={'pk':self.pk})

def __str__(self):
    return self.name

class StudentMarks(models.Model):
    teacher =
models.ForeignKey(Teacher,related_name='given_marks',on_delete=m
odels.CASCADE)

```

				ПГУ1.090301.01.001 ПЗ	Лист
					60
Изм.	Лист	№ докум	Подпись Дата		

```

        student =
models.ForeignKey(Student,related_name="marks",on_delete=models.
CASCADE)

        subject_name = models.CharField(max_length=250)

        marks_obtained = models.IntegerField()

        maximum_marks = models.IntegerField()


    def __str__(self):
        return self.subject_name


class StudentsInClass(models.Model):

        teacher =
models.ForeignKey(Teacher,related_name="class_teacher",on_delete
=models.CASCADE)

        student =
models.ForeignKey(Student,related_name="user_student_name",on_de
lete=models.CASCADE)


    def __str__(self):
        return self.student.name


    class Meta:

        unique_together = ('teacher','student')


class MessageToTeacher(models.Model):

        student =
models.ForeignKey(Student,related_name='student',on_delete=model
s.CASCADE)

        teacher =
models.ForeignKey(Teacher,related_name='messages',on_delete=mode
ls.CASCADE)

        created_at = models.DateTimeField(auto_now=True)

        message = models.TextField()

        message_html = models.TextField(editable=False)

```

```

def __str__(self):
    return self.message

def save(self, *args, **kwargs):
    self.message_html = misaka.html(self.message)
    super().save(*args, **kwargs)

class Meta:
    ordering = ['-created_at']
    unique_together = ['student', 'message']

class ClassNotice(models.Model):
    teacher =
models.ForeignKey(Teacher, related_name='teacher', on_delete=models.CASCADE)

    students =
models.ManyToManyField(Student, related_name='class_notice')

    created_at = models.DateTimeField(auto_now=True)
    message = models.TextField()
    message_html = models.TextField(editable=False)

def __str__(self):
    return self.message

def save(self, *args, **kwargs):
    self.message_html = misaka.html(self.message)
    super().save(*args, **kwargs)

class Meta:
    ordering = ['-created_at']
    unique_together = ['teacher', 'message']

class ClassAssignment(models.Model):

```

				ПГУ1.090301.01.001 ПЗ	Лист
					62
Изм.	Лист	№ докум	Подпись Дата		

```

        student =
models.ManyToManyField(Student,related_name='student_assignment'
)

        teacher =
models.ForeignKey(Teacher,related_name='teacher_assignment',on_d
elete=models.CASCADE)

        created_at = models.DateTimeField(auto_now=True)

        assignment_name = models.CharField(max_length=250)

        assignment = models.FileField(upload_to='assignments')

def __str__(self):
    return self.assignment_name

class Meta:
    ordering = ['-created_at']

class SubmitAssignment(models.Model):

    student =
models.ForeignKey(Student,related_name='student_submit',on_delet
e=models.CASCADE)

    teacher =
models.ForeignKey(Teacher,related_name='teacher_submit',on_delet
e=models.CASCADE)

    created_at = models.DateTimeField(auto_now=True)

    submitted_assignment =
models.ForeignKey(ClassAssignment,related_name='submission_for_a
ssignment',on_delete=models.CASCADE)

    submit = models.FileField(upload_to='Submission')

def __str__(self):
    return
"Submitted"+str(self.submitted_assignment.assignment_name)

class Meta:
    ordering = ['-created_at']

```

				ПГУ1.090301.01.001 ПЗ	Лист
					63
Изм.	Лист	№ докум	Подпись Дата		

Файл urls.py

```
from django.urls import path
from classroom import views

app_name = 'classroom'

urlpatterns = [
    path('signup/', views.SignUp, name="signup"),

    path('signup/student_signup/', views.StudentSignUp, name="StudentS
ignUp"),

    path('signup/teacher_signup/', views.TeacherSignUp, name="TeacherS
ignUp"),

    path('login/', views.user_login, name="login"),
    path('logout/', views.user_logout, name="logout"),

    path('student/<int:pk>/', views.StudentDetailView.as_view(), name=
"student_detail"),

    path('teacher/<int:pk>/', views.TeacherDetailView.as_view(), name=
"teacher_detail"),

    path('update/student/<int:pk>/', views.StudentUpdateView, name="st
udent_update"),

    path('update/teacher/<int:pk>/', views.TeacherUpdateView, name="te
acher_update"),

    path('student/<int:pk>/enter_marks', views.add_marks, name="enter_
marks"),

    path('student/<int:pk>/marks_list', views.student_marks_list, name
="student_marks_list"),

    path('marks/<int:pk>/update', views.update_marks, name="update_mar
ks"),
```

				ПГУ1.090301.01.001 ПЗ	Лист
					64
Изм.	Лист	№ докум	Подпись Дата		

```

path('student/<int:pk>/add',views.add_student.as_view(),name="ad
d_student"),

path('student_added/',views.student_added,name="student_added"),

path('students_list/',views.students_list,name="students_list"),

path('teachers_list/',views.teachers_list,name="teachers_list"),

path('teacher/class_students_list',views.class_students_list,nam
e="class_student_list"),

path('student/<int:pk>/all_marks',views.StudentAllMarksList.as_v
iew(),name="all_marks_list"),

path('student/<int:pk>/message',views.write_message,name="write_
message"),

path('teacher/<int:pk>/messages_list',views.messages_list,name="
messages_list"),

path('teacher/write_notice',views.add_notice,name="write_notice"
),

path('student/<int:pk>/class_notice',views.class_notice,name="cl
ass_notice"),

path('upload_assignment/',views.upload_assignment,name="upload_a
ssignment"),

path('class_assignment/',views.class_assignment,name="class_assi
gnment"),

path('assignment_list/',views.assignment_list,name="assignment_l
ist"),

path('update_assignment/<int:id>/',views.update_assignment,name=
"update_assignment"),

```

				ПГУ1.090301.01.001 ПЗ	Лист
					65
Изм.	Лист	№ докум	Подпись Дата		

```

path('assignment_delete/<int:id>/', views.assignment_delete, name=
"assignment_delete"),

path('submit_assignment/<int:id>/', views.submit_assignment, name=
"submit_assignment"),

    path('submit_list/', views.submit_list, name="submit_list"),

path('change_password/', views.change_password, name="change_passw
ord"),
]

```

Файл views.py

```

from django.shortcuts import render, get_object_or_404, redirect
from django.views import generic
from django.views.generic import (View, TemplateView,
                                   ListView, DetailView,
                                   CreateView, UpdateView,
                                   DeleteView)

from django.utils.decorators import method_decorator
from django.contrib import messages
from django.contrib.auth.mixins import LoginRequiredMixin

# Create your views here.

from classroom.forms import
UserForm, TeacherProfileForm, StudentProfileForm, MarksForm, Message
Form, NoticeForm, AssignmentForm, SubmitForm, TeacherProfileUpdateFo
rm, StudentProfileUpdateForm

from django.urls import reverse

from django.contrib.auth.decorators import login_required
from django.contrib.auth import authenticate, login,
logout, update_session_auth_hash

from django.http import HttpResponseRedirect, HttpResponse

from classroom import models

from classroom.models import
StudentsInClass, StudentMarks, ClassAssignment, SubmitAssignment, St
udent, Teacher

from django.contrib.auth.forms import PasswordChangeForm

```

				ПГУ1.090301.01.001 ПЗ	Лист
					66
Изм.	Лист	№ докум	Подпись Дата		

```

from django.db.models import Q

# For Teacher Sign Up
def TeacherSignUp(request):
    user_type = 'teacher'
    registered = False

    if request.method == "POST":
        user_form = UserForm(data = request.POST)
        teacher_profile_form = TeacherProfileForm(data =
request.POST)

        if user_form.is_valid() and
teacher_profile_form.is_valid():

            user = user_form.save()
            user.is_teacher = True
            user.save()

            profile = teacher_profile_form.save(commit=False)
            profile.user = user
            profile.save()

            registered = True
        else:
            print(user_form.errors,teacher_profile_form.errors)
    else:
        user_form = UserForm()
        teacher_profile_form = TeacherProfileForm()

    return
render(request, 'classroom/teacher_signup.html',{'user_form':user

```

				ПГУ1.090301.01.001 ПЗ	Лист
					67
Изм.	Лист	№ докум	Подпись Дата		

```
_form,'teacher_profile_form':teacher_profile_form,'registered':registered,'user_type':user_type})
```

```
### For Student Sign Up
```

```
def StudentSignUp(request):
```

```
    user_type = 'student'
```

```
    user_form = ''
```

```
    registered = False
```

```
    if request.method == "POST":
```

```
        user_form = UserForm(data = request.POST)
```

```
        student_profile_form = StudentProfileForm(data = request.POST)
```

```
        if user_form.is_valid() and student_profile_form.is_valid():
```

```
            user = user_form.save()
```

```
            user.is_student = True
```

```
            user.save()
```

```
            profile = student_profile_form.save(commit=False)
```

```
            profile.user = user
```

```
            profile.save()
```

```
            registered = True
```

```
        else:
```

```
            print(user_form.errors,student_profile_form.errors)
```

```
    else:
```

```
        user_form = UserForm()
```

```
        student_profile_form = StudentProfileForm()
```

				ПГУ1.090301.01.001 ПЗ	Лист
					68
Изм.	Лист	№ докум	Подпись Дата		

```

return render(request, 'classroom/student_signup.html',
                {'user_form':user_form,
                 'student_profile_form':student_profile_form,
                 'registered':registered,
                 'user_type':user_type})

## Sign Up page which will ask whether you are teacher or
student.

def SignUp(request):
    return render(request, 'classroom/signup.html', {})

## login view.

def user_login(request):
    if request.method == "POST":
        username = request.POST.get('username')
        password = request.POST.get('password')

        user = authenticate(username=username,password=password)

        if user:
            if user.is_active:
                login(request,user)
                return HttpResponseRedirect(reverse('home'))

            else:
                return HttpResponse("Account not active")

        else:
            messages.error(request, "Invalid Details")
            return redirect('classroom:login')

    else:
        return render(request, 'classroom/login.html', {})

```

				ПГУ1.090301.01.001 ПЗ	Лист
					69
Изм.	Лист	№ докум	Подпись Дата		

```

## logout view.
@login_required
def user_logout(request):
    logout(request)
    return HttpResponseRedirect(reverse('home'))

## User Profile of student.
class StudentDetailView(LoginRequiredMixin,DetailView):
    context_object_name = "student"
    model = models.Student
    template_name = 'classroom/student_detail_page.html'

## User Profile for teacher.
class TeacherDetailView(LoginRequiredMixin,DetailView):
    context_object_name = "teacher"
    model = models.Teacher
    template_name = 'classroom/teacher_detail_page.html'

## Profile update for students.
@login_required
def StudentUpdateView(request,pk):
    profile_updated = False
    student = get_object_or_404(models.Student,pk=pk)
    if request.method == "POST":
        form =
StudentProfileUpdateForm(request.POST,instance=student)
        if form.is_valid():
            profile = form.save(commit=False)
            if 'student_profile_pic' in request.FILES:
                profile.student_profile_pic =
request.FILES['student_profile_pic']

```

				ПГУ1.090301.01.001 ПЗ	Лист
					70
Изм.	Лист	№ докум	Подпись Дата		

```

        profile.save()

        profile_updated = True

    else:

        form = StudentProfileUpdateForm(request.POST or
None,instance=student)

        return
render(request,'classroom/student_update_page.html',{'profile_up
dated':profile_updated,'form':form})

## Profile update for teachers.
@login_required
def TeacherUpdateView(request,pk):
    profile_updated = False

    teacher = get_object_or_404(models.Teacher,pk=pk)

    if request.method == "POST":

        form =
TeacherProfileUpdateForm(request.POST,instance=teacher)

        if form.is_valid():

            profile = form.save(commit=False)

            if 'teacher_profile_pic' in request.FILES:

                profile.teacher_profile_pic =
request.FILES['teacher_profile_pic']

                profile.save()

                profile_updated = True

        else:

            form = TeacherProfileUpdateForm(request.POST or
None,instance=teacher)

            return
render(request,'classroom/teacher_update_page.html',{'profile_up
dated':profile_updated,'form':form})

## List of all students that teacher has added in their class.
def class_students_list(request):

    query = request.GET.get("q", None)

```

				ПГУ1.090301.01.001 ПЗ	Лист
					71
Изм.	Лист	№ докум	Подпись Дата		

```

        students =
StudentsInClass.objects.filter(teacher=request.user.Teacher)

        students_list = [x.student for x in students]

        qs = Student.objects.all()

        if query is not None:

            qs = qs.filter(

                Q(name__icontains=query)

            )

        qs_one = []

        for x in qs:

            if x in students_list:

                qs_one.append(x)

            else:

                pass

        context = {

            "class_students_list": qs_one,

        }

        template = "classroom/class_students_list.html"

        return render(request, template, context)

class ClassStudentsListView(LoginRequiredMixin,DetailView):

    model = models.Teacher

    template_name = "classroom/class_students_list.html"

    context_object_name = "teacher"

## For Marks obtained by the student in all subjects.

class StudentAllMarksList(LoginRequiredMixin,DetailView):

    model = models.Student

    template_name = "classroom/student_allmarks_list.html"

    context_object_name = "student"

## To give marks to a student.

```

				ПГУ1.090301.01.001 ПЗ	Лист
					72
Изм.	Лист	№ докум	Подпись Дата		

```

@login_required
def add_marks(request,pk):
    marks_given = False
    student = get_object_or_404(models.Student,pk=pk)
    if request.method == "POST":
        form = MarksForm(request.POST)
        if form.is_valid():
            marks = form.save(commit=False)
            marks.student = student
            marks.teacher = request.user.Teacher
            marks.save()
            messages.success(request,'Marks uploaded
successfully!')
            return redirect('classroom:submit_list')
        else:
            form = MarksForm()
    return
render(request,'classroom/add_marks.html',{'form':form,'student'
:student,'marks_given':marks_given})

## For updating marks.
@login_required
def update_marks(request,pk):
    marks_updated = False
    obj = get_object_or_404(StudentMarks,pk=pk)
    if request.method == "POST":
        form = MarksForm(request.POST,instance=obj)
        if form.is_valid():
            marks = form.save(commit=False)
            marks.save()
            marks_updated = True
        else:
            form = MarksForm(request.POST or None,instance=obj)

```

				ПГУ1.090301.01.001 ПЗ	Лист
					73
Изм.	Лист	№ докум	Подпись Дата		

```

        return
    render(request, 'classroom/update_marks.html', {'form': form, 'marks_
_updated': marks_updated})

## For writing notice which will be sent to all class students.
@login_required
def add_notice(request):
    notice_sent = False
    teacher = request.user.Teacher
    students = StudentsInClass.objects.filter(teacher=teacher)
    students_list = [x.student for x in students]

    if request.method == "POST":
        notice = NoticeForm(request.POST)
        if notice.is_valid():
            object = notice.save(commit=False)
            object.teacher = teacher
            object.save()
            object.students.add(*students_list)
            notice_sent = True
        else:
            notice = NoticeForm()

    return
    render(request, 'classroom/write_notice.html', {'notice': notice, 'n
otice_sent': notice_sent})

## For student writing message to teacher.
@login_required
def write_message(request, pk):
    message_sent = False
    teacher = get_object_or_404(models.Teacher, pk=pk)

    if request.method == "POST":

```

```

        form = MessageForm(request.POST)

        if form.is_valid():

            mssg = form.save(commit=False)

            mssg.teacher = teacher

            mssg.student = request.user.Student

            mssg.save()

            message_sent = True

        else:

            form = MessageForm()

        return
    render(request, 'classroom/write_message.html', {'form': form, 'teacher': teacher, 'message_sent': message_sent})

## For the list of all the messages teacher have received.
@login_required
def messages_list(request, pk):

    teacher = get_object_or_404(models.Teacher, pk=pk)

    return
    render(request, 'classroom/messages_list.html', {'teacher': teacher
    })

## Student can see all notice given by their teacher.
@login_required
def class_notice(request, pk):

    student = get_object_or_404(models.Student, pk=pk)

    return
    render(request, 'classroom/class_notice_list.html', {'student': student
    })

## To see the list of all the marks given by the teacher to a
specific student.
@login_required
def student_marks_list(request, pk):

    error = True

```

				ПГУ1.090301.01.001 ПЗ	Лист
					75
Изм.	Лист	№ докум	Подпись Дата		

```

        student = get_object_or_404(models.Student,pk=pk)

        teacher = request.user.Teacher

        given_marks =
StudentMarks.objects.filter(teacher=teacher,student=student)

        return
render(request,'classroom/student_marks_list.html',{'student':st
udent,'given_marks':given_marks})

## To add student in the class.
class add_student(LoginRequiredMixin,generic.RedirectView):

    def get_redirect_url(self,*args,**kwargs):
        return reverse('classroom:students_list')

    def get(self,request,*args,**kwargs):
        student =
get_object_or_404(models.Student,pk=self.kwargs.get('pk'))

        try:

StudentsInClass.objects.create(teacher=self.request.user.Teacher
,student=student)

        except:

            messages.warning(self.request,'warning, Student
already in class!')

        else:

            messages.success(self.request,'{} successfully
added!'.format(student.name))

        return super().get(request,*args,**kwargs)

@login_required
def student_added(request):
    return render(request,'classroom/student_added.html',{})

```

				ПГУ1.090301.01.001 ПЗ	Лист
					76
Изм.	Лист	№ докум	Подпись Дата		

```
## List of students which are not added by teacher in their
class.
```

```
def students_list(request):
    query = request.GET.get("q", None)

    students =
StudentsInClass.objects.filter(teacher=request.user.Teacher)
    students_list = [x.student for x in students]
    qs = Student.objects.all()
    if query is not None:
        qs = qs.filter(
            Q(name__icontains=query)
        )
    qs_one = []
    for x in qs:
        if x in students_list:
            pass
        else:
            qs_one.append(x)

    context = {
        "students_list": qs_one,
    }
    template = "classroom/students_list.html"
    return render(request, template, context)
```

```
## List of all the teacher present in the portal.
```

```
def teachers_list(request):
    query = request.GET.get("q", None)
    qs = Teacher.objects.all()
    if query is not None:
        qs = qs.filter(
```

				ПГУ1.090301.01.001 ПЗ	Лист
					77
Изм.	Лист	№ докум	Подпись Дата		

```

        Q(name__icontains=query)
    )

    context = {
        "teachers_list": qs,
    }

    template = "classroom/teachers_list.html"
    return render(request, template, context)

#####

## Teacher uploading assignment.
@login_required
def upload_assignment(request):
    assignment_uploaded = False
    teacher = request.user.Teacher

    students =
Student.objects.filter(user_student_name__teacher=request.user.T
eacher)

    if request.method == 'POST':
        form = AssignmentForm(request.POST, request.FILES)
        if form.is_valid():
            upload = form.save(commit=False)
            upload.teacher = teacher

            students =
Student.objects.filter(user_student_name__teacher=request.user.T
eacher)

            upload.save()
            upload.student.add(*students)
            assignment_uploaded = True
        else:
            form = AssignmentForm()

```

				ПГУ1.090301.01.001 ПЗ	Лист
					78
Изм.	Лист	№ докум	Подпись Дата		

```

        return
    render(request, 'classroom/upload_assignment.html', {'form': form, '
assignment_uploaded': assignment_uploaded})

## Students getting the list of all the assignments uploaded by
their teacher.
@login_required
def class_assignment(request):
    student = request.user.Student

    assignment =
SubmitAssignment.objects.filter(student=student)

    assignment_list = [x.submitted_assignment for x in
assignment]

    return
    render(request, 'classroom/class_assignment.html', {'student': stud
ent, 'assignment_list': assignment_list})

## List of all the assignments uploaded by the teacher himself.
@login_required
def assignment_list(request):
    teacher = request.user.Teacher

    return
    render(request, 'classroom/assignment_list.html', {'teacher': teach
er})

## For updating the assignments later.
@login_required
def update_assignment(request, id=None):
    obj = get_object_or_404(ClassAssignment, id=id)
    form = AssignmentForm(request.POST or None, instance=obj)
    context = {
        "form": form
    }

    if form.is_valid():
        obj = form.save(commit=False)

```

				ПГУ1.090301.01.001 ПЗ	Лист
					79
Изм.	Лист	№ докум	Подпись Дата		

```

        if 'assignment' in request.FILES:
            obj.assignment = request.FILES['assignment']
            obj.save()

            messages.success(request, "Updated
Assignment".format(obj.assignment_name))

            return redirect('classroom:assignment_list')

        template = "classroom/update_assignment.html"
        return render(request, template, context)

## For deleting the assignment.
@login_required
def assignment_delete(request, id=None):
    obj = get_object_or_404(ClassAssignment, id=id)
    if request.method == "POST":
        obj.delete()

        messages.success(request, "Assignment Removed")

        return redirect('classroom:assignment_list')

    context = {
        "object": obj,
    }

    template = "classroom/assignment_delete.html"
    return render(request, template, context)

## For students submitting their assignment.
@login_required
def submit_assignment(request, id=None):
    student = request.user.Student
    assignment = get_object_or_404(ClassAssignment, id=id)
    teacher = assignment.teacher

    if request.method == 'POST':
        form = SubmitForm(request.POST, request.FILES)

        if form.is_valid():

```

```

        upload = form.save(commit=False)
        upload.teacher = teacher
        upload.student = student
        upload.submitted_assignment = assignment
        upload.save()

        return redirect('classroom:class_assignment')

    else:

        form = SubmitForm()

        return
render(request, 'classroom/submit_assignment.html', {'form': form, }
)

## To see all the submissions done by the students.
@login_required
def submit_list(request):
    teacher = request.user.Teacher

    return
render(request, 'classroom/submit_list.html', {'teacher': teacher})

#####

## For changing password.
@login_required
def change_password(request):
    if request.method == 'POST':

        form = PasswordChangeForm(data=request.POST ,
user=request.user)

        if form.is_valid():
            form.save()
            update_session_auth_hash(request, form.user)
            messages.success(request, "Password changed")

```

				ПГУ1.090301.01.001 ПЗ	Лист
					81
Изм.	Лист	№ докум	Подпись Дата		

```

        return redirect('home')
    else:
        return redirect('classroom:change_password')
else:
    form = PasswordChangeForm(user=request.user)
    args = {'form':form}
    return
render(request, 'classroom/change_password.html', args)

```

Файл 0001_initial.py

Generated by Django 3.0.5 on 2020-05-25 19:21

```

from django.conf import settings
import django.contrib.auth.models
import django.contrib.auth.validators
from django.db import migrations, models
import django.db.models.deletion
import django.utils.timezone

```

```
class Migration(migrations.Migration):
```

```
    initial = True
```

```

    dependencies = [
        ('auth', '0011_update_proxy_permissions'),
    ]

```

```

    operations = [
        migrations.CreateModel(
            name='User',
            fields=[

```

				ПГУ1.090301.01.001 ПЗ	Лист
					82
Изм.	Лист	№ докум	Подпись Дата		

```

        ('id', models.AutoField(auto_created=True,
primary_key=True, serialize=False, verbose_name='ID')),

        ('password', models.CharField(max_length=128,
verbose_name='password')),

        ('last_login', models.DateTimeField(blank=True,
null=True, verbose_name='last login')),

        ('is_superuser',
models.BooleanField(default=False, help_text='Designates that
this user has all permissions without explicitly assigning
them.', verbose_name='superuser status')),

        ('username',
models.CharField(error_messages={'unique': 'A user with that
username already exists.'}, help_text='Required. 150 characters
or fewer. Letters, digits and @/./+/-/_ only.', max_length=150,
unique=True,
validators=[django.contrib.auth.validators.UnicodeUsernameValida
tor()], verbose_name='username')),

        ('first_name', models.CharField(blank=True,
max_length=30, verbose_name='first name')),

        ('last_name', models.CharField(blank=True,
max_length=150, verbose_name='last name')),

        ('email', models.EmailField(blank=True,
max_length=254, verbose_name='email address')),

        ('is_staff', models.BooleanField(default=False,
help_text='Designates whether the user can log into this admin
site.', verbose_name='staff status')),

        ('is_active', models.BooleanField(default=True,
help_text='Designates whether this user should be treated as
active. Unselect this instead of deleting accounts.',
verbose_name='active')),

        ('date_joined',
models.DateTimeField(default=django.utils.timezone.now,
verbose_name='date joined')),

        ('is_student',
models.BooleanField(default=False)),

        ('is_teacher',
models.BooleanField(default=False)),

        ('groups', models.ManyToManyField(blank=True,
help_text='The groups this user belongs to. A user will get all
permissions granted to each of their groups.',

```

				ПГУ1.090301.01.001 ПЗ	Лист
					83
Изм.	Лист	№ докум	Подпись Дата		

```

related_name='user_set', related_query_name='user',
to='auth.Group', verbose_name='groups')),

        ('user_permissions',
models.ManyToManyField(blank=True, help_text='Specific
permissions for this user.', related_name='user_set',
related_query_name='user', to='auth.Permission',
verbose_name='user permissions')),

    ],

    options={
        'verbose_name': 'user',
        'verbose_name_plural': 'users',
        'abstract': False,
    },

    managers=[
        ('objects',
django.contrib.auth.models.UserManager()),

    ],

),

migrations.CreateModel(
    name='ClassAssignment',
    fields=[
        ('id', models.AutoField(auto_created=True,
primary_key=True, serialize=False, verbose_name='ID')),
        ('created_at',
models.DateTimeField(auto_now=True)),
        ('assignment_name',
models.CharField(max_length=250)),
        ('assignment',
models.FileField(upload_to='assignments')),
    ],
    options={
        'ordering': ['-created_at'],
    },

),

migrations.CreateModel(

```

				ПГУ1.090301.01.001 ПЗ	Лист
					84
Изм.	Лист	№ докум	Подпись Дата		

```

        name='StudentsInClass',

        fields=[

            ('id', models.AutoField(auto_created=True,
primary_key=True, serialize=False, verbose_name='ID')),

            ],

        ),

migrations.CreateModel(

    name='Student',

    fields=[

        ('user',
models.OneToOneField(on_delete=django.db.models.deletion.CASCADE
, primary_key=True, related_name='Student', serialize=False,
to=settings.AUTH_USER_MODEL)),

        ('name', models.CharField(max_length=250)),

        ('roll_no', models.CharField(max_length=50)),

        ('email', models.EmailField(max_length=254)),

        ('phone', models.IntegerField()),

        ('student_profile_pic',
models.ImageField(blank=True,
upload_to='classroom/student_profile_pic')),

        ],

        options={

            'ordering': ['roll_no'],

        },

    ),

migrations.CreateModel(

    name='Teacher',

    fields=[

        ('user',
models.OneToOneField(on_delete=django.db.models.deletion.CASCADE
, primary_key=True, related_name='Teacher', serialize=False,
to=settings.AUTH_USER_MODEL)),

        ('name', models.CharField(max_length=250)),

        ('subject_name',
models.CharField(max_length=250)),

```

```

        ('email', models.EmailField(max_length=254)),
        ('phone', models.IntegerField()),
        ('teacher_profile_pic',
models.ImageField(blank=True,
upload_to='classroom/teacher_profile_pic')),
        ('class_students',
models.ManyToManyField(through='classroom.StudentsInClass',
to='classroom.Student')),
    ],
),
migrations.CreateModel(
    name='SubmitAssignment',
    fields=[
        ('id', models.AutoField(auto_created=True,
primary_key=True, serialize=False, verbose_name='ID')),
        ('created_at',
models.DateTimeField(auto_now=True)),
        ('submit',
models.FileField(upload_to='Submission')),
        ('submitted_assignment',
models.ForeignKey(on_delete=django.db.models.deletion.CASCADE,
related_name='submission_for_assignment',
to='classroom.ClassAssignment')),
        ('student',
models.ForeignKey(on_delete=django.db.models.deletion.CASCADE,
related_name='student_submit', to='classroom.Student')),
        ('teacher',
models.ForeignKey(on_delete=django.db.models.deletion.CASCADE,
related_name='teacher_submit', to='classroom.Teacher')),
    ],
    options={
        'ordering': ['-created_at'],
    },
),
migrations.AddField(
    model_name='studentsinclass',

```

```

        name='student',

field=models.ForeignKey(on_delete=django.db.models.deletion.CASCADE, related_name='user_student_name', to='classroom.Student'),

    ),

    migrations.AddField(

        model_name='studentsinclass',

        name='teacher',

field=models.ForeignKey(on_delete=django.db.models.deletion.CASCADE, related_name='class_teacher', to='classroom.Teacher'),

    ),

    migrations.CreateModel(

        name='StudentMarks',

        fields=[

            ('id', models.AutoField(auto_created=True, primary_key=True, serialize=False, verbose_name='ID')),

            ('subject_name', models.CharField(max_length=250)),

            ('marks_obtained', models.IntegerField()),

            ('maximum_marks', models.IntegerField()),

            ('student', models.ForeignKey(on_delete=django.db.models.deletion.CASCADE, related_name='marks', to='classroom.Student')),

            ('teacher', models.ForeignKey(on_delete=django.db.models.deletion.CASCADE, related_name='given_marks', to='classroom.Teacher')),

        ],

    ),

    migrations.AddField(

        model_name='classassignment',

        name='student',

field=models.ManyToManyField(related_name='student_assignment', to='classroom.Student'),

    ),

```

```

migrations.AddField(
    model_name='classassignment',
    name='teacher',

    field=models.ForeignKey(on_delete=django.db.models.deletion.CASCADE, related_name='teacher_assignment', to='classroom.Teacher'),
),
migrations.AlterUniqueTogether(
    name='studentsinclass',
    unique_together=({'teacher', 'student'}),
),
migrations.CreateModel(
    name='MessageToTeacher',
    fields=[
        ('id', models.AutoField(auto_created=True, primary_key=True, serialize=False, verbose_name='ID')),
        ('created_at', models.DateTimeField(auto_now=True)),
        ('message', models.TextField()),
        ('message_html', models.TextField(editable=False)),
        ('student', models.ForeignKey(on_delete=django.db.models.deletion.CASCADE, related_name='student', to='classroom.Student')),
        ('teacher', models.ForeignKey(on_delete=django.db.models.deletion.CASCADE, related_name='messages', to='classroom.Teacher')),
    ],
    options={
        'ordering': ['-created_at'],
        'unique_together': (('student', 'message')),
    },
),
migrations.CreateModel(
    name='ClassNotice',

```

				ПГУ1.090301.01.001 ПЗ	Лист
					88
Изм.	Лист	№ докум	Подпись Дата		

```

        fields=[
            ('id', models.AutoField(auto_created=True,
primary_key=True, serialize=False, verbose_name='ID')),
            ('created_at',
models.DateTimeField(auto_now=True)),
            ('message', models.TextField()),
            ('message_html',
models.TextField(editable=False)),
            ('students',
models.ManyToManyField(related_name='class_notice',
to='classroom.Student')),
            ('teacher',
models.ForeignKey(on_delete=django.db.models.deletion.CASCADE,
related_name='teacher', to='classroom.Teacher')),
        ],
        options={
            'ordering': ['-created_at'],
            'unique_together': {('teacher', 'message')},
        },
    ),
]

```

Файл base.css

```

.login-page body{
    margin: 0;
    padding: 0;
    font-family: sans-serif;
}

nav{
    width: 100%;
    background: #00316b;
    margin-bottom: 0;
}

ul{

```

				ПГУ1.090301.01.001 ПЗ	Лист
					89
Изм.	Лист	№ докум	Подпись Дата		

```

        width: 100%;
        margin: 0 auto;
        padding: 0;
        display: none;
        margin-bottom: 0;
        background: #00316b;
    }

    ul li{
        list-style: none;
        display: block;
        padding: 20px;
        text-align: center;
        margin: 0;
    }

    ul li:hover{
        background: #e91e63;
        display: block;
        margin-bottom: 0;
    }

    ul li a
    {
        color: #fff;
        text-decoration: none;
        display: block;
    }

    .toggle{
        width: 100%;
        padding: 10px;
        background: #00316b;
        text-align: right;
    }

```

				ПГУ1.090301.01.001 ПЗ	Лист
					90
Изм.	Лист	№ докум	Подпись Дата		

```

    box-sizing: border-box;
    color: #fff;
    font-size: 30px;
    display: block;
}

.toggle a{
    text-align: left;
    display: block;
    color: #fff;
    text-decoration: none;
    margin-top: 5px;
    margin-bottom: -25px;
    padding: 0px;
}

.active{
    display: block;
}

/* INDEX HOME PAGE #####
*/

.banner{
    margin: 0;
    position: relative;
    width: 100%;
    height: 100vh;
    background: #3475ca;
    display: flex;
    justify-content: center;
    align-items: center;

```

				ПГУ1.090301.01.001 ПЗ	Лист
					91
Изм.	Лист	№ докум	Подпись Дата		

```

}

.banner img{
    position: absolute;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    object-fit: cover;
    opacity: 0.25;
    background-attachment: fixed;
}

.banner .content{
    position: relative;
    max-width: 750px;
    text-align: center;
    color: white;
    font-weight: 1000;
}

.banner .content{
    font-size: 40px;
}

.about{
    position: relative;
    background: #00316b;
    color: white;
    text-align: center;
    padding: 100px;
    letter-spacing: 2px;
    font-weight: 600;
}

```

				ПГУ1.090301.01.001 ПЗ	Лист
					92
Изм.	Лист	№ докум	Подпись Дата		

```

        font-size: 25px;
    }

    .use-content{
        background: #e91e63;
        padding: 20px;
        text-align: center;
        color: white;
        font-weight: 600;
    }

    .use-content h1{
        font-weight: 700;
        font-size: 50px;
    }

    .research{
        background: #00316b;
        padding:5%;
        text-align: center;
        color: white;
        font-weight: 600;
    }

    .research img{
        width:100%;
        height: 100%;
    }

    .research .digital-content{
        padding: 30px;

```

				ПГУ1.090301.01.001 ПЗ	Лист
					93
Изм.	Лист	№ докум	Подпись Дата		

```

padding-top: 80px;
text-align: center;
font-weight: 700;
font-size: 30px;
}

.box {
margin: 10%;
margin-left: 15%;
position: relative;
width: 400px;
height: 500px;
background: #000;
box-shadow: 0 30px 30px rgba(0,0,0,.5);
}

@media only screen and (max-width: 600px){
    .box{
        margin: 5%;
    }
}

.box:hover{
    transform: scale(1.1);
    transition: 0.5s;
}

.box .imgBx {
position: relative;
top: 0;
left: 0;

```

				ПГУ1.090301.01.001 ПЗ	Лист
					94
Изм.	Лист	№ докум	Подпись Дата		

```

    width: 100%;
    height: 100%;
}

.box .imgBx img {
    position: absolute;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    object-fit: cover;
    transition: 0.5s;
}

.box:hover .imgBx img {
    opacity: 0;
}

.box .content {
    position: absolute;
    bottom: 20px;
    left: 10%;
    width: 80%;
    height: 60px;
    background: #00316b;
    transition: 0.5s;
    overflow: hidden;
    color: white;
    padding: 15px;
    box-sizing: border-box;
}

```

				ПГУ1.090301.01.001 ПЗ	Лист
					95
Изм.	Лист	№ докум	Подпись Дата		

```

.box:hover .content {
    width: 100%;
    height: 100%;
    bottom: 0;
    left: 0;
}

.box .content h3 {
    margin: 0;
    padding: 5px;
    font-size: 30px;
    font-weight: 700;
}

.box .content li {
    margin: 10px 10px;
    padding: 0;
    opacity: 0;
    line-height: 1.2em;
    transition: 0.5s;
    text-align: justify;
}

.box:hover .content li {
    opacity: 1;
    transition-delay: 0.5s;
}

.social{
    background: #e91e63;

```

				ПГУ1.090301.01.001 ПЗ	Лист
					96
Изм.	Лист	№ докум	Подпись Дата		

```

padding:5%;
color: white;
text-align: center;
}

.social i{
margin-top: 10px;
color: white;
font-size: 40px;

}

.social .social-icon{
display: inline-block;
width: 90px;
height: 90px;
padding: 10px;
margin: 10px;
background: #00316b;
border-radius: 30%;
box-shadow: 0 5px 15px -5px #00000070
}

.social-icon:hover{
transform: scale(1.2);
transition: 0.5s;
}

.social-icon:hover i{
transform: scale(1.5);
transition: 0.5s;
}

```

				ПГУ1.090301.01.001 ПЗ	Лист
					97
Изм.	Лист	№ докум	Подпись Дата		

```

        color: #e91e63;
    }

    /* SCROOL BAR ##### */

    /* width
    ::-webkit-scrollbar {
        width: 20px;
    }

    /* Track
    ::-webkit-scrollbar-track {
        box-shadow: inset 0 0 5px grey;
        border-radius: 10px;
    }

    /* Handle
    ::-webkit-scrollbar-thumb {
        background: red;
        border-radius: 10px;
    }

    /* Handle on hover
    ::-webkit-scrollbar-thumb:hover{
        background: #e91e63;
    }

    /* LOGIN PAGE ##### */

    .login-page{
        position: absolute;

```

				ПГУ1.090301.01.001 ПЗ	Лист
					98
Изм.	Лист	№ докум	Подпись Дата		

```

width: 100%;
height: 120vh;
background: #3475ca;
display: flex;
justify-content: center;
align-items: center;
}

.login-page h1{
font-size: 40px;
font-weight: 600;
}

.login-page img{
position: absolute;
top: 0;
left: 0;
width: 100%;
height: 100%;
object-fit: cover;
opacity: 0.25;
background-attachment: fixed;
overflow: hidden;
}

.login-form {
margin-top: 30px;
height: 600px;
width: 400px;
background: #e91e63;
color: white;

```

				ПГУ1.090301.01.001 ПЗ	Лист
					99
Изм.	Лист	№ докум	Подпись Дата		

```
padding: 70px 100px;
position: absolute;

}
```

```
.avatar img{
height:100px;
width:100px;
border-radius: 50%;
position: absolute;
top: -50px;
left: 37%;
opacity: 1;
overflow: auto;
opacity: 1;
}
```

```
.login-form h1 {
margin: 0;
padding: 0;
text-align: center;
font-size: 40px;
font-weight: 700;
opacity: 1;
}
```

```
.login-form p{
margin-top: 30px;
padding:0;
font-size: 20px;
font-weight: bold;
```

				ПГУ1.090301.01.001 ПЗ	Лист
					100
Изм.	Лист	№ докум	Подпись Дата		

```

        opacity: 1;
    }

    .login-form input{
        width: 100%;
        margin-bottom: 20px;
        color: white;
        opacity: 1;
    }

    .login-form input[type="text"], input[type="password"]
    {
        border: none;
        border-bottom: 1px solid white;
        background: transparent;
        outline: none;
        height: 40px;
        color: white;
        font-size: 20px;
    }

    .answer{
        border: none;
        border-bottom: 1px solid white;
        background: transparent;
        outline: none;
        height: 40px;
        color: white;
        font-size: 20px;
    }

    .answer:hover{

```

				ПГУ1.090301.01.001 ПЗ	Лист
					101
Изм.	Лист	№ докум	Подпись Дата		

```

    transform: scale(1.1);
    transition: 0.5;
    border-bottom: 1px solid #00316b;
}

.login-form input[type="text"]:hover,
input[type="password"]:hover{
    transform: scale(1.1);
    transition: 0.5;
    border-bottom: 1px solid #00316b;
}

.login-form input[type="submit"]{
    border: 1px solid #00316b;
    outline: none;
    height: 40px;
    background: #00316b;
    color: white;
    font-size: 18px;
}

.login-form input[type="submit"]:hover{
    cursor: pointer;
    transform: scale(1.2);
    transition: 0.5;
}

.login-form a{
    text-align: center;
    text-decoration: none;
    font-size: 15px;

```

				ПГУ1.090301.01.001 ПЗ	Лист
					102
Изм.	Лист	№ докум	Подпись Дата		

```

    line-height: 20px;
    color: white;
    opacity: 1;
}

.login-form a:hover{
    color: #00316b;
    transform: scale(1.2);
    transition: 0.5s;
}

/*STUDENT SIGN UP
***** /

.student-signup-form{
    margin-top: 2px;
    background: #e91e63;
    padding: 20px;
    padding-top: 3%;
    position: absolute;
    text-align: center;
    color: white;
}

.student-signup-form h1,h3{
    margin: 10px;
    padding: 0;
    text-align: center;
    font-size: 40px;
    font-weight: 300;
    opacity: 1;
}

```

				ПГУ1.090301.01.001 ПЗ	Лист
					103
Изм.	Лист	№ докум	Подпись Дата		

```
.student-signup-form label{
  margin: 5px;
  margin-right: 10px;
  padding:0;
  font-size: 20px;
  font-weight: bold;
  opacity: 1;
}
```

```
.student-signup-form input[type="text"]{
  color: white;
}
```

```
.student-signup-form input[type="submit"]{
  border: 1px solid #00316b;
  outline: none;
  height: 40px;
  background: #00316b;
  color: white;
  font-size: 18px;
  margin: 15px;
}
```

```
.student-signup-form input[type="submit"]:hover{
  cursor: pointer;
  transform: scale(1.2);
  transition: 0.5;
}
```

				ПГУ1.090301.01.001 ПЗ	Лист
					104
Изм.	Лист	№ докум	Подпись Дата		

```

.student-signup-form a{
    text-align: center;
    text-decoration: none;
    font-size: 15px;
    line-height: 20px;
    color: white;
    opacity: 1;
}

.student-signup-form a:hover{
    color: #00316b;
    transform: scale(1.2);
    transition: 0.5s;
}

/* SIGN UP PAGE
*****/

.login-page .signup{
    background: #e91e63;
    height: 500px;
    width: 500px;
    margin-top: 10%;
    margin-left: 10%;
    transform: translate(-50px,-50px);
    color: #fff;
    text-align: center;
    border-radius: 20%;
    box-shadow: 0 30px 30px rgba(0,0,0,.5);
    padding: 30px;
}

```

```

.login-page .signup h1{
    margin-top: 15%;
    font-size: 50px;
    font-weight: 600;
}

.login-page .signup p{
    display: block;
    height: 50px;
    color: #fff;
    background: #00316b;
    margin: 20px;
    padding: 5px;
    box-shadow: 0 30px 30px rgba(0,0,0,.5);
    font-size: 30px;
}

.login-page .signup p:hover{
    transform: scale(1.2);
    transition: 0.5s;
}

.login-page .signup a{
    display: block;
    margin-top: 5px;
    text-decoration: none;
    color: #fff;
    font-size: 15px;
    font-weight: 400;
}

```

```

.notice{
    margin-top: 50px;
}

.notice .notice-details{
    background: #c7c7c5;
    padding: 20px;
    color: black;
}

.notice .notice-img{
    width:75px;
    height:75px;
    border-radius: 50%;
}

```

Файл newbase.css

```

.profile{
    margin:0;
    padding: 0;
    position: absolute;
    width:100%;
    height: 100vh;
    display:flex;
    justify-content: center;
    align-items: center;
}

.profile img{
    position: absolute;
    margin: 0;
    width: 100%;
}

```

				ПГУ1.090301.01.001 ПЗ	Лист
					107
Изм.	Лист	№ докум	Подпись Дата		

```

height: 100%;
object-fit: cover;
background-attachment:
}

.profile-box{
width: 600px;
margin-top: 20px;
background: #e91e63;
position: absolute;
padding: 15px;
border-radius: 10%;
box-shadow: 0 5px 15px -5px #00000070
}

@media only screen and (max-width: 600px){
.profile-box{
font-size: 15px;
}
}

.profile-box img{
width: 200px;
height: 200px;
border-radius: 50%;
top: -10%;
left: 35%;
box-shadow: 0 5px 15px -5px #00000070
}

.profile-details{

```

				ПГУ1.090301.01.001 ПЗ	Лист
					108
Изм.	Лист	№ докум	Подпись Дата		

```

    color: white;
    margin-top: 30%;
    margin-left: 10%;
}

.profile-details h1{
    font-size: 40px;
    font-weight: 600;
    text-align: center;
}

.profile-details p{
    font-size: 30px;
    font-weight: 400;
}

.profile-details .edit-button{
    margin: 20px;
    margin-left: 25%;
    height: 70px;
    width: 200px;
    text-align: center;
    background: #00316b;
    text-decoration: none;
    color: white;
    padding: 5px;
}

.student-list{
    margin: 0;
    padding: 0;
}

```

				ПГУ1.090301.01.001 ПЗ	Лист
					109
Изм.	Лист	№ докум	Подпись Дата		

```

background: url("/static/images/list-background.PNG");
background-size:auto;
background-repeat: repeat;
width: 100%;
padding-bottom: 600px;
padding-top: 5%;
color: black;
}

```

```

.student-list .student-box{
margin: 10%;
position: relative;
width: 100%;
height: 400px;
background: #000;
}

```

```

.student-list .student-box:hover{
transform: scale(1.2);
transition: 0.5s;
}

```

```

.student-list .student-box .imgBx{
position: relative;
top: 0;
left: 0;
width: 100%;
height: 100%;
}

```

```

.student-list .student-box .imgBx img{
position: absolute;

```

				ПГУ1.090301.01.001 ПЗ	Лист
					110
Изм.	Лист	№ докум	Подпись Дата		

```

top: 0;
left: 0;
width: 100%;
height: 100%;
object-fit: cover;
transition: 0.5s;
}

.student-list .student-box:hover .imgBx img{
  opacity: 0;
}

.student-list .student-box .content{
  position: absolute;
  bottom: 20px;
  left: 10%;
  width: 80%;
  height: 60px;
  background: #00316b;
  transition: 0.5s;
  overflow: hidden;
  color: white;
  padding: 15px;
  box-sizing: border-box;
}

.student-list .student-box:hover .content{
  width: 100%;
  height: 100%;
  bottom: 0;
  left: 0;

```

				ПГУ1.090301.01.001 ПЗ	Лист
					111
Изм.	Лист	№ докум	Подпись Дата		

```

}

.student-list .student-box .content h3{
    margin: 0;
    padding: 5px;
    font-size: 30px;
    font-weight: 700;
}

.student-list .student-box .content p{
    margin: 30px 10px;
    padding: 0;
    opacity: 0;
    line-height: 1.2em;
    transition: 0.5s;
    text-align: justify;
    font-size: 15px;
}

.student-list .student-box:hover .content p{
    opacity: 1;
    transition-delay: 0.5s;
}

.student-list .student-box .content .add-button{
    margin: 20px;
    padding: 10px;
    margin-left: 0;
    text-decoration: none;
    color: white;
    background: #e91e63;

```

				ПГУ1.090301.01.001 ПЗ	Лист
					112
Изм.	Лист	№ докум	Подпись Дата		

```

        text-align: center;

    }

    Файл searchbar.css

input {
    outline: none;
}

input[type=search] {
    -webkit-appearance: textfield;
    -webkit-box-sizing: content-box;
    font-family: inherit;
    font-size: 100%;
}

input::-webkit-search-decoration,
input::-webkit-search-cancel-button {
    display: none;
}

input[type=search] {
    background: #ededed
url(https://static.tumblr.com/ftv85bp/MIXmud4tx/search-icon.png)
no-repeat 9px center;

    border: solid 1px #ccc;
    padding: 9px 10px 9px 32px;
    width: 55px;

    -webkit-border-radius: 10em;
    -moz-border-radius: 10em;
    border-radius: 10em;

    -webkit-transition: all .5s;

```

				ПГУ1.090301.01.001 ПЗ	Лист
					113
Изм.	Лист	№ докум	Подпись Дата		

```

        -moz-transition: all .5s;
        transition: all .5s;
    }
    input[type=search]:focus {
        width: 130px;
        background-color: #fff;
        border-color: #66CC75;

        -webkit-box-shadow: 0 0 5px rgba(109,207,246,.5);
        -moz-box-shadow: 0 0 5px rgba(109,207,246,.5);
        box-shadow: 0 0 5px rgba(109,207,246,.5);
    }

```

```

input:-moz-placeholder {
    color: #999;
}
input::-webkit-input-placeholder {
    color: #999;
}

```

Файл add_marks.html

```
<!--
```

Template for adding marks. This template can only be accessed by teachers and it contains form where teachers can add marks for specific student.

```
-->
```

```

{% extends 'classroom/base.html' %}
{% load bootstrap4 %}
{% block content %}
<div class="container">
<div class="jumbotron">

```

				ПГУ1.090301.01.001 ПЗ	Лист
					114
Изм.	Лист	№ докум	Подпись Дата		

```

    {% if marks_given %}

    <h1>Оценка успешно выставлена</h1>

    {% else %}

    <h1>Добавить оценку {{ student.name }}</h1>

    <form method="post">

        {% csrf_token %}

        {% bootstrap_form form %}

        <input type="submit" class="btn btn-danger" name=""
value="Добавить оценку">

    </form>

    {% endif %}

</div>

</div>

{% endblock %}

```

Файл assignment_delete.html

```

{% extends 'classroom/base.html' %}

{% block content %}

<div class="container">

    <div class="jumbotron">

        <h1>Удалить {{ object.assignment_name }}?</h1>

        <form method='POST'>

            {% csrf_token %}

            <p>Вы уверены что хотите удалить задачу?</p>

            <input class="btn btn-primary" type='submit' value='Да'>

            <a class="btn btn-danger" href="{% url
'classroom:assignment_list' %}">Отменить</a>

        </form>

    </div>

</div>

</div>

```

```
{% endblock %}
```

Файл assignment_list.html

```
{% extends 'classroom/base.html' %}
```

```
{% block content %}
```

```
<div class="container">
```

```
  <div class="jumbotron">
```

```
{% if teacher.teacher_assignment.count == 0 %}
```

```
  <h2>Задачи не найдены</h2>
```

```
{% else %}
```

```
  <table class="table">
```

```
    <thead class="thead-dark">
```

```
      <tr>
```

```
        <th scope="col">#</th>
```

```
        <th scope="col">Задача</th>
```

```
        <th scope="col">Дата загрузки</th>
```

```
        <th scope="col">Скачать</th>
```

```
        <th scope="col">Обновить</th>
```

```
        <th scope="col">Удалить</th>
```

```
      </tr>
```

```
    </thead>
```

```
    <tbody>
```

```
      {% for assignment in teacher.teacher_assignment.all %}
```

```
        <tr>
```

```
          <th scope="row">{{ forloop.counter }}</th>
```

```
          <td>{{ assignment.assignment_name }}</td>
```

```
          <td>{{ assignment.created_at }}</td>
```

```
          <td><a href="{{ assignment.assignment.url }}"
class="btn btn-primary" target="_blank">Скачать</a></td>
```

```
          <td><a href="{% url 'classroom:update_assignment'
id=assignment.id %}" class="btn btn-primary">Обновить</a></td>
```

				ПГУ1.090301.01.001 ПЗ	Лист
					116
Изм.	Лист	№ докум	Подпись Дата		

```

        <td><a href="{% url 'classroom:assignment_delete'
id=assignment.id %}" class="btn btn-danger">Удалить</a></td>

    </tr>

    {% endfor %}

</tbody>

</table>

{% endif %}

</div>

</div>

{% endblock %}

```

Файл base.html

```

<!DOCTYPE html>

{% load static %}


<html lang="ru" dir="ltr"> <head> <meta charset="utf-8">
<title>Управление студентами</title> <!-- Latest compiled and
minified CSS --> <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.0/css/bootst
rap.min.css"> <!-- jQuery library --> <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.m
in.js"></script> <!-- Popper JS --> <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd
/popper.min.js"></script> <!-- Latest compiled JavaScript -->
<script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.0/js/bootstra
p.min.js"></script>

<link href="https://fonts.googleapis.com/css?family=Montserrat"
rel="stylesheet">


<link rel="stylesheet" href="{% static 'css/base.css' %}">

<link rel="stylesheet" href="{% static 'css/searchbar.css' %}">

<link rel="stylesheet" href="{% static 'css/newbase.css' %}">

<link rel="stylesheet"
href="https://pro.fontawesome.com/releases/v5.10.0/css/all.css">

<link
href="https://fonts.googleapis.com/css2?family=Source+Sans+Pro&d
isplay=swap" rel="stylesheet">

```

				ПГУ1.090301.01.001 ПЗ	Лист
					117
Изм.	Лист	№ докум	Подпись Дата		

```

<link
href="https://fonts.googleapis.com/css2?family=Permanent+Marker&
family=Source+Sans+Pro&display=swap" rel="stylesheet">

{% load bootstrap4 %}
{% bootstrap_css %}
{% bootstrap_javascript jquery='full' %}
</head> <body>

<nav>

    <div class="toggle">

        <a href="{% url 'home' %}">Управление классом</a>

        <i class="fa fa-bars menu"></i>

    </div>

</nav>

<div class="sidebar">

    <ul>

        {% if user.is_authenticated %}

            {% if user.is_teacher%}

                <li> <a href="{% url 'classroom:upload_assignment'
%}">Загрузить задание</a> </li>

                <li> <a href="{% url 'classroom:assignment_list'
%}">Список заданий</a> </li>

                <li> <a href="{% url 'classroom:submit_list'
%}">Просмотреть Задания</a> </li>

                <li> <a href="{% url 'classroom:students_list' %}">Список
всех студентов</a> </li>

                <li> <a href="{% url
'classroom:class_student_list'%}">Список студентов класса</a>
</li>

                <li> <a href="{% url 'classroom:write_notice' %}">Написать
уведомление</a> </li>

                <li> <a href="{% url 'classroom:messages_list' pk=user.pk
%}">Входящие сообщения</a> </li>

                <li> <a href="{% url 'classroom:teacher_detail' pk=user.pk
%}">Профиль</a> </li>

```

				ПГУ1.090301.01.001 ПЗ	Лист
					118
Изм.	Лист	№ докум	Подпись Дата		

```

        {% else %}

        <li> <a href="{% url 'classroom:all_marks_list' pk=user.pk
%}">Оценки</a> </li>

        <li> <a href="{% url 'classroom:class_notice' pk=user.pk
%}">Уведомление класса</a> </li>

        <li> <a href="{% url 'classroom:class_assignment'
%}">Задание класса</a> </li>

        <li> <a href="{% url 'classroom:teachers_list' %}">Список
учителей</a> </li>

        <li> <a href="{% url 'classroom:student_detail' pk=user.pk
%}">Профиль</a> </li>

        {% endif %}

        <li> <a href="{% url 'classroom:change_password'
%}">Изменить пароль</a> </li>

        <li> <a href="{% url 'classroom:logout' %}">Выход</a>
</li>

        {% else %}

        <li><a class="head-left" href="{% url 'classroom:login'
%}">Логин</a></li>

        <li><a class="head-left" href="{% url 'classroom:signup'
%}">Регистрация</a></li>

        {% endif %}

    </ul>
</div>

{% block content %}

{% endblock %}

<script src="https://code.jquery.com/jquery-
3.5.1.js"></script>

<script type="text/javascript">

    $(document).ready(function(){

        $('.menu').click(function(){

            $('ul').toggleClass('active');

```

```

    })

  })

</script>

</body>

</html>

```

Файл change_password.html

```

{% extends "classroom/base.html" %} {% load bootstrap4 %} {%
block content %}

<div class="container">

  <div class="jumbotron">

    <h1>Изменение пароля.</h1>

    <form enctype="multipart/form-data" method="POST">

      {% csrf_token %}

      {% bootstrap_form form %}

      <input class="btn btn-primary" type="submit" name=""
value="Изменить пароль" />

    </form>

  </div>

</div>

{% endblock %}

```

Файл class_assignment.html

```

{% extends 'classroom/base.html' %} {% block content %}

<div class="container">

  <div class="jumbotron">

    {% if student.student_assignment.count == 0 %}

      <h2>До сих пор нет заданий.</h2>

      {% else %}

        <table class="table">

          <thead class="thead-dark">

```

```

        <tr>

            <th scope="col">#</th>

            <th scope="col">Название задания</th>

            <th scope="col">Загрузил</th>

            <th scope="col">Дата загрузки</th>

            <th scope="col">Скачать</th>

            <th scope="col">Статус</th>

        </tr>

    </thead>

    <tbody>

        {% for assignment in
student.student_assignment.all %}

            <tr>

                <th scope="row">{{ forloop.counter }}</th>

                <td>{{ assignment.assignment_name }}</td>

                <td>{{ assignment.teacher }}</td>

                <td>{{ assignment.created_at }}</td>

                <td><a href="{{ assignment.assignment.url
}}" class="btn btn-primary" target="_blank">Скачать</a></td>

                {% if assignment in assignment_list %}

                    <td>Отправлено</td>

                    {% else %}

                        <td><a href="{% url
'classroom:submit_assignment' id=assignment.id %}" class="btn
btn-primary">Отправить</a></td>

                    {% endif %}

            </tr>

        {% endfor %}

    </tbody>

</table>

{% endif %}

</div>

</div>

```

```
{% endblock %}
```

Файл class_notice_list.html

```
{% extends 'classroom/base.html' %} {% load static %} {% block  
content %}
```

```
<div class="container">
```

```
    {% if student.class_notice.count == 0 %}
```

```
    <h2>Пока что нет объявлений.</h2>
```

```
    {% else %} {% for notice in student.class_notice.all %}
```

```
    <div class="notice">
```

```
        <div class="row">
```

```
            <div class="col-md-1">
```

```
                {% if notice.teacher.teacher_profile_pic %}
```

```
                
```

```
                {% else %}
```

```
                 {% endif %}
```

```
            </div>
```

```
            <div class="col-md-11">
```

```
                <div class="notice-details">
```

```
                    <div class="row">
```

```
                        <div class="col-md-3"><h5>От: {{  
notice.teacher.name }}</h5></div>
```

```
                        <div class="col-md-3"><h5>Предмет: {{  
notice.teacher.subject_name }}</h5></div>
```

```
                        <div class="col-md-3"><h5>{{  
notice.created_at }}</h5></div>
```

```
                    </div>
```

```
                    <hr />
```

```
                    <h2>{{ notice.message_html|safe }}</h2>
```

```
                </div>
```

				ПГУ1.090301.01.001 ПЗ	Лист
					122
Изм.	Лист	№ докум	Подпись Дата		

```

        </div>

    </div>

</div>

{% endfor %} {% endif %}

</div>

{% endblock %}

```

Файл class_students_list.html

```

{% extends "classroom/base.html" %} {% load static %} {% block
content %}

<div class="student-list">

    <div class="container">

        {% if class_students_list.count == 0 %}

            <h1>Нет студентов в классе.</h1>

        {% else %}

            <form method="GET" action="{% url
'classroom:class_student_list' %}">

                <p>Поиск студента: <input type="search" name="q"
placeholder="Поиск" /></p>

            </form>

            <div class="row">

                {% for student in class_students_list %}

                    <div class="col-6 col-md-4">

                        <div class="student-box">

                            <div class="imgBx">

                                {% if student.student_profile_pic %}
                                 {%
                                else %} {% if forloop.counter0|divisibleby:"2" %}  {% else

                                    %}  {% endif %} {% endif %}

                                </div>

                                <div class="content">

                                    <h3>{{ student.name }}</h3>

```

```

        <p><i class="far fa-user"></i> Имя
пользователя: {{ student.user.username }}</p>

        <p><i class="fal fa-user-tag"></i> Роль
№.: {{ student.roll_no }}</p>

        <p><i class="far fa-phone"></i> Телефон:
{{ student.phone }}</p>

        <p><i class="far fa-envelope-open-
text"></i> Email: {{ student.email }}</p>

        <a href="{% url
'classroom:student_marks_list' pk=student.pk %}"><p class="add-
button">Оценки</p></a>

        </div>

    </div>

</div>

    {% endfor %}

</div>

{% endif %}

</div>

{% endblock %}

```

Файл index.html

```

{% extends 'classroom/base.html' %} {% load static %} {% block
content %}

{% if messages %}

<div class="alert alert-primary alert-dismissible fade show"
role="alert">

    {% for message in messages %}

    <div class="alert {% if message.tags %} alert-{{ message.tags
}}{% endif %}">

        {{ message|safe }}

    </div>

    {% endfor %}


```

				ПГУ1.090301.01.001 ПЗ	Лист
					124
Изм.	Лист	№ докум	Подпись Дата		

```

        <button type="button" class="close" data-dismiss="alert" aria-
label="Close">

        <span aria-hidden="true">&times;</span>

    </button>
</div> {% endif %}
<div class="banner">
    
    <div class="content">
        <h1>Class Manager</h1>
        <p>Портал студентов и учителей</p>
    </div>
</div>
<div class="about">
    <h1>О нас</h1>
    <p>Class Manager – это портал студентов и учителей с
различными функциями классной комнаты, такими как загрузка
заданий, сдача работ, объявления о классе и многое другое.
Студенты и учителя могут войти на портал и получить
интерактивный опыт классной комнаты онлайн.

    Одной из самых мощных функций Class Manager является
возможность управление студентами. Учитель легко может управлять
своими студентами и держать их в курсе всех дел в классе.</p>
</div>

<div class="use-content">
    <h1>Наши услуги</h1>
    <div class="row">
        <div class="col-12 col-md-6">
            <div class="box">
                <div class="imgBx">
                    
                </div>
                <div class="content">
                    <h3>Для учителя</h3>

```

```

        <ol>

            <li>Учителя могут добавлять студентов в свой класс
            через опцию «Добавить студента».</li>

            <li>Добавив студентов в свой класс, учитель может
            давать или править оценки своих студентов.</li>

            <li>Учителя также могут писать объявления, которые
            будут отправлены всем студентам в их классе.</li>

            <li>Учителя могут загружать задания через опцию
            «Загрузить задания».</li>

            <li>Учитель может проверять сдачу заданий своих
            студентов и давать им оценки соответственно.</li>

            <li>Студенты могут напрямую отправлять сообщения
            учителям, и их сообщения можно увидеть во входящих.</li>

            <li>Учитель также может просматривать все оценки,
            которые он дал студенту, обратившись к его профилю оценок, и
            может изменять его оценки.</li>

            <li>Пользователь может добавлять изображение профиля
            через опцию «Изменить профиль».</li>

        </ol>

    </div>

</div>

</div>

<div class="col-12 col-md-6">

    <div class="box">

        <div class="imgBx">

        </div>

        <div class="content">

            <h3>Для студента</h3>

            <ol>

                <li>Студенты могут входить в портал и быть в курсе
                всех происходящих в классе.</li>

                <li>Студенты могут видеть объявления, которые дали
                их учителя, в опции «Объявления класса».</li>

```

Студенты могут загружать свои задания в опции «Задания» и также могут их отправлять. (Однако после отправки задания его нельзя будет изменить.)

Студенты могут связаться с любым учителем через опцию сообщения в списке «Все учителя».

Студенты могут проверять свои оценки в опции «Оценки».

Пользователь может изменять изображение профиля через опцию «Изменить профиль».

</div>

</div>

</div>

</div>

</div>

<div class="research">

<div class="row">

<div class="col-md-4">

</div>

<div class="col-md-8">

<div class="digital-content">

<h2>Цифровое обучение</h2>

<p>Мы провели опрос более 16 000 учителей и администраторов, чтобы лучше понимать состояние цифрового обучения. Class Manager предоставляет все необходимые возможности для усовершенствования цифрового обучения.</p>

</div>

</div>

</div>

</div>

				ПГУ1.090301.01.001 ПЗ	Лист
					127
Изм.	Лист	№ докум	Подпись Дата		

```

<div class="social">

    <h1>Готовы продвигать возможности в обучении?</h1>

    <h3>Тогда присоединитесь к нам в социальных сетях.</h3>

    <a class="social-icon" href="#"><i class="fab fa-facebook-
f"></i></a>

    <a class="social-icon" href="#"><i class="fab fa-
twitter"></i></a>

    <a class="social-icon" href="#"><i class="fab fa-
instagram"></i></a>

    <a class="social-icon" href="#"><i class="fab fa-
google"></i></a>

    <a class="social-icon" href="#"><i class="fab fa-linkedin-
in"></i></a>

</div>

{% endblock %}

```

Файл login.html

```

{% extends 'classroom/base.html' %} {% load static %} {% block
content %}

<div class="login-page">

    <div class="login-form">

        <div class="avatar"></div>

        {% if messages %}

            <div class="alert alert-primary alert-dismissible fade
show" role="alert">

                {% for message in messages %}

                    <div class="alert {% if message.tags %} alert-{{
message.tags }}{% endif %}">{{ message|safe }}</div>

                    {% endfor %} <button type="button" class="close"
data-dismiss="alert" aria-label="Close"><span aria-
hidden="true">&times;</span></button>

                </div>

            {% endif %}


```

				ПГУ1.090301.01.001 ПЗ	Лист
					128
Изм.	Лист	№ докум	Подпись Дата		

```

        <h1>Вход в систему</h1>

        <form action="{% url 'classroom:login' %}"
method="post">

            {% csrf_token %}

            <p>Имя пользователя:</p>

            <input type="text" name="username"
placeholder="Введите ваше имя пользователя" /> <br />

            <br />

            <p>Пароль:</p>

            <input type="password" name="password"
placeholder="Введите пароль" /> <br />

            <br />

            <input class="btn btn-primary" type="submit" name=""
value="Войти" />

        </form>

        <a href="{% url 'classroom:signup' %}">Еще не
зарегистрированы?</a>

    </div>
</div>
{% endblock %}

```

Файл messages_list.html

```

{% extends 'classroom/base.html' %} {% load static %} {% block
content %}

<div class="container">

    {% if teacher.messages.count == 0 %}

    <h2>Еще нет сообщений.</h2>

    {% else %} {% for mssg in teacher.messages.all %}

    <div class="notice">

        <div class="row">

            <div class="col-md-1">

                {% if mssg.student.student_profile_pic %}

```

				ПГУ1.090301.01.001 ПЗ	Лист
					129
Изм.	Лист	№ докум	Подпись Дата		

```

        {% else %}

        {% endif %}

    </div>

    <div class="col-md-11">

        <div class="notice-details">

            <div class="row">

                <div class="col-md-3"><h5>От: {{
mssg.student.name }}</h5></div>

                <div class="col-md-3"><h5>{{
mssg.created_at }}</h5></div>

            </div>

            <hr />

            <h2>{{ mssg.message_html|safe }}</h2>

        </div>

    </div>

</div>

</div>

{% endfor %} {% endif %}

</div>

{% endblock %}

```

Файл signup.html

```

{% extends 'classroom/base.html' %} {% load static %} {% block
content %}

<div class="login-page">

    <div class="signup">

        <h1>Регистрация</h1>

        <br />

        <a href="{% url 'classroom:StudentSignUp' %}"><p>Я
студент</p> </a> <br />

```

				ПГУ1.090301.01.001 ПЗ	Лист
					130
Изм.	Лист	№ докум	Подпись Дата		

```
        <a href="{% url 'classroom:TeacherSignUp' %}"><p>Я
учитель</p> </a>
```

```
    </div>
```

```
</div>
```

```
{% endblock %}
```

Файл student_added.html

```
{% extends 'classroom/base.html' %} {% block content %}
```

```
<h1>Студент успешно добавлен в класс.</h1>
```

```
<h3>Перейдите к списку всех студентов, чтобы добавить еще
студентов.</h3>
```

```
{% endblock %}
```

Файл student_allmarks_list.html

```
{% extends "classroom/base.html" %} {% block content %}
```

```
<div class="container">
```

```
    <div class="jumbotron">
```

```
        {% if student.marks.count == 0 %}
```

```
        <h2>Оценки еще не опубликованы!</h2>
```

```
        {% else %}
```

```
        <table class="table">
```

```
            <thead class="thead-dark">
```

```
                <tr>
```

```
                    <th scope="col">№</th>
```

```
                    <th scope="col">Предмет</th>
```

```
                    <th scope="col">Имя учителя</th>
```

```
                    <th scope="col">Полученные оценки</th>
```

```
                    <th scope="col">Максимальный балл</th>
```

```
                </tr>
```

```
            </thead>
```

```
            <tbody>
```

```
                {% for object in student.marks.all %}
```

				ПГУ1.090301.01.001 ПЗ	Лист
					131
Изм.	Лист	№ докум	Подпись Дата		

```

        <tr>

            <th scope="row">{{ forloop.counter }}</th>
            <th>{{ object.subject_name }}</th>
            <th>{{ object.teacher.name }}</th>
            <th>{{ object.marks_obtained }}</th>
            <th>{{ object.maximum_marks }}</th>

        </tr>

        {% endfor %}

    </tbody>

</table>

{% endif %}

</div>

</div>

{% endblock %}

```

Файл student_detail_page.html

```

{% extends 'classroom/base.html' %} {% load static %} {% block
content %}

<div class="profile">

    <div class="profile-box">

        {% if student.student_profile_pic %}

        {% else %}

        {% endif %}

        <div class="profile-details">

            <h1>{{ student.name }}</h1>

            <br />

            <p><i class="far fa-user"></i> Имя пользователя: {{
student.user.username }}</p>

```

```

        <p><i class="fal fa-user-tag"></i> Номер ролла: {{
student.roll_no }}</p>

        <p><i class="far fa-phone"></i> Телефон: {{
student.phone }}</p>

        <p><i class="far fa-envelope-open-text"></i> Email:
{{ student.email }}</p>

        <a href="{% url 'classroom:student_update'
pk=user.pk %}"> <p class="edit-button">Редактировать профиль</p>
</a>

    </div>

</div>

</div>

{% endblock %}

```

Файл student_marks_list.html

```

{% extends 'classroom/base.html' %} {% block content %}

<div class="container">

    <div class="jumbotron">

        {% if given_marks.count == 0 %}

        <h2>Вы не выставляли оценки {{ student.name }}.</h2>

        {% else %}

        <h3>Оценки {{ student.name }}</h3>

        <table class="table">

            <thead class="thead-dark">

                <tr>

                    <th scope="col">№</th>

                    <th scope="col">Тема</th>

                    <th scope="col">Полученные оценки</th>

                    <th scope="col">Максимальный балл</th>

                    <th scope="col"></th>

                </tr>

            </thead>

            <tbody>

```

```

        {% for marks in given_marks %}
        <tr>
            <th scope="row">{{ forloop.counter }}</th>
            <th>{{ marks.subject_name }}</th>
            <th>{{ marks.marks_obtained }}</th>
            <th>{{ marks.maximum_marks }}</th>
            <th><a class="btn btn-primary" href="{% url
'classroom:update_marks' pk=marks.pk %}">Обновить</a></th>
        </tr>
        {% endfor %}
    </tbody>
</table>
<br />
{% endif %}

```

```

        <p><a class="btn btn-danger" href="{% url
'classroom:enter_marks' pk=student.pk %}">Добавить
оценки</a></p>
    </div>
</div>
{% endblock %}

```

Файл student_signup.html

```

{% extends 'classroom/base.html' %} {% load static %} {% block
content %}

```

```

<div class="login-page">  <div class="student-signup-form"> {%
if registered %} <h1>Спасибо за регистрацию!</h1> {% else %}
<h1>Зарегистрируйтесь как студент.</h1> <h3>Заполните
форму:</h3>

```

```

    <form method="post">

```

```

        {% csrf_token %}

```

```

        {{ student_profile_form.errors }}

```

```

    <p> <label>{{ student_profile_form.name.label_tag
}}</label>{{ student_profile_form.name }}</p>

    <p> <label >{{ student_profile_form.roll_no.label_tag
}}</label>  {{ student_profile_form.roll_no }}</p>

    <p> <label for="">{{ student_profile_form.phone.label_tag
}}</label> {{ student_profile_form.phone }}</p>

    <p> <label for="">{{ student_profile_form.email.label_tag
}}</label>{{ student_profile_form.email }}</p>


    {{ user_form.errors }}


    <p> <label for="">{{ user_form.username.label_tag
}}</label>{{ user_form.username }}</p>

    <p> {{ user_form.username.help_text }}</p>

    <p> <label for="">{{ user_form.password1.label_tag
}}</label>{{ user_form.password1 }}</p>

    <p>Пароль не может совпадать с другими полями и должен
содержать не менее 8 символов.</p>

    <p> <label for="">{{ user_form.password2.label_tag
}}</label>{{ user_form.password2 }}</p>

    <p> {{ user_form.password2.help_text }} </p>

    <input type="submit" name="" value="Зарегистрироваться">

</form>

<a href="{% url 'classroom:login' %}">Уже есть аккаунт.</a>

{% endif %}

</div>

</div>

{% endblock %}

```

Файл student_update_page.html

```

{% extends "classroom/base.html" %} {% load bootstrap4 %} {%
block content %}

<div class="container">

    <div class="jumbotron">

```

				ПГУ1.090301.01.001 ПЗ	Лист
					135
Изм.	Лист	№ докум	Подпись Дата		

```

    {% if profile_updated %}
    <h1>Ваш профиль успешно обновлен.</h1>
    {% else %}
    <h1>Обновить профиль</h1>
    <form enctype="multipart/form-data" method="post">
        {% csrf_token %}
        {% bootstrap_form form %}
        <input class="btn btn-primary" type="submit" name=""
value="Отправить" />
    </form>
    {% endif %}
</div>
</div>
{% endblock %}

```

Файл students_list.html

```

{% extends "classroom/base.html" %} {% load static %} {% block
content %}

<div class="student-list">
    <div class="container">
        {% if messages %}
        <div class="alert alert-primary alert-dismissible fade
show" role="alert">
            {% for message in messages %}
            <div class="alert {% if message.tags %} alert-{{
message.tags }}{% endif %}">{{ message|safe }}</div>
            {% endfor %} <button type="button" class="close"
data-dismiss="alert" aria-label="Close"><span aria-
hidden="true">&times;</span></button>
        </div>
        {% endif %}
        <form method="GET" action="{% url
'classroom:students_list' %}" class="searchme">

```

```

        <p>Поиск студента: <input type="search" name="q"
placeholder="Поиск" /></p>

    </form>

    <div class="row">

        {% for student in students_list %}

        <div class="col-6 col-md-4">

            <div class="student-box">

                <div class="imgBx">

                    {% if student.student_profile_pic %}
 {%
else %} {% if forloop.counter0|divisibleby:"2" %}  {% else

                    {% static 'images/login-
avatar-2.PNG' }}" alt="" /> {% endif %} {% endif %}

                </div>

                <div class="content">

                    <h3>{{ student.name }}</h3>

                    <p><i class="far fa-user"></i> Имя
пользователя: {{ student.user.username }}</p>

                    <p><i class="fal fa-user-tag"></i>
Регистрационный №.: {{ student.roll_no }}</p>

                    <p><i class="far fa-phone"></i> Телефон:
{{ student.phone }}</p>

                    <p><i class="far fa-envelope-open-
text"></i> Электронный адрес: {{ student.email }}</p>

                    <a href="{% url 'classroom:add_student'
pk=student.pk %}"><p class="add-button">ДОБАВИТЬ</p> </a>

                </div>

            </div>

        </div>

        {% endfor %}

    </div>

</div>

{% endblock %}

```

Файл submit_assignment.html

```
{% extends 'classroom/base.html' %} {% load bootstrap4 %} {%
block content %}

<div class="container">

    <div class="jumbotron">

        <h1>Отправить задание</h1>

        <form enctype="multipart/form-data" method="post">

            {% csrf_token %}

            {% bootstrap_form form %}

            <input type="submit" class="btn btn-primary" name=""
value="Отправить" />

        </form>

    </div>

</div>

{% endblock %}
```

Файл submit_list.html

```
{% extends 'classroom/base.html' %}

{% block content %}

<div class="container">

    <div class="jumbotron">

        {% if messages %}

            <div class="alert alert-primary alert-dismissible fade
show" role="alert">

                {% for message in messages %}

                    <div class="alert {% if message.tags %} alert-{{
message.tags }}{% endif %}">{{ message|safe }}</div>

                    {% endfor %} <button type="button" class="close"
data-dismiss="alert" aria-label="Close"><span aria-
hidden="true">&times;</span></button>

                </div>

            {% endif %} {% if teacher.teacher_submit.count == 0 %}
```

				ПГУ1.090301.01.001 ПЗ	Лист
					138
Изм.	Лист	№ докум	Подпись Дата		

```

<h2>Нет посылок</h2>

{% else %}

<table class="table">
    <thead class="thead-dark">
        <tr>
            <th scope="col">#</th>
            <th scope="col">Отправлено</th>
            <th scope="col">Задание</th>
            <th scope="col">Отправлено</th>
            <th scope="col">Скачать</th>
            <th scope="col">Ввести оценки</th>
        </tr>
    </thead>
    <tbody>
        {% for submission in teacher.teacher_submit.all %}

        <tr>
            <th scope="row">{{ forloop.counter }}</th>
            <td>{{ submission.student.name }}</td>
            <td>{{
submission.submitted_assignment.assignment_name }}</td>
            <td>{{ submission.created_at }}</td>
            <td><a href="{{ submission.submit.url }}"
class="btn btn-primary" target="_blank">Скачать</a></td>
            <td><a class="btn btn-danger" href="{% url
'classroom:enter_marks' pk=submission.student.pk %}">Ввести
оценки</a></td>
        </tr>

        {% endfor %}

    </tbody>
</table>

{% endif %}

</div>

```

</div>

{% endblock %}

Файл teacher_detail_page.html

{% extends 'classroom/base.html' %}

{% load static %}

{% block content %}

<div class="profile">

 <div class="profile-box">

 {% if teacher.teacher_profile_pic %} {% else %} {% endif %}

 <div class="profile-details">

 <h1>{{ teacher.name }}</h1>

 <p><i class="far fa-user"></i> Имя пользователя: {{ teacher.user.username }}</p>

 <p><i class="fas fa-books"></i> Название предмета: {{ teacher.subject_name }}</p>

 <p><i class="far fa-phone"></i> Телефон: {{ teacher.phone }}</p>

 <p><i class="far fa-envelope-open-text"></i> Адрес электронной почты: {{ teacher.email }}</p>

 <p class="edit-button">Редактировать профиль</p>

 </div>

</div>

</div>

{% endblock %}

				ПГУ1.090301.01.001 ПЗ	Лист
					140
Изм.	Лист	№ докум	Подпись Дата		

Приложение В

(обязательное)

Система контроля успеваемости учащихся

Презентация

				<i>ПГУ1.090301.01.001 ПЗ</i>	Лист
					141
Изм.	Лист	№ докум	Подпись Дата		

Тема работы: Система контроля успеваемости учащихся

Автор работы: студент группы 19ВВП1 Анохин А. А.

Научный руководитель: к.т.н., доцент каф. ВТ ПГУ Дубравин А. В.

Пенза 2023

Актуальность работы

Различные системы контроля сегодня используются повсеместно: для ввода, хранения и обработки различных данных. В образовательном процессе они также активно внедряются и используются, поскольку позволяют преподавателя быстро предоставлять и оценивать учебную успеваемость учеников, а также эффективно рассылать учебные материалы, а ученикам системы позволяют просматривать свою успеваемость и иметь доступ к домашним заданиям

Предмет и объект работы

Предмет

- Разработка системы контроля успеваемости учащихся на Python

Объект

- Разработка клиент – серверного web - приложения

Цель работы

Цель работы – разработать систему контроля успеваемости учащихся. Данная система позволит преподавателям сохранять оценки учеников в базу данных и рассылать им домашние задания и напоминания, а ученикам, в свою очередь, просматривать данные учебные материалы.

Постановка задачи

Приложение должно быть спроектировано по принципу клиент-серверной архитектуры.

Клиентская часть программы должна предоставлять интерфейс для работы с приложением в виде веб – сайта в браузере.

Серверная часть должна предоставлять возможность обработки и хранения в базе данных пользовательских данных, таких как оценки по различным предметам, актуальные напоминания и т. д.

Сервис “Госуслуги Пензенская область”



Сервис “Дневник.ру”

No	Предметы	Сумма	Прогнозы Освоения по боязни	Средний балл	1 семестр Итог
1	Астрономия	554	0	0	4.66
2	Геометричес. задачи		0	0	
3	Иностраный язык	4554454545	0	0	4.57
4	Информатика	44554	0	0	4.4
5	История	54544	0	2	4.5
6	Литература	45443534	0	0	4
7	Математика	4545445544	0	4	4.5
8	ОБЖ	4455	0	1	4.66
9	Рус. язык	544	0	0	4.33
10	Русская словесность	4	0	0	4
11	Физика	4454455544	0	3	4.5
12	Физикультура	45454545554	0	0	4.54
13	Химия	4	0	0	4
14	ЭК (алгебра)		0	0	
15	ЭК (физика)	54455	0	1	4.75
16	ЭК (Черчение)	334	0	0	3.33
17	ЭУП (обществознание)		0	0	

A 100% increase in the number of people who are...

Стек технологий



Изм.	Лист	№ докум	Подпись Дата

ПГУ1.090301.01.001 ПЗ

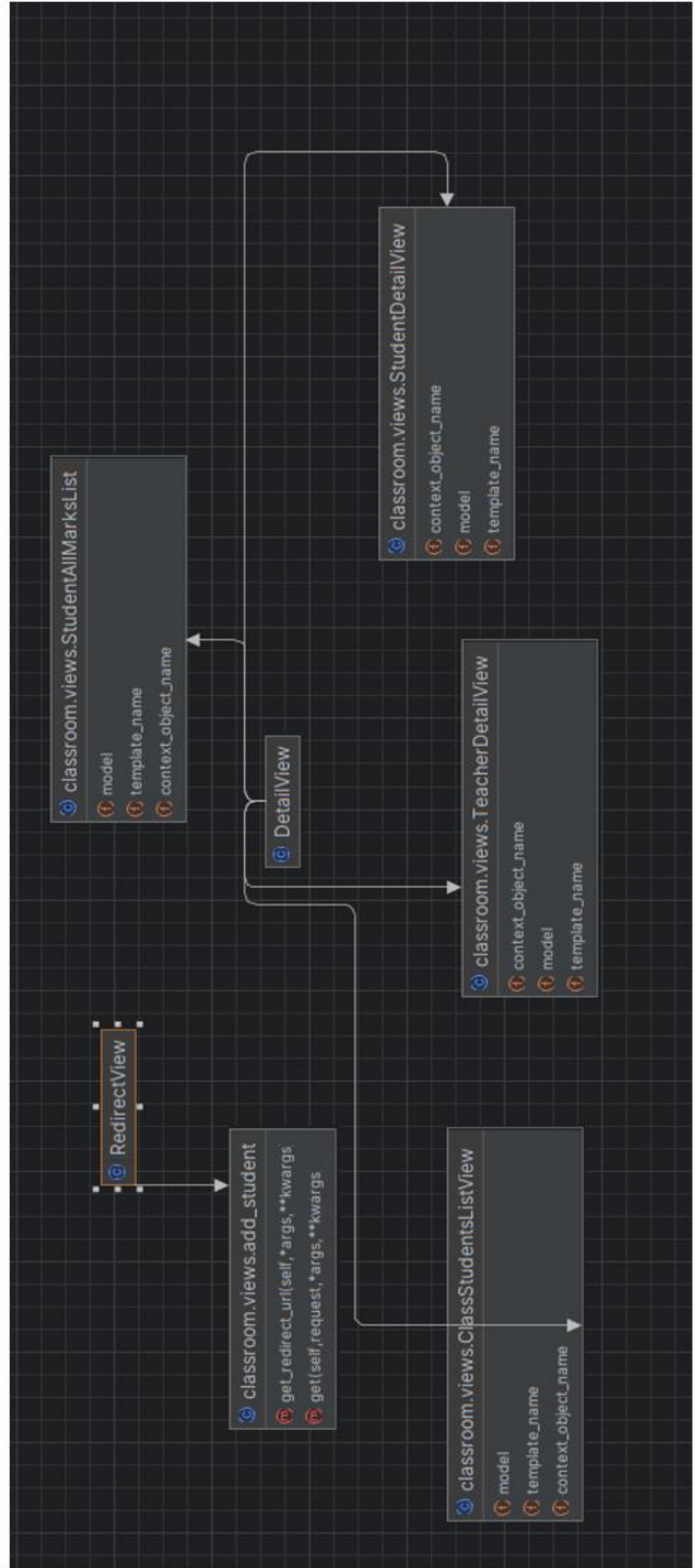
Лист

149

Диаграмма классов (модели)



Диаграммы классов (view)



Диаграммы развертывания



Диаграмма прецедентов (преподаватель)

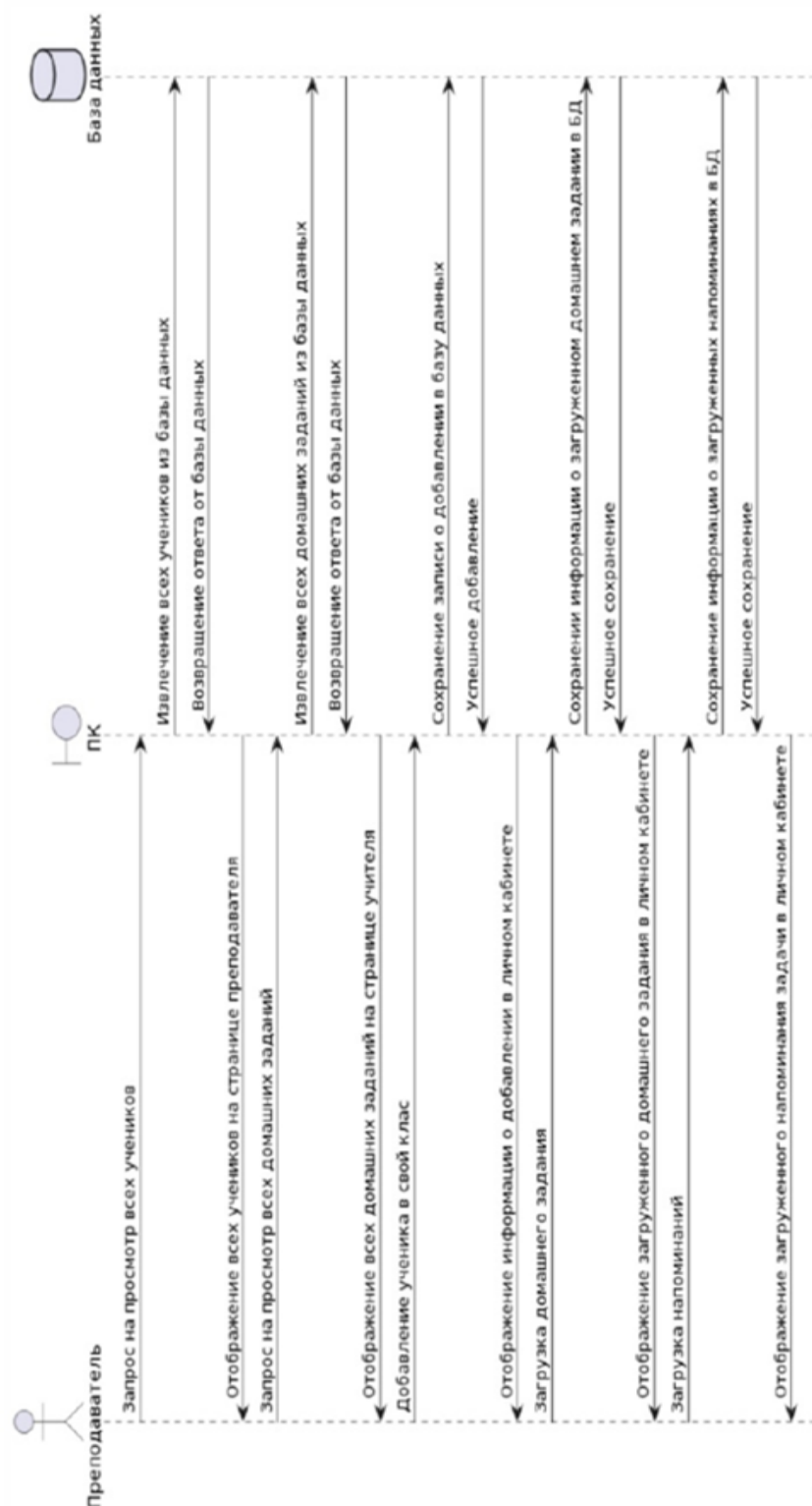
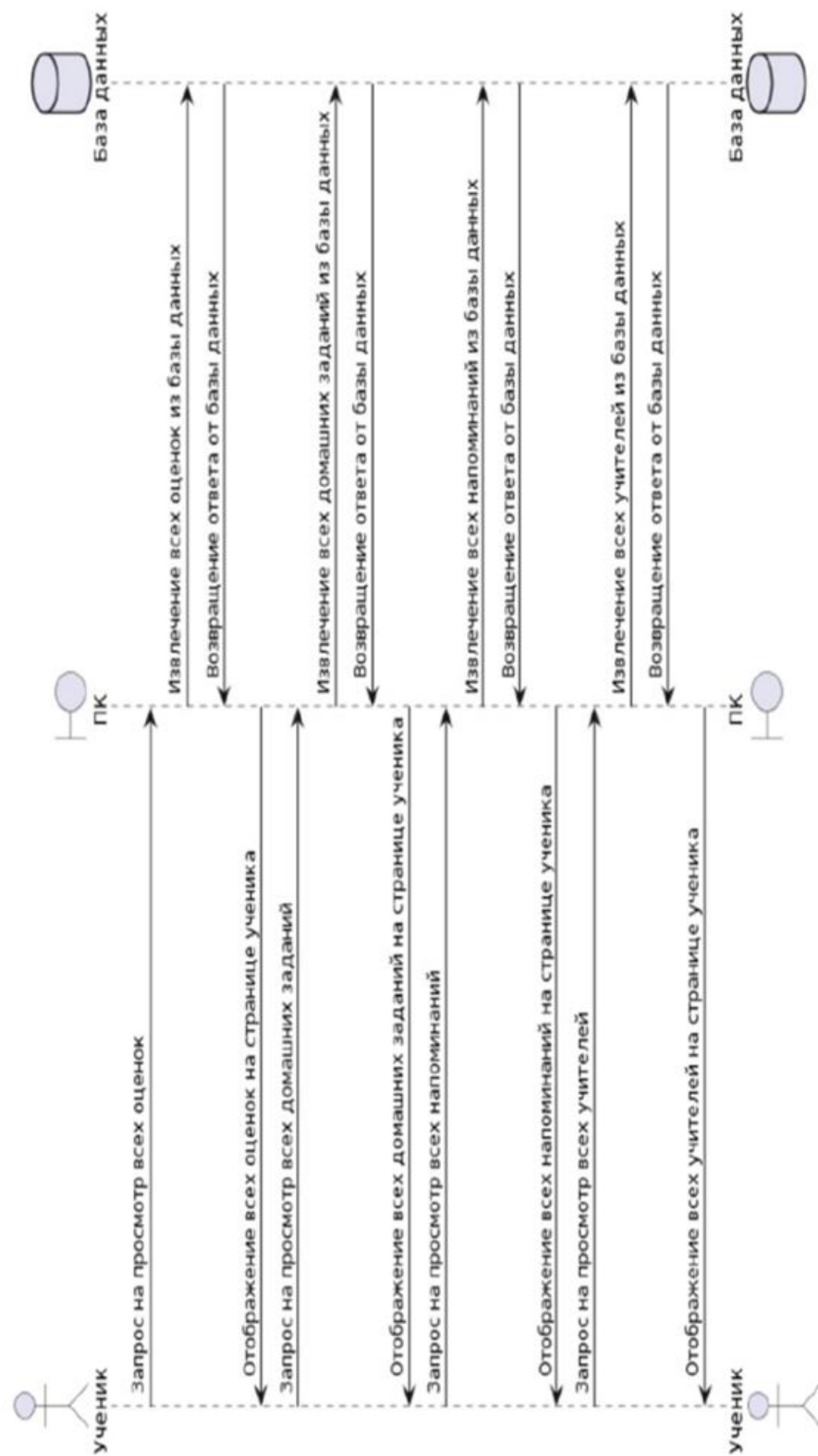


Диаграмма прецедентов (ученик)



Демонстрация работы



Добавим ученика anton себе
в личный кабинет

Ученик был успешно
добавлен в кабинет

Демонстрация работы

Оценки anton

№	Тема	Полученные оценки	Максимальный балл
1	math	4	4

Добавить оценки

Обновить

Выставляем оценку ученику

№	Предмет	Имя учителя	Полученные оценки	Максимальный балл
1	math	denis	4	4

Проверяем, что оценка была выставлена

Демонстрация работы

#	Задача	Дата загрузки	Скачать	Обновить	Удалить
1	Решит пример на 14 странице, номер примера 10	June 21, 2023, 7:38 a.m.	Скачать	Обновить	Удалить

#	Название задания	Загрузил	Дата загрузки	Скачать	Статус
1	Решит пример на 14 странице, номер примера 10	denis	June 21, 2023, 7:38 a.m.	Скачать	Отправить

Добавим домашнее задание
ученику

У ученика домашнее задание
успешно отображена

Заключение

Для реализации данного программного продукта было решено выбрать наиболее подходящий стек: *Python + Django + HTML + CSS + Sqlite3*. Таким примитивные, на первый взгляд, инструменты, позволили сделать сервис достаточно простым и удобным, с интуитивно понятным интерфейсом.

В ходе выполнения работы была разработана система контроля, обладающая всем необходимым заявленным во введении функционалом, с разделением по ролям: и для учителей, и для учеников. Система строится на базе данных, хранящей информацию о пользователях, сервисах для загрузки домашних заданий, напоминаний, а также сервисе, дающем возможность послать сообщение от ученика учителю.

Система была успешно протестирована, весь заявленный функционал отработал, никаких огрехов в приложении при тестировании замечено не было.

Изм.	Лист	№ докум	Подпись Дата

Спасибо за внимание!

				ПГУ1.090301.01.001 ПЗ	Лист
					159
Изм.	Лист	№ докум	Подпись Дата		