Департамент образования и науки города Москвы

Государственное бюджетное профессиональное образовательное учреждение

города Москвы

«Колледж малого бизнеса № 4»

(ГБПОУ КМБ № 4)

**КУРСОВОЙ ПРОЕКТ**

по МДК05.02 Разработка кода информационных систем

для специальности09.02.07 Информационные системы и программирование

Базовая подготовка

Тема Разработка информационной системы для контролирования посещаемости

Выполнил(а) студент(ка)

2 курса группы № ИПС-21.22

Плиева Елизавета Цезаровна

(Фамилия, имя, отчество студента)

Проверил

руководитель курсовой работы

Минеева Кристина Константиновна

Работа защищена с оценкой

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024

Москва, 2024

# **СОДЕРЖАНИЕ**

**Оглавление**

[СОДЕРЖАНИЕ 2](#_Toc166752393)

[ВВЕДЕНИЕ 3](#_Toc166752394)

[ГЛАВА 1. Теоретическая часть 6](#_Toc166752395)

[1.1 Понятие, свойства, актуальные типы и особенности баз данных 6](#_Toc166752396)

[1.2 Система управления базами данных, структурированный язык запросов 11](#_Toc166752397)

[1.3. Front-end часть информационной системы. HTML, CSS, JavaScript 12](#_Toc166752398)

[ГЛАВА 2. Практическая часть создания проекта 15](#_Toc166752399)

[2.1 Определение Visual Studio Code 15](#_Toc166752400)

[2.2 Преимущества и недостатки Visual Studio Code 16](#_Toc166752401)

[2.3 Обоснование выбора программы для разработки 17](#_Toc166752402)

[2.4. Обоснование выбора метода для разработки. Python и фреймворк Django 18](#_Toc166752403)

[2.5. Процесс создание информационной системы 20](#_Toc166752404)

[Заключение 32](#_Toc166752405)

[Используемая литература 33](#_Toc166752406)

# **ВВЕДЕНИЕ**

В современном мире становиться все больше образовательных учреждений. Помимо государственных школ или университетов, стало появляться больше частных организаций. За последние десятилетия люди стали активнее пользоваться их услугами. Это не единственное изменение, которое затронуло образовательную сферу. Если раньше вся документация подобных организаций хранилась исключительно в бумажном виде, то сейчас все чаще используют различные электронные ресурсы.

Одной из важнейших задач, которые стоят перед образовательной организацией — это контроль посещаемости учащихся. Это один из главных аспектов, из которых складывается оценка при аттестации (при очной форме обучения). Поэтому отслеживать такую важную информацию на бумажных носителях не кажется разумным. Как минимум, такой метод ненадежен – высокий риск потери или порчи информации. В наше веке лучшим методом будет использование информационных систем, информацию с которых можно хранить облачно и в нескольких копиях. Так же в электронном формате легче вносить правки и вести совместный доступ.

Невозможно не согласиться, что любому учебному учреждению нужен способ контролировать посещаемость именно в электронном формате, так как это значительно облегчит этот процесс.

Онлайн таблицы посещаемости — это прекрасное решение данной задачи. Раньше преподавателям приходилось вручную отмечать присутствующих в бумажных журналах. Это неудобно, так как учащийся может опоздать, а пометка о прогуле уже поставлена. Приходилось либо зачеркивать, либо использовать корректор. В электронном формате можно будет легко поменять статус учащегося без лишних усилий.

Подобная информационная система облегчит жизнь и ученикам. Еще пару десятков лет назад, чтобы узнать свои текущие оценки и посещаемость, школьникам и студентам приходилось лично подходить к ответственным лицам, у которых хранился бумажный журнал. С помощью электронной таблицы посмотреть интересующую информацию можно будет в любой момент и из любого места. Более того, можно будет воспользоваться такими функциями, которые на бумажных носителях просто невозможно (например сортировка) .

Обычных электронных таблиц существует множество, но найти специализированную для учебного учреждения не так просто. На практике учебные заведения преимущественно пользуются таблицами Excel, либо Google Sheets. Этот вариант нельзя назвать плохим, ведь данные программы справляются с поставленной задачей. Но если говорить о комфорте, то пользоваться ими не самый удобный вариант. Если хранить всю информацию в одной такой таблице, то она будет перегружена и, как следствие, найти в ней что-то конкретное не так быстро. А если разбить данные на несколько таблиц, то все равно придется постоянно переключаться между ними, что также тратит лишнее время.

Следовательно, идеальным решением будет создать сайт, на котором преподаватели могут в одном месте контролировать посещаемость всех учащихся, в то время как те смогут в любой момент получить доступ к этим данным.

Актуальность дипломного проекта обусловлена популярностью частных учебных учреждений, необходимостью контролировать оценки и посещаемость учащихся, потребностью иметь дистанционный доступ к таблицам посещаемости, а также спросом на повышение комфорта.

Новизна исследования состоит в том, что разработчик с помощью авторского подхода к решению поставленной практической задачи определит самые востребованные функции, которые облегчат задачи стоящие перед учебными учреждениями.

Основная цель курсовой работы заключается в разработке информационной системы для контроля посещаемости. В связи с поставленной целью необходимо решить следующие задачи: рассмотреть понятие баз данных, рассмотреть языки программирования HTML, CSS, JavaScript, SQL; фреймворк Django, разработка и внедрение, тестирование информационной системы .

Для достижения данной цели в работе решаются следующие задачи:

* Изучение понятия баз данных;
* Выбрать наиболее подходящую программу для выполнения
* Описание программы для создания таблицы посещаемости;
* Разработка программы для контролирования посещаемости.

Практическая значимость работы: повышение уровня профессионализма в программировании, результаты курсовой работы могут быть использованы как дополнительная работа к портфолио (резюме).

Заказчиком модель может использоваться на сайте образовательного учреждения для личного и многопользовательского просмотра.

Выполненная работа состоит из введения, основной части из двух разделов (теоретической и практической), заключения и списка литературы.

# **ГЛАВА 1. Теоретическая часть**

Данная глава представляет собой описание существующих и актуальных видов баз данных, особенности их использования, и программирования с их помощью.

## **1.1 Понятие, свойства, актуальные типы и особенности баз данных**

База данных - совокупность взаимосвязанных данных, совместно хранимых в одном или нескольких компьютерных файлах. Имеются и другие определения из международных и российских стандартов: совместно используемый набор логически связанных данных (и описание этих данных), предназначенный для удовлетворения информационных потребностей организации.

Скорее всего вы ежедневно, хотя бы косвенно, сталкиваетесь с базами данных в повседневной жизни. Неважно, находитесь ли вы непосредственно в магазине или изучаете каталог в интернете - когда вы проверяете, есть ли какой-либо товар на складе, вероятно, где-то в системе используется база данных. Amazon и Facebook, YouTube и iTunes — все эти крупные копании используют базы данных для предоставления продуктов и услуг своим пользователям.

Цель баз данных — хранение информации, с целью дальнейшего ее использования. Это весьма гибкая структура, которую можно использовать в самых различных областях. Большая часть информационных систем не может быть реализована без соответствующей базы данных.

Примеры могут быть абсолютно разными:

* Маркетплейсы
* Медицинские учреждения
* Социальные сети
* Спортивные залы с системой абонементов
* Учебные заведения

Это лишь малая часть области применения. База данных и ее структура могут быть, как и явными для пользователя, например, в библиотечном каталоге или онлайн-магазине, так и менее очевидными, например, позволяя компании отслеживать штат сотрудников и поставщиков, либо помогая рекламодателю отслеживать посетителей веб-страниц на разных сайтах, подстраивая свою рекламу под пользователя.

Система баз данных — это система, хранящая данные. Чтобы считаться системой баз данных, она должна обладать некоторыми свойствами:

* + Находить (извлекать) данные
  + Добавлять (вставлять) новые данные
  + Удалять ненужные данные
  + Изменять (обновлять) данные.

Рассмотрим эти возможности на примере. Представим абстрактный магазин, специализирующийся на продаже одежды. Магазин хранит информацию о продаваемых товарах. Эта информация может быть организована в виде таблицы (таблица 1) и может быть частью базы данных магазина.

Таблица 1

|  |  |  |  |
| --- | --- | --- | --- |
| Модель | Бренд | Цена | Количество на складе |
| Красное платье | ZARA | 3000р | 10 |
| Атласная юбка | Bershka | 1500p | 14 |
| Белая рубашка | H&M | 2000р | 6 |

Модель «Красное платье» бренда «ZARA» скоро будет снята с производства, и магазин больше не будет их продавать. Как только все имеющиеся на складе платья будут распроданы, информация об этой модели (строка в таблице) - будет удалена.

Атласная юбка «Bershka» продается плохо, но при этом занимает много места на складе, поэтому цена товара будет снижена до 1000 рублей. Соответствующее поле для цены этого товара (ячейка в таблице) будет обновлено. Только что были выпущены новые джинсы от ZARA, поэтому необходимо вставить новую запись (новую строку в таблице).

В этом примере мы наглядно увидели, как работает каждое свойство

* Мы можем посмотреть в таблице всю информацию о каждом товаре
* Мы можем добавлять новые товары
* Мы можем менять цену (либо любую другую характеристику) товара
* Мы можем удалять определенные товары

При внедрении базы данных в свой проект, следует заранее ответить на ряд вопросов:

* Какая часть информации будет пополняться, а какая оставаться неизменной?
* В каких ситуациях и с какой целью будет использоваться информация из базы данных?
* Какая информация из базы данных будет доступна пользователю?
* Каким образом пользователь сможет получить доступную ему информацию из базы данных?
* Подразумевается ли возможность изменения информации в базе данных пользователем?
* Какая функция у базы данных в конкретном проекте?

Так же важно определиться, какой тип БД выбрать. Среди множества типов баз данных можно выделить следующие:

1. Реляционные БД.

В данном типе баз данных информация представлена в формате таблиц со столбцами и строками. Реляционная модель данных основана на математической теории множеств.

Реляционная база данных состоит из трех частей:

* Структурная часть определяет отношения данных и их взаимосвязь
* Часть целостности обеспечивает уникальность каждого вхождения в отношение
* Манипуляционная часть предоставляет операторы для обработки отношений

Для того чтобы организовать данные в таблицах, используется техника, называемая нормализацией. Нормализация — это техника, которая помогает определить наиболее подходящую группировку элементов данных в записи. Она обладает следующими свойствами:

* В любом столбце таблицы все элементы имеют одинаковый вид.
* Каждый элемент — это простое число или строка.
* Каждый столбец имеет отдельное имя, и порядок следования столбцов несущественен.
* Каждая строка таблицы является отдельной. Другими словами, нет двух одинаковых строк, в которых каждый столбец был бы идентичен.
* Последовательность строк в таблице несущественна

1. Объектно-ориентированные БД.

Здесь данные рассматриваются как объекты, их атрибуты, методы и классы. Объектно-ориентированная модель данных позволяет работать с объектами, а не с записями. В объектно-ориентированной модели элемент представлен как объект класса, который имеет набор методов, применяемых к объектам. Класс представляет собой абстрактный тип данных и является шаблоном, по которому мы можем создать столько экземпляров, сколько захотим. Класс и, следовательно, объект могут наследовать свойства и методы от других классов. Объекты и классы динамичны и могут быть созданы в любое время.

Рассмотрение данных в виде объектов, а не записей обеспечивает большую гибкость и устраняет необходимость нормализации данных

1. Иерархические БД.

В качестве формы представления используется древовидная структура. Иерархическая база данных состоит из набора записей, объединённых между собой связями. Связь — это ассоциация между двумя записями. Каждая запись представляет собой набор полей (атрибутов), каждое из которых содержит одно значение.

Структурно иерархическая модель может быть представлена в виде перевернутого дерева, самый верхний уровень которого называется корнем. Дерево имеет один корень и множество ветвей, выходящих из него. Каждая ветвь может иметь подветви. Место, где появляются ветви, называется узлом. Таким образом, дерево — это совокупность узлов. Один узел обозначается как корневой, остальные узлы образуют деревья или поддеревья.

Иерархическая модель данных имеет следующие особенности:

* Каждое иерархическое дерево может иметь только один корень, и он не имеет родителя.
* Корень может иметь любое количество дочерних записей, каждый из которых сам может быть корнем иерархического поддерева.
* Каждый дочерний тип записи может иметь только один родительский тип записи.
* Данные в родительской записи применяются ко всем ее дочерним записям.

1. Сетевые базы данных

Сетевая модель базы данных была разработана для решения некоторых проблем, связанных с иерархической моделью базы данных. Сетевая модель очень похожа на иерархическую модель баз данных. Однако в сетевой модели отношения представлены в качестве множеств, а не иерархии. Это позволило сетевой модели решить проблему избыточности данных.

1. Хранилище данных:
2. Распределенные БД
3. Графовые БД.
4. Нереляционные БД.
5. Аналитические БД

## **1.2 Система управления базами данных, структурированный язык запросов**

Для создания информационной системы для отслеживания посещаемости, необходима база данных. Это логично, так как сама концепция такой программы подразумевает хранение большого количества информации. Разобравшись с типами баз данных, можем сделать вывод что наиболее подходящей для данной задачи является реляционная база данных. Но мало просто создать базу данных, для взаимодействия с ней нам необходим “посредник”

Система управления базами данных — это программное обеспечение, с помощью которого осуществляется доступ к базе данных. Это краткое, но ограниченное определение. На самом деле СУБД отвечает за гораздо большее:

1. Определение данных. СУБД должна обеспечивать поддержку определения или модификации схемы базы данных. Определение схемы включает в себя указание типов данных, структур, ограничений и ограничений безопасности. Это достигается с помощью языка определения данных (DDL).
2. Манипулирование данными. СУБД должна обеспечивать поддержку манипулирования данными. В частности, она должна поддерживать:
   * извлечение существующих данных, содержащихся в базе данных
   * удаление старых данных из базы данных
   * вставку новых данных
   * модификацию устаревших данных.
3. Поддержка целостности данных. Система должна гарантировать отсутствие противоречий между значениями данных в базе данных
4. Поддержка контроля безопасности. Система должна гарантировать, что данные не будут доступны неавторизованным пользователям или приложениям
5. Восстановление данных. Возможность восстановления базы данных в прежнее состояние в случае сбоя или ошибки
6. Параллелизм. Возможность одновременного доступа к базе данных более чем одного пользователя
7. Обмен данными

Когда данные объединены в единую базу данных, они могут использоваться многими приложениями. Соответственно, логичной организацией системы баз данных является архитектура клиент-сервер. СУБД является сервером, а прикладные программы - клиентами. Сервер также можно назвать backend, а клиента — frontend.

В архитектуре клиент-сервер СУБД работает на выделенной машине - сервере. Сервер приспособлен для поддержки СУБД, как с точки зрения пространства для хранения данных, так и с точки зрения вычислительной мощности в сочетании с достаточным объемом внутренней памяти.

Основное требование к серверу — предоставить ресурсы, необходимые СУБД для эффективного реагирования на запросы клиентов (то есть предоставить то, что им нужно, с соответствующей скоростью).

Раннее мы уже определились, что в нашем случае нужна реляционная база данных, а также СУБД.

Однако для систем управления базами данных существуют специальные языки программирования, называемые языками запросов, которые можно использовать для доступа к данным из базы.

Структурированный язык запросов (SQL) — самый популярный язык запросов, используемый в основных реляционных системах управления базами данных, таких как MySQL, ORACLE, SQL Server и др.

Язык SQL прост в изучении, поскольку запросы состоят из описательных английских слов и не чувствительны к регистру. Преимущество SQL в том, что нам не нужно указывать, как получить данные из базы данных. Скорее, мы просто указываем, что нужно получить, а SQL делает все остальное.

## **1.3. Front-end часть информационной системы. HTML, CSS, JavaScript**

Теперь, когда мы разобрались с базой данных, можно приступать к созданию информационной системы. Наиболее удобным форматом будет создание сайта. Для этого нам нужно Знание HTML, СSS, JS для создания frontend части.

Итак, HTML-документ — это просто текстовый файл, содержащий информацию, которую вы хотите опубликовать, и соответствующие инструкции по разметке, указывающие, как браузер должен структурировать или представить документ.

Веб-сайты, созданные с помощью HTML и CSS, обычно состоят из HTML-файла, содержащего текст, ссылки на изображения и HTML-теги, и CSS-файла, содержащего стили, которые применяются к HTML-файлу. Например, HTML-файл может содержать текст заголовка (обозначенный HTML-тегом <h1>) и текст абзаца (обозначенный HTML-тегом <p>). Соответствующий CSS-файл может содержать правила, предписывающие браузеру сделать весь текст заголовка размером 20 пикселей, а весь текст абзаца - синего цвета. Эти стилевые правила CSS будут применяться ко всем заголовкам и абзацам, где бы они ни появлялись в HTML-документе, без необходимости каждый раз добавлять стилевые инструкции в HTML-документ. CSS также является мощным инструментом для упорядочивания содержимого сайта. Задавая размер, цвет и другие свойства элементам HTML, вы можете использовать CSS для создания блоков содержимого, которые структурируют и стилизуют макет веб-страницы.

Интернет по своей природе является интерактивной средой, и хороший тому пример - гиперссылки. Даже самый простой сайт требует взаимодействия с пользователем. HTML предоставляет очень мало возможностей для этого, но даже добавив CSS лишь, вы лишь слегка расширите функционал. У CSS есть определенные ограничения, которые можно преодолеть с помощью языка сценариев JavaScript

JavaScript позволяет расширить функциональность и внешний вид сайта с помощью ряда интерактивных задач, которые варьируются от простых, таких как проверка формы, до сложных, таких как анимированные слайд-шоу. JavaScript — это язык сценариев, который сложнее в изучении и использовании, чем HTML и CSS. Однако появление библиотек JavaScript упростило добавление интерактивных элементов, что привело к росту числа разработчиков, использующих JavaScript. Библиотеки JavaScript предоставляют интерактивные функции, в значительной степени скрытые от глаз дизайнера, которые могут быть добавлены на страницу без особых усилий.

# **ГЛАВА 2. Практическая часть создания проекта**

Данная глава представляет собой основную информацию по редакторам кода, его особенности, технологии и причину выбора данного приложения.

## **2.1 Определение Visual Studio Code**

Как уже было упомянуто раннее, для комфортного создания сайта просто необходим редактор кода. Мы будем использовать Visual Studio Code. VS Code — это легкий, но мощный редактор исходного кода, который доступен для Windows, macOS и Linux. Он занимает относительно немного памяти, гибкий и удобный.

Visual Studio Code обладает широкими возможностями персонализации, что позволяет настраивать интерфейс и сочетания клавиш по своему вкусу. Это делает его отличным выбором для разработчиков, которые хотят адаптировать свою среду кодирования к своим конкретным потребностям и вкусовым предпочтениям.

VS Code не стоит путать с Visual Studio — это IDE, очень мощная и масштабная, но одновременно с этим тяжеловесная. Названия похожи, потому что у обоих средств разработки один и тот же создатель, но продукты разные по своей сути. VS Code меньше весит и проще в освоении.

На рисунке 2 изображено окно открытия программы.

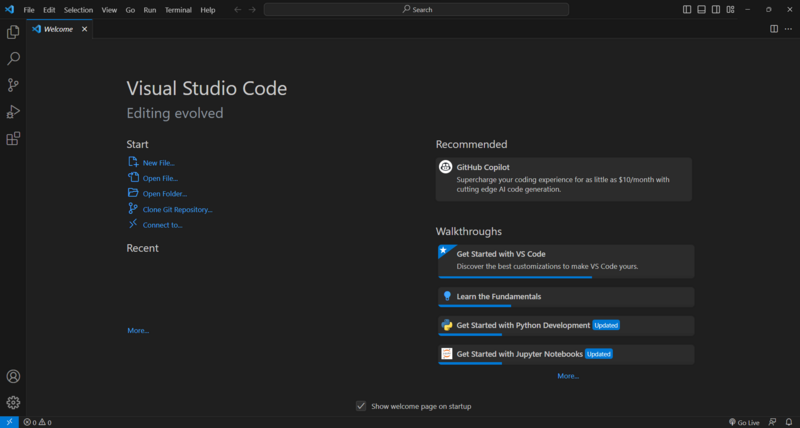


Рисунок 2. Окно открытия программы

Изначально программа запускается с интерфейсом на английском, но при желании легко можно поменять на русский (либо другой удобный для разработчика)

## **2.2 Преимущества и недостатки Visual Studio Code**

Visual Studio Code - хороший выбор редактора кода для веб-разработки. Он предлагает невероятный функционал с тысячами инструментов и расширений, готовых работать в вашу пользу.

Из преимуществ можно выделить следующие:

* VS Code — это кроссплатформенный редактор кода, который может легко работать на macOS, Windows и Linux. Все веб-технологии VS Code в конечном итоге используют Electron Framework, а это значит, что приложения, созданные с помощью VS Code, будут легкими и беспроблемными при их обновлении.
* Если пользователь/программист хочет использовать язык программирования, который не поддерживается, он может легко его загрузить и использовать расширение. Что касается производительности, расширение не замедляет работу редактора, поскольку оно выполняется как отдельный процесс
* Несмотря на то, что она создана как редактор кода, вы можете повторить то, что делает интегрированная среда разработки. VS Code довольно прост и понятен, поэтому разработчик не запутается в сложностях.
* VS Code автоматически выделяет красным цветом области, в которых обнаружены потенциальные ошибки, что упрощает их поиск и исправление. Более того, есть специальная вкладка «Проблемы», в которой перечислены все потенциальные ошибки в одном месте и упрощается их просмотр, как показано на рисунке 3.



Рисунок 3

* Он не выглядит загроможденным, и вы можете легко найти любую нужную вам функцию, выполнив поиск.
* Так же VS Code позволяет открыть несколько проектов, содержащих несколько файлов/папок. Эти проекты/папки могут быть связаны или не связаны друг с другом.

К сожалению, в приложении имеются и недостатки, среди которых в основном называют:

* Интеграция базы данных доступна в VS Code с использованием расширения SQLTools. Однако интеграция VS Code по-прежнему оставляет место для большего. Новичку может быть сложно использовать или перемещаться по нему, что может привести к ошибкам.
* Чтобы персонализировать VS Code, вам придется потратить некоторое время на настройку в соответствии с вашими потребностями, для чего вам потребуется просмотреть и загрузить различные расширения. Однако это дает вам больше контроля над тем, как выглядит, что содержит и что делает ваше рабочее пространство.

## **2.3 Обоснование выбора программы для разработки**

Поскольку мы создаем сайт, нам необходимо писать много кода, как для front-end части, так и для back-end. VS CODE, если сделать выводы на основе вышеперечисленного, прекрасно подходит для этих целей.

Visual Studio Code невероятно гибок. Он может делать практически все, что захочет разработчик, поэтому мы можем выполнить практически всю работу в одной программе, что очень удобно. VS Code позволяет открыть несколько проектов, содержащих несколько файлов/папок. Эти проекты/папки могут быть связаны или не связаны друг с другом.

VS Code — это мощный и универсальный текстовый редактор, который можно легко адаптировать к широкому спектру проектов кодирования. Гибкость и возможности настройки, предоставляемые программным обеспечением, делают его ценным инструментом для проектов кодирования, а встроенные функции отладки и интегрированные функции терминала делают его отличным выбором. Различные расширения, доступные для разных языков программирования и инструментов, также очень полезны.

С его помощью можно быстро создать проект и структуру файлов в нем, он подсвечивает синтаксис кода и помогает автоматически править ошибки. В нем есть возможности для отладки и запуска кода на некоторых языках и подходит в том числе для начинающих разработчиков.

В VS Code легко ориентироваться, понимать и работать, поскольку интерфейс удобен для новичков, приветлив и прост в навигации.

## **2.4. Обоснование выбора метода для разработки. Python и фреймворк Django**

Кроме очевидных языков программирования HTML, CSS, JS, которые сделают внешнюю оболочку нашему сайту, мы так же будем использовать фреймворк Django. Это фреймворк основан на языке программирования python.

У python есть много преимуществ, например, в последние годы он лидирует по сравнению с другими языками программирования в использовании и широко используется программистами. С момента выхода языка Python 25 лет назад, он претерпел значительные изменения: появилось множество дополнительных функций. Многие компании, занимающиеся разработкой программного обеспечения, предпочитают этот язык из-за его универсальных возможностей и чистых и аккуратных кодов программирования. Python обладает множеством характерных особенностей, таких как интерактивность, интерпретируемость, модульность, динамичность, объектно-ориентированность и т. д. Можно много говорить о преимуществах Python. Он был самым популярным языком разработки в последние годы и продолжает оставаться любимым среди многих опытных разработчиков. Но давайте сфокусируемся на том, почему мы выбирает именно фреймворк Django.

1. Выгода для заказчика: если на середине проекта вам потребуется сменить команду разработчиков, Django позволит вам найти новую команду, которая сможет понять архитектуру проекта без особых усилий. Это может сэкономить много времени и денег.
2. Выгода для разработчика: даже если проект от начала до конца разрабатывается одной командой, использование Django значительно ускоряет процесс разработки - от идеи до релиза, запуска и производства. Благодаря прозрачному, чистому коду разработка может быть эффективной и результативной.
3. Django быстр: Он был разработан таким образом, чтобы помочь разработчикам сделать приложение как можно быстрее. От идеи до производства и выпуска, Django помогает сделать его экономичным и эффективным. Таким образом, он становится идеальным решением для разработчиков, ориентированных на соблюдение сроков.
4. Django безопасен: когда вы работаете на Django, разработчики не допускают никаких ошибок, связанных с безопасностью. Для эффективного управления именами пользователей и паролями ключевое значение имеет система аутентификации пользователей.
5. Django отличается гибкостью: Управление контентом, научные вычислительные платформы и даже крупные организации - все эти аспекты очень эффективно управляются с помощью Django.

## **2.5. Процесс создание информационной системы**

Теперь можно приступать к созданию проекта. Первым делом запустим VS Code. Нас встречает пока что пустое окно (рисунок 4), где вскоре будет располагаться наш код.

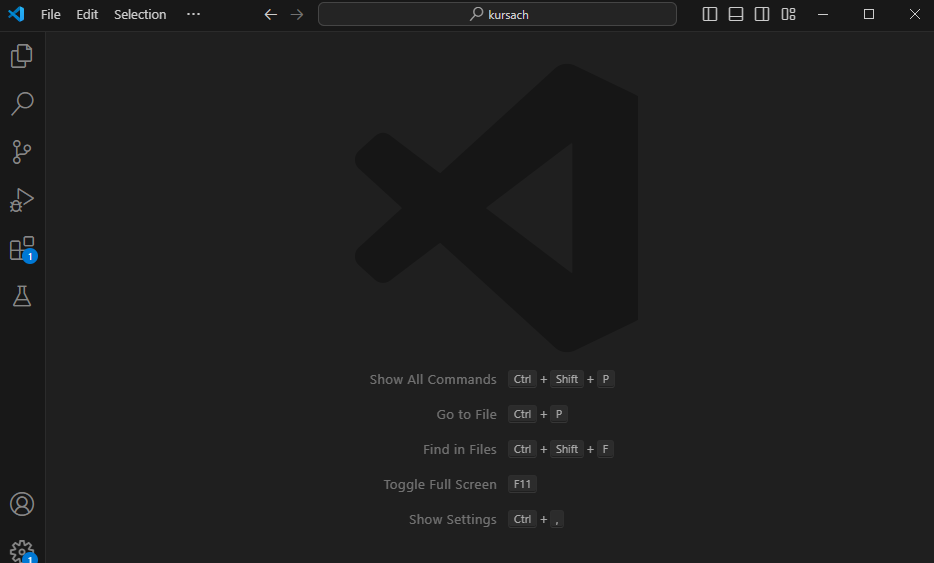


Рисунок 4

Заранее создаем на рабочем столе (либо в другом месте) пустую папку, в которой будет находиться наш проект. Далее, в левом углу VS Coda’a, находим раздел file и нажимаем на него. Перед нами раскрывается список. (Рисунок 5)

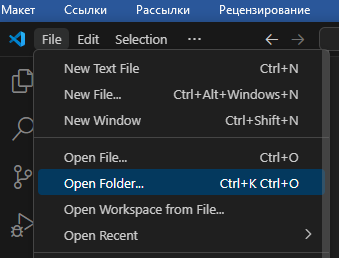


Рисунок 5

Нас интересует кнопка ‘Open folder’, нажимаем на нее. В открывшемся окне ищем заранее созданную папку и выбираем ее. (Рисунок 6)

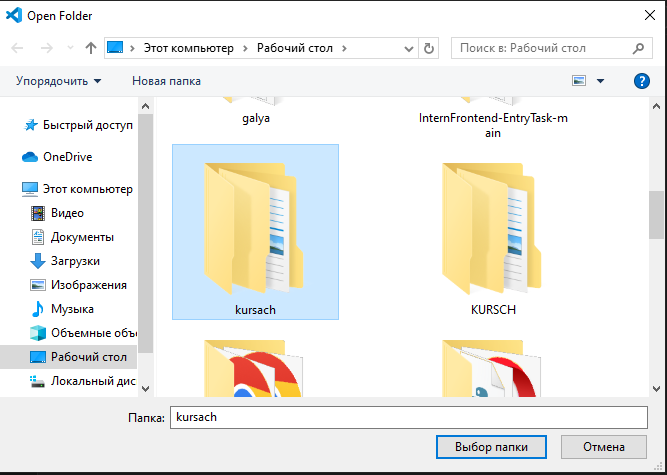


Рисунок 6

Прежде чем приступить к началу нового проекта, нужно убедиться, установлен ли на устройстве python. Для этого в терминале введите *python3 --version* и нажмите Enter. Если на экран выведется версия – все установлено и можно продолжать. Если нет, то нужно просто установить его через официальный сайт. (рисунок 7)

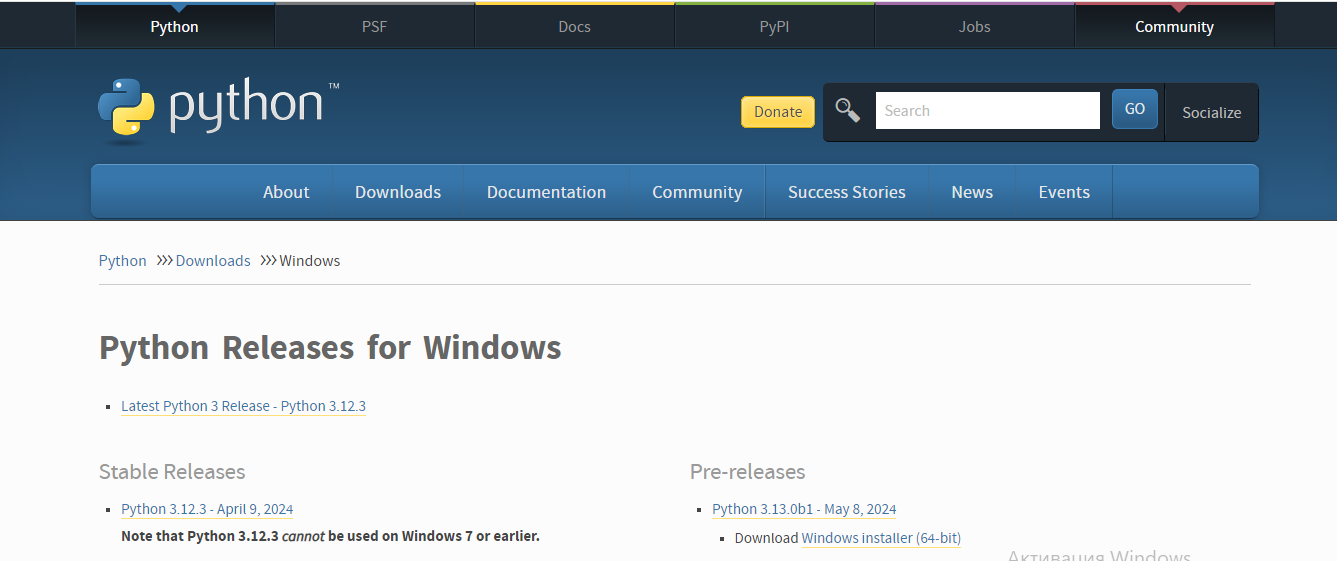


Рисунок 7

Когда python установлен, нужно установить Django. Это делается через пакетный менеджер pip. Для установки Django выполним в консоли следующую команду: *python -m pip install Django*. Теперь можно приступать к созданию сайта. Первым делом создадим новый проект Django. Для этого вводим в терминал *Django-admin startproject [Название проекта] (В нашем случае mysite)*(Рисунок 8). После этого в нашей папке появиться новая папка, содержащая наш проект, внутри которого так же уже автоматически созданы файлы (Рисунок 9)

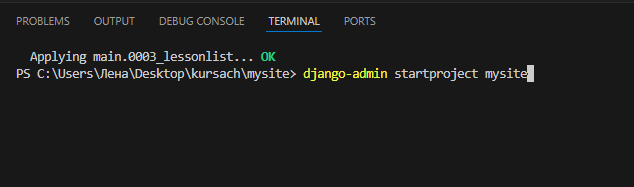


Рисунок 8

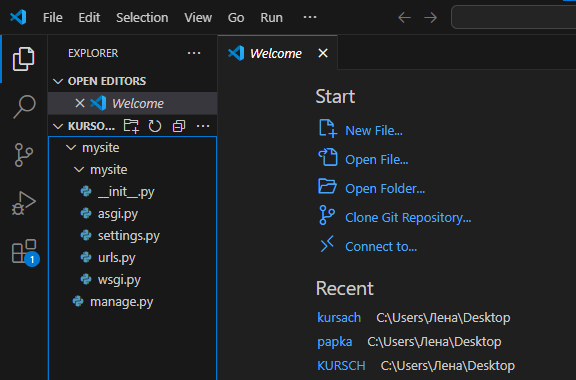


Рисунок 9

Что бы запускать наш сайт, нам потребуется сервер, мы будем использовать локальный. Это делается через файл manage.py, поэтому в терминале нам нужно перейти на папку с нашим проектом (в нашем случае mysite). (Рисунок 10)

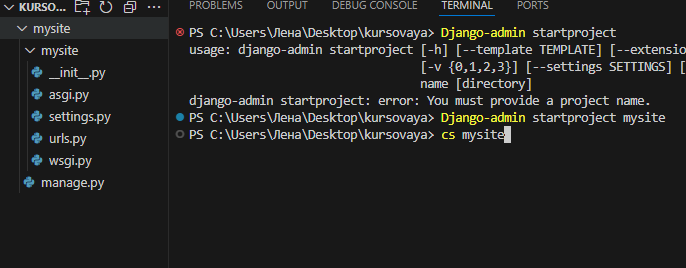


Рисунок 10

Чтобы подключиться к серверу, нужно в терминал ввести *python manage.py runserver*. После выполнения команды нам предоставится ссылка (Рисунок 11), через которую можно перейти на страницу нашего будущего сайта (Рисунок 12). Чтобы выйти из локального сервера, нужно в терминале нажать на сочетание клавиш Ctrl+C.

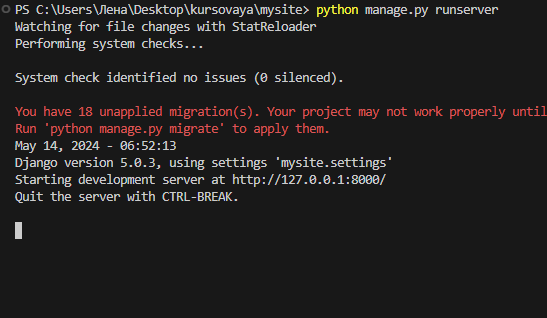


Рисунок 11

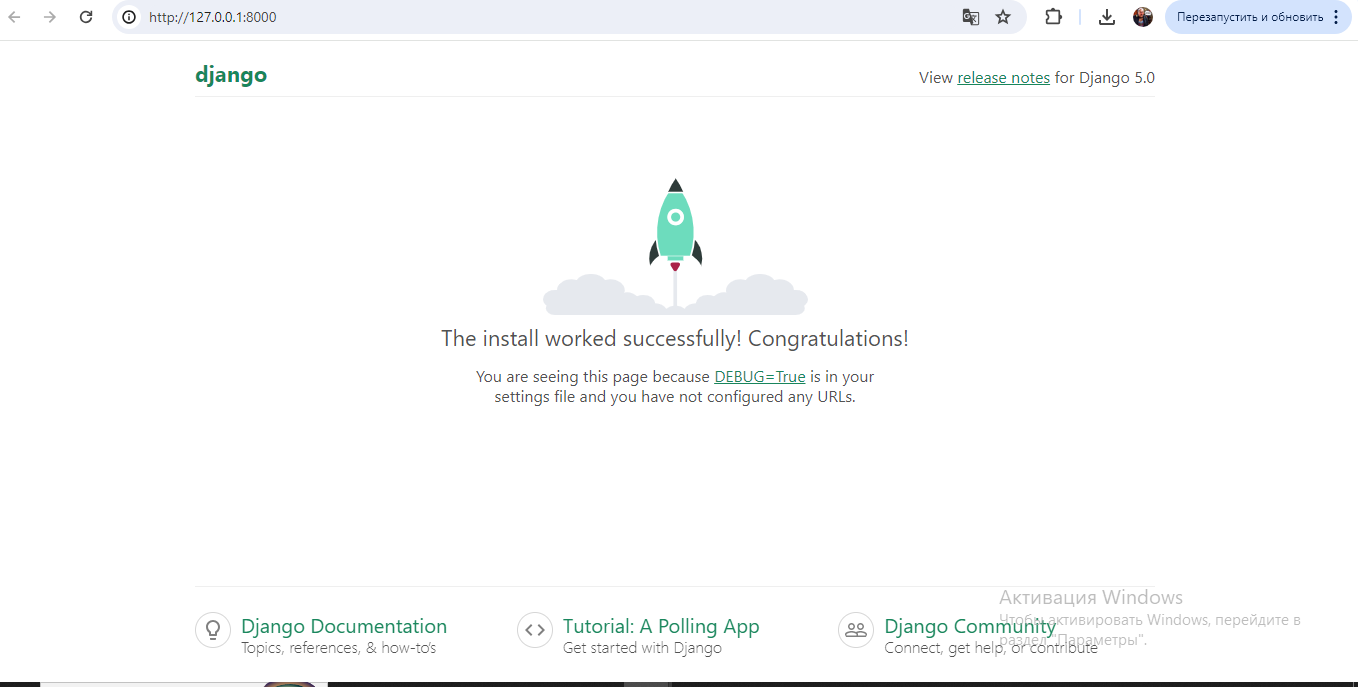


Рисунок 12

Теперь создадим наше первое приложение, которое будет главной страницей нашего сайта. Для этого вводим в терминал команду *python manage.py startapp main* (либо любое другое название приложения), после чего в нашем проекте появиться папка с новым приложением. (Рисунок 13)

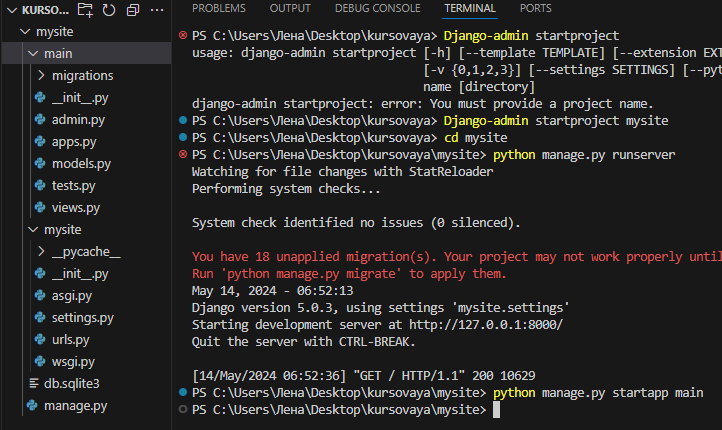


Рисунок 13

Теперь главное зарегистрировать наше приложение. Для этого заходим в папку mysite и находим файл settings.py. В этом файле ищем INSTALLED\_APPS и вписываем туда наше приложение (Рисунок 14). Это обязательно, без этого не будет работать. Это необходимо делать со всеми приложениями.



Рисунок 14

Не выходя из папки mysite, заходим в файл urls.py и прописывает url адресс, который будет ссылаться к приложению main. (Рисунок 15). Обратите внимание, что мы делаем переход к файлу main/urls.py, которого сейчас еще нет. Поэтому нужно самим создать такой файл в папке main. В этом файле прописываем практически все тоже самое, но импортируем метод views и через него ссылаемся к views.index (Рисунок 15). Такого представления тоже пока нет, поэтому переходим к файлу main.views и добавляем его (Рисунок 16)

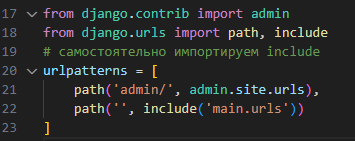


Рисунок 14

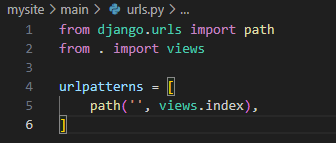


Рисунок 15

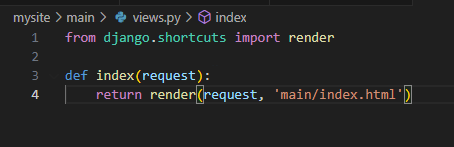


Рисунок 16

Здесь мы с помощью метода render выводим HTML страницу с названием index.html, которой пока не существует. Чтобы создавать HTML страницы, нам нужно в папке main создать папку с названием templates (название только такое), внутри нее создать еще одну папку с названием нашего приложения (main) и уже там создать файл index.html (Рисунок 17). Добавим в этот файл немного содержания (Рисунок 18), чтобы проверить как все работает и откроем страницу через локальный сервер (Рисунок 19)

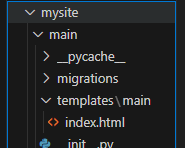


Рисунок 17

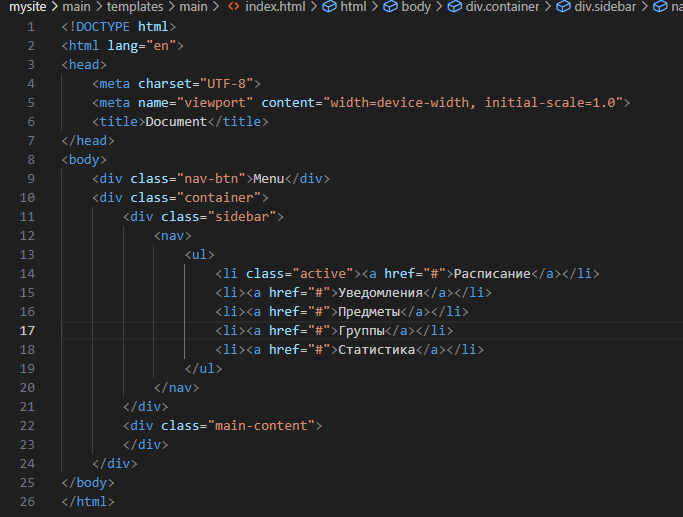


Рисунок 18

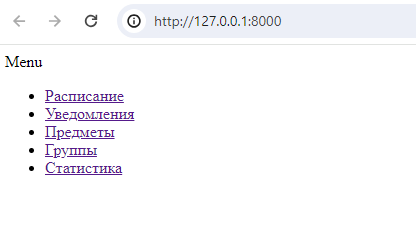


Рисунок 19

Уже хорошо, но визуальная часть сайта не очень приятна глазу. Очевидно, что не хватает CSS файлов, поэтому добавим их. В папке main создаем папку static ( название только такое), внутри нее папку main, а внутри нее папку CSS. Там уже и создаем файл style.css (Рисунок 20). В HTML файле нужно подключить папку static и в ссылках указать путь к нашему CSS файлу (Рисунок 21). Заполним наш стиль (Рисунок 22). Но чтобы это все заработало, нужно в файле mysite/settings прописать специальный список (Рисунок 22). Теперь наша страница выглядит лучше (Рисунок 23)

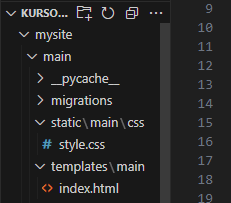


Рисунок 20

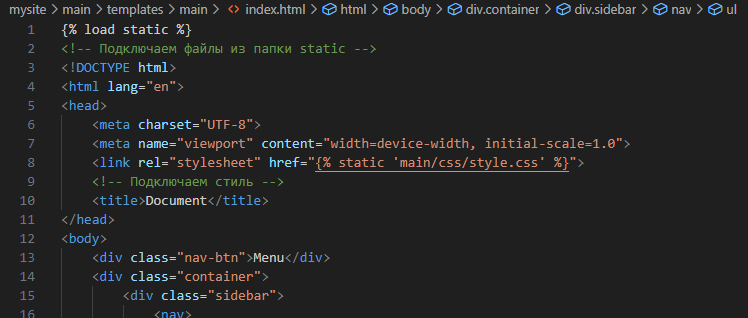


Рисунок 21



Рисунок 22

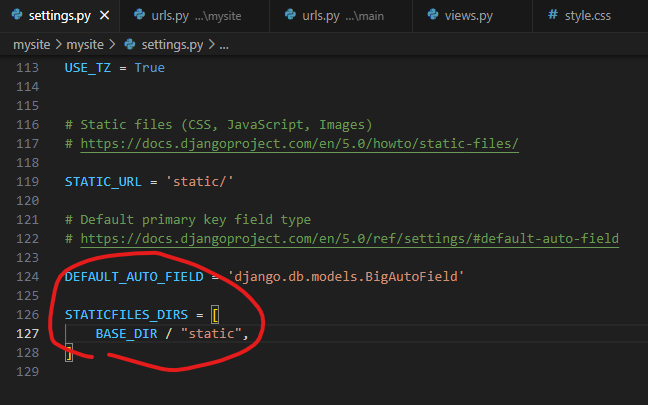


Рисунок 23



Рисунок 24

Теперь у нас есть страница с боковым меню. Логично, что на всех страницах сайта это меню будет неизменным, поэтому будет удобно создать шаблон, в котором оно будет постоянным, а содержимое сайта будет меняется. Что бы сделать это, создадим в папке templates/main файл layout.html и скопируем туда содержимое index.html. Но в местах, где планируем менять контент, пишем специальную конструкцию (Рисунок 25). А уже в самом index.html сначала подгружаем содержимое шаблона, а в места блоков вставляем контент (рисунок 26). Проверяем (Рисунок 27)

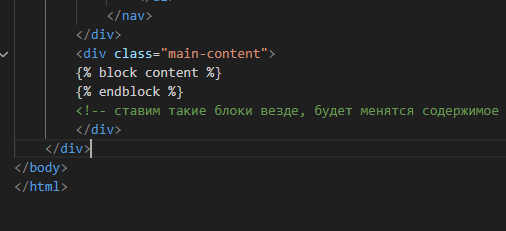


Рисунок 25

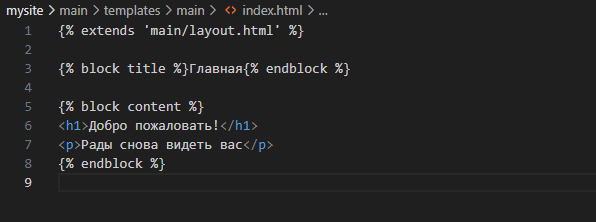
****

Рисунок 26

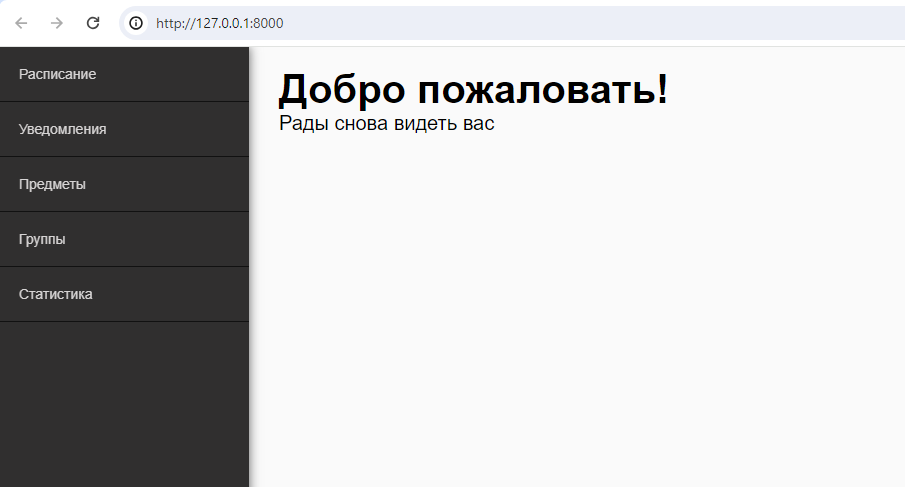
****

Рисунок 27

Теперь можно начинать делать страницы сайта. Начнем с вкладки “Расписание”. По плану там должно находиться расписание на текущий день, а также следующий. В будущем расписание будет браться из базы данных, но пока создадим саму страницу. Повторяем все тоже, что и с приложением main (рисунок 28)

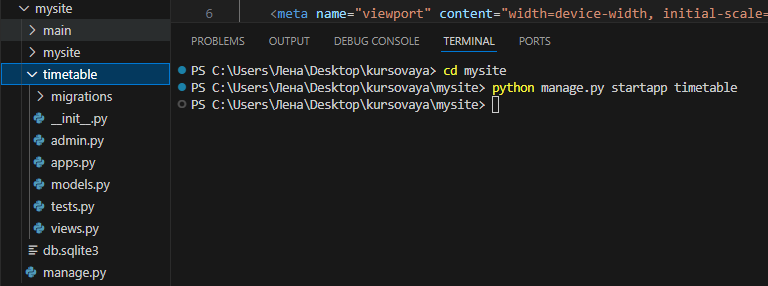


Рисунок 28

Заполним страницу (Рисунок 29). Позже будем добавлять сюда данные из БД, а пока сделаем просто шаблон (Рисунок 30)

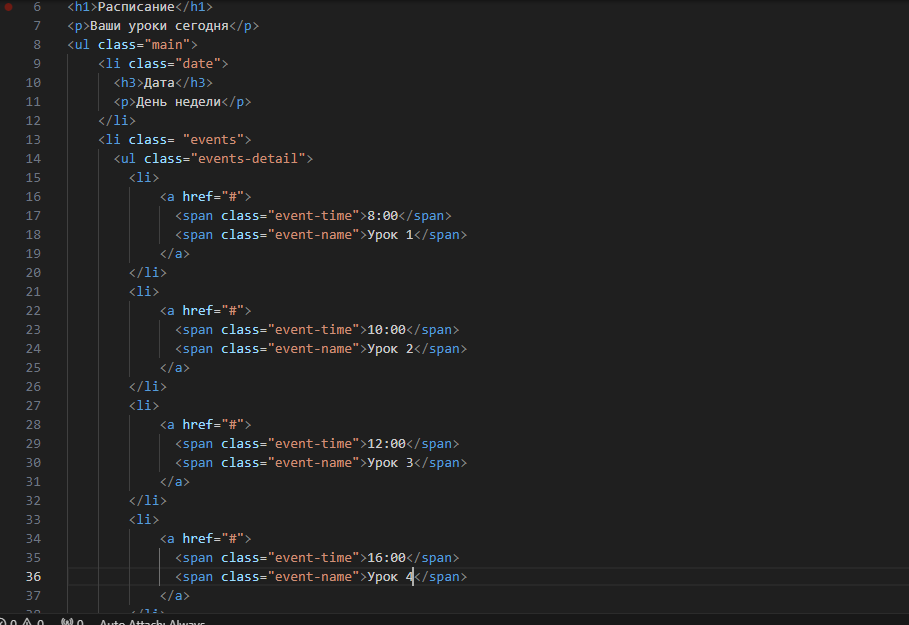


Рисунок 29

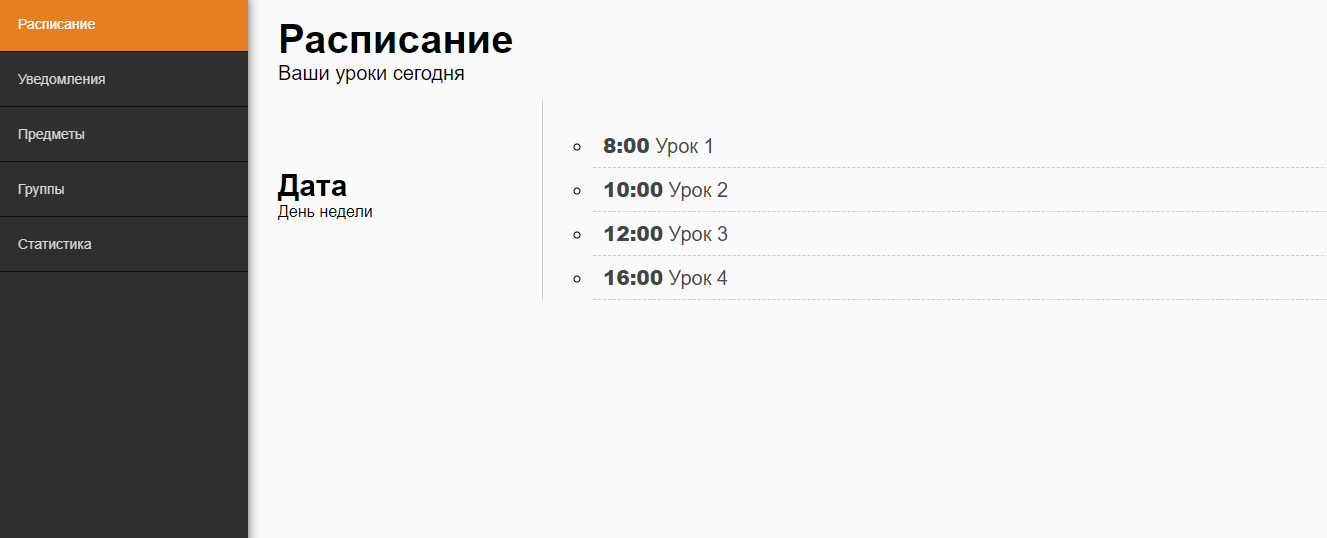
****

Рисунок 30

Теперь можно приступить к созданию первых баз данных. Это будет две небольшие таблицы, в первой будет храниться список предметов, а во второй расписание на неделю,. Для этого заходим в файл models и пишем наши первые модели, при чем свяжем между собой связью один-ко-многим. После этого уже в терминал вводим *команды python manage.py makemigrations* и *python manage.py migrate* (рисунок 31)

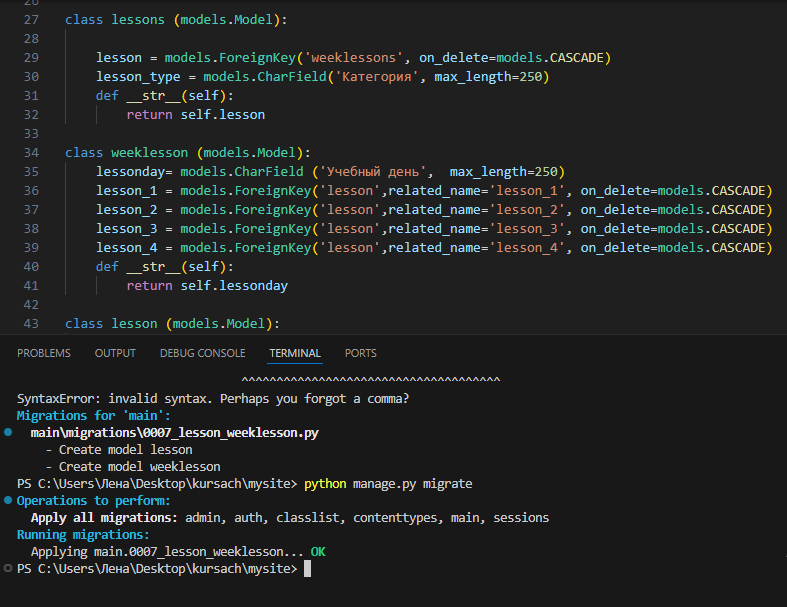


Рисунок 31

Чтобы проверить создалась ли таблица, на нашем сайте нам нужно перейти в панель администратора. Но для этого нам нужно сначала создать логин и пароль администратора. Вводим в терминал *python manage.py createsuperuser* и регистрируем администратора. (Рисунок 32). После этого возвращаемся к панели администратора и входим. Теперь мы видим нашу, пока пустую, таблицу (Рисунок 33)

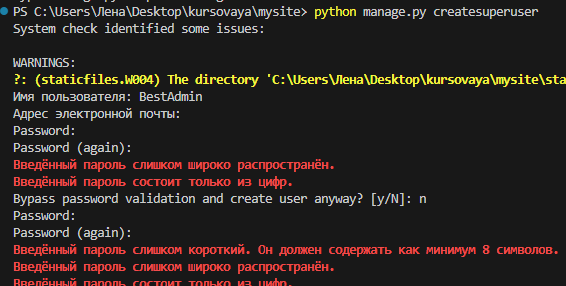


Рисунок 32

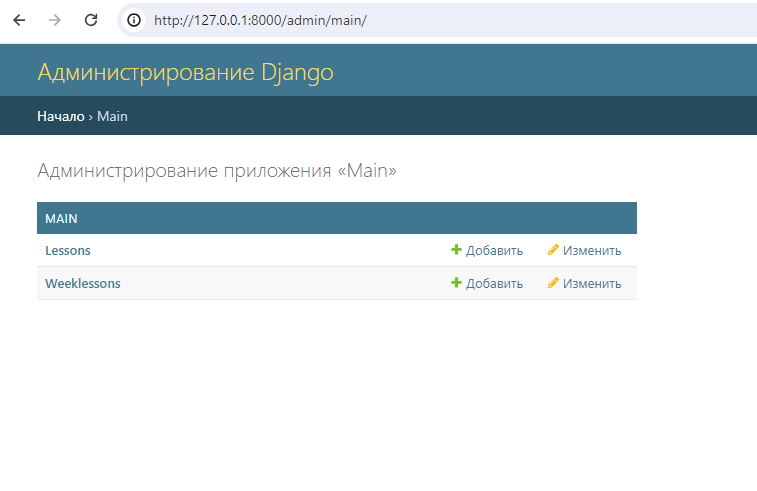
****

Рисунок 33

Заполним нашу таблицу с предметами. Теперь, когда у нас есть список предметов (Рисунок 34), в таблице с расписанием можно выбрать только те предметы, которые есть в первой таблице (рисунок 35)

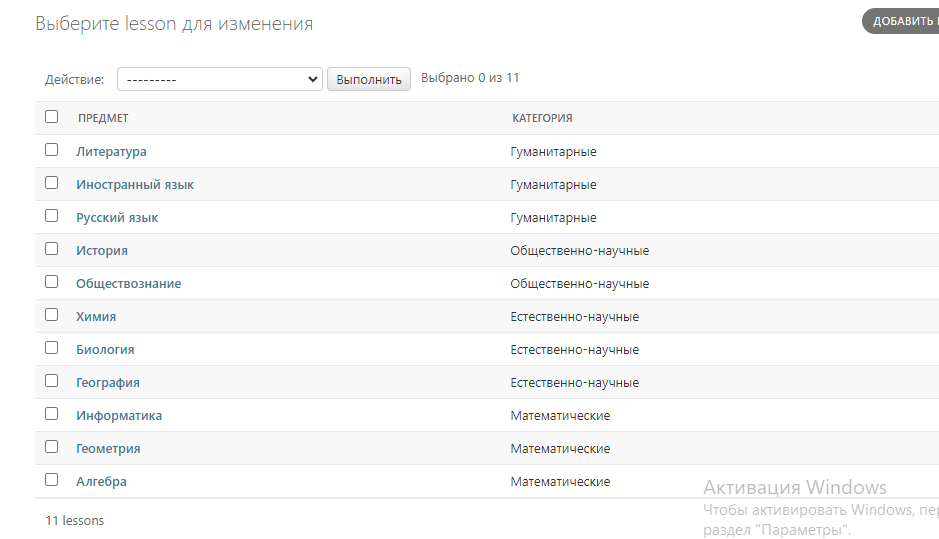


Рисунок 34

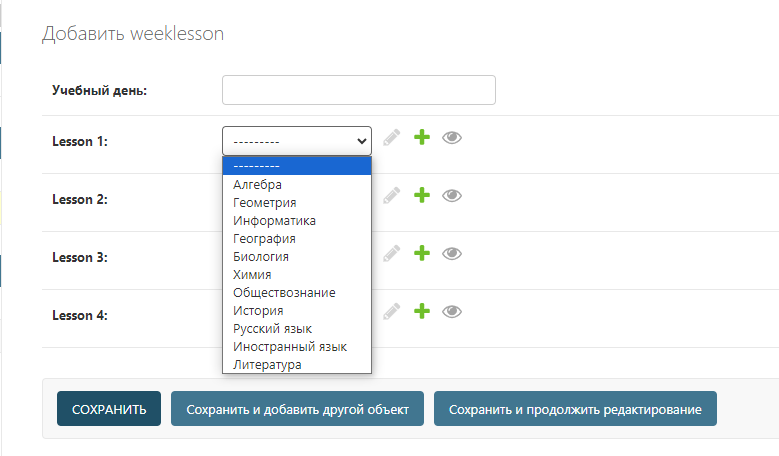


Рисунок 35

Теперь выведем эти данные на нашу страницу. Для этого зайдем в файл views. Сначала импортируем все данные из таблицы, а затем, с помощью специального метода, поместим их в переменную. Создадим еще две переменные отвечающие за сегодняшнюю дату и день недели. Поскольку день недели считается с 0, а данные в нашей БД начинаются с 1, то прибавляем к этому числу единицу. Теперь списком передаем эти данные в html документ. (Рисунок 36)

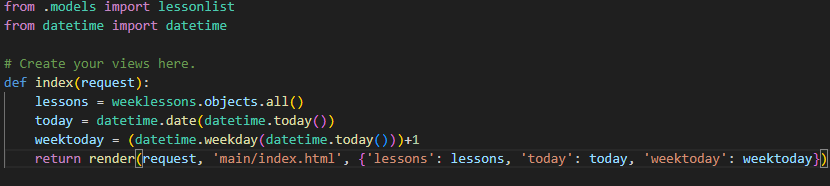


Рисунок 36

Теперь осталось вставить эти данные на страницу сайта. Для этого нужно поместить ключи из переданного списка в данную конструкцию: {{}}.

С помощью цикла for перебираем данные из таблицы и когда ID строки совпадает со днем недели – вставляем нужную ячейку (Рисунок 37). По образцу сделаем расписание и на завтра. В итоге получаем такую страницу. (Рисунок 38)

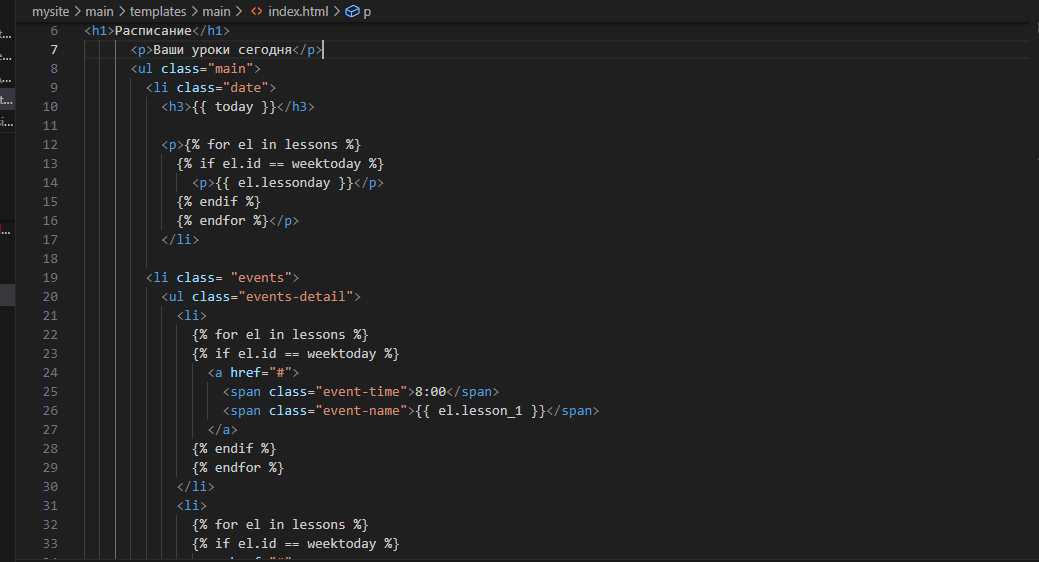


Рисунок 37

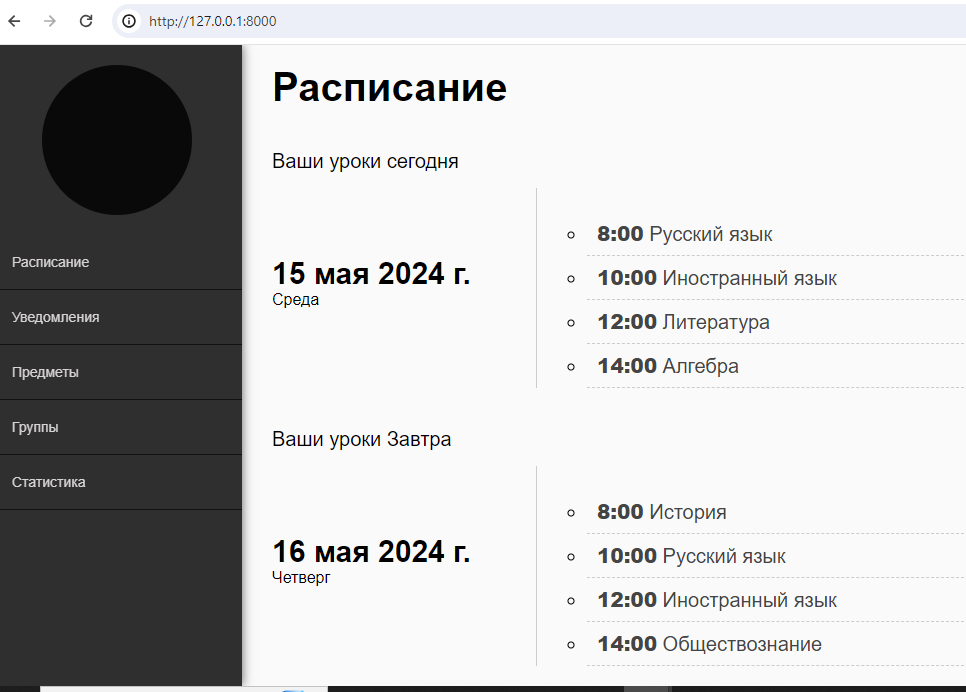


Рисунок 38

Страница с уведомлениями будет максимально простой. На данный момент оставим ее пустой (Рисунок 39)

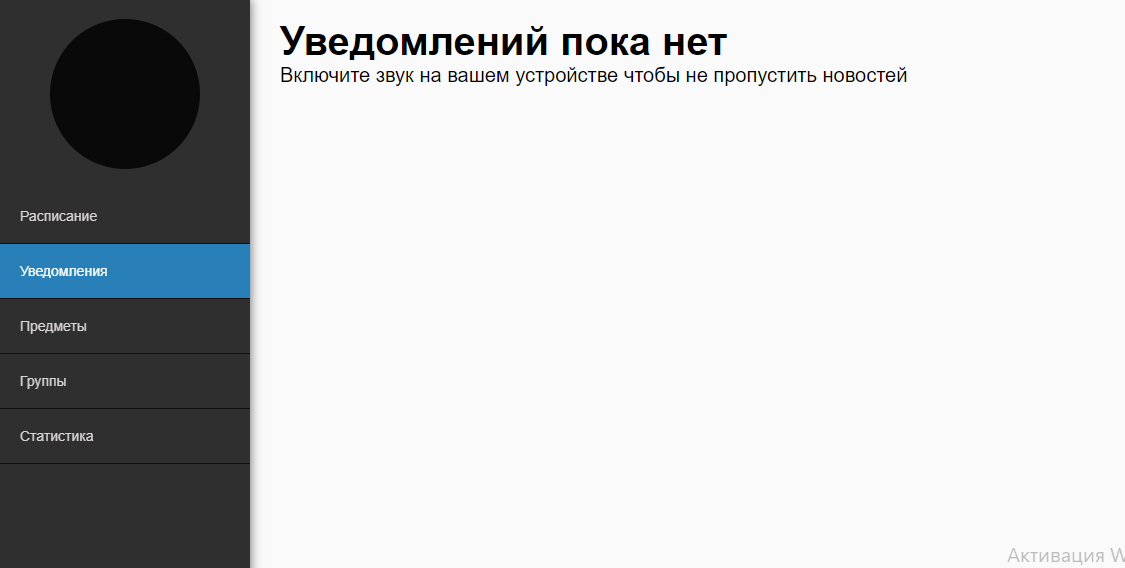
****

Рисунок 39

Создадим еще таблицы, в которых будет содержаться информация о существующих группах и предметах (Рисунок 40 и 41). А данные этих таблиц выведем на специальные страницы. (Рисунок 42 и 43). Так же создадим и подключим JS файл, который будет сортировать информацию в этих таблицах (Рисунок 44)

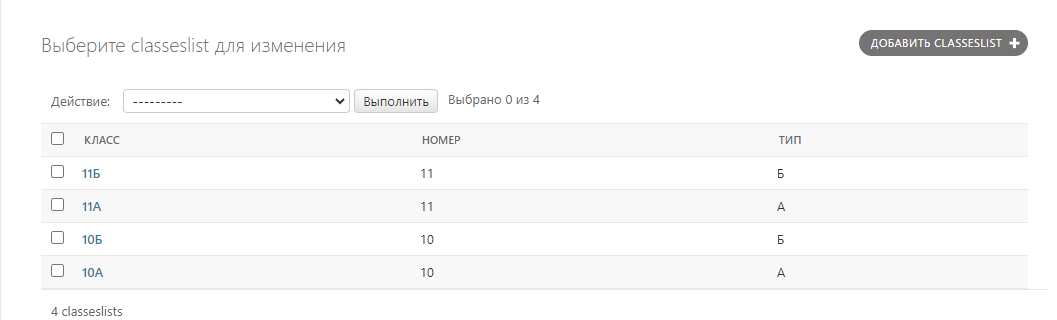


Рисунок 40

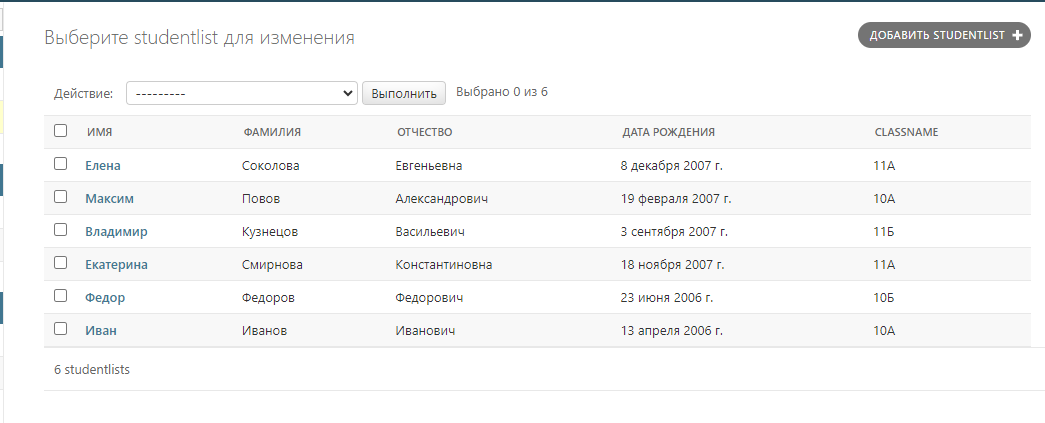


Рисунок 41

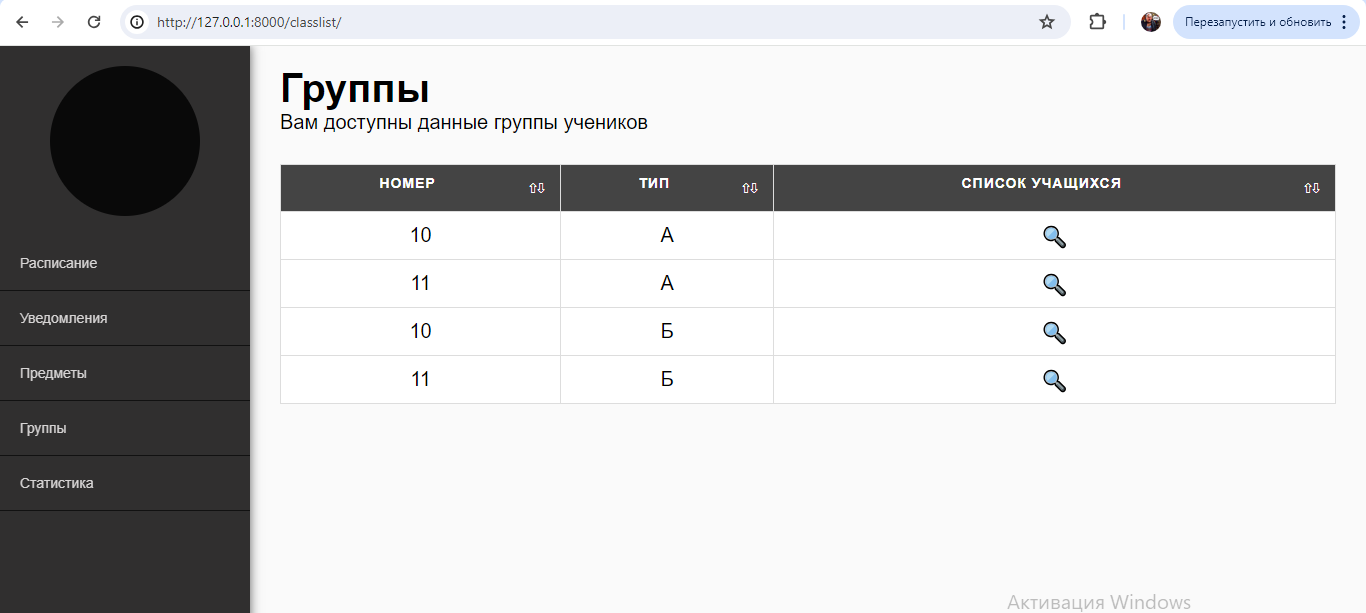


Рисунок 42

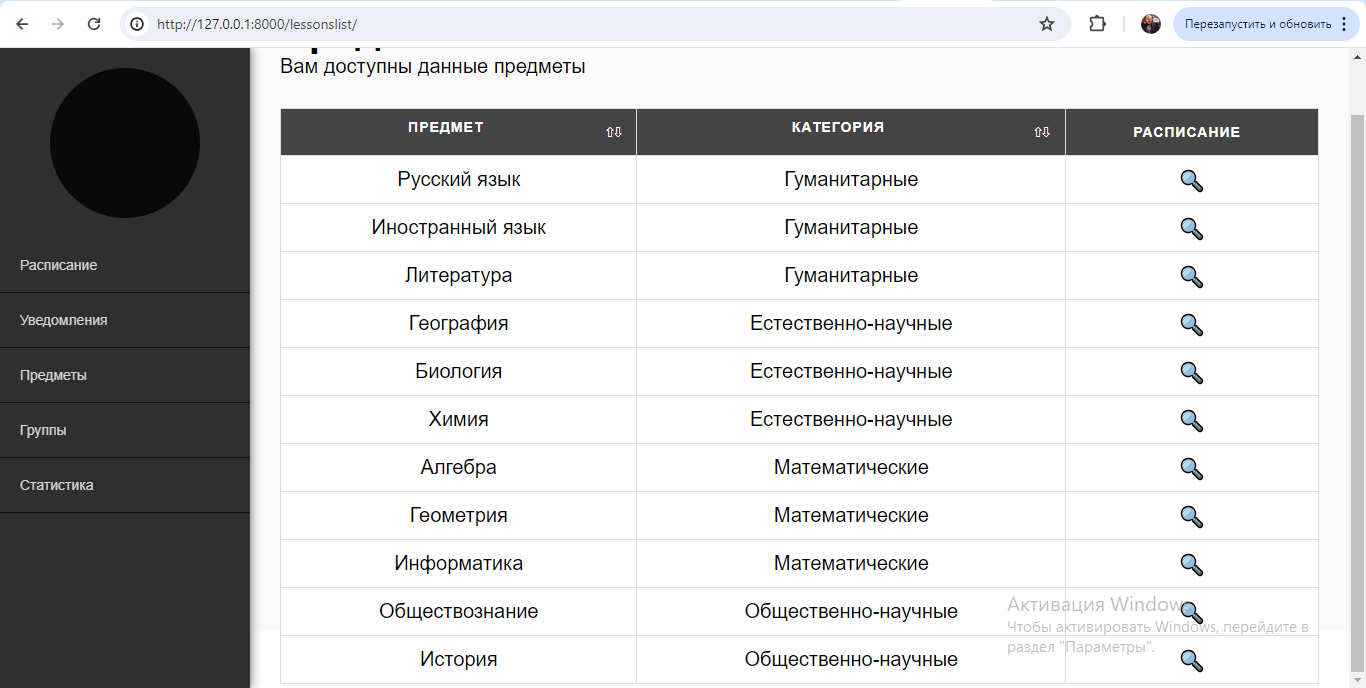


Рисунок 43

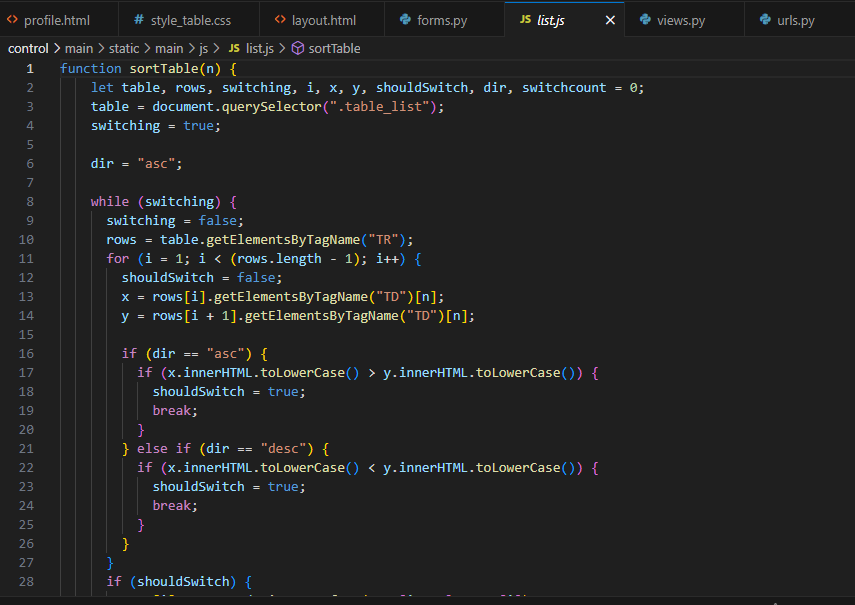


Рисунок 44

Добавим авторизацию пользователей. Создадим для этого новое приложение с названием Users. Внутри него создадим три html шаблона: для авторизации, регистрации и просмотра/редактирования профиля. Так же создадим в приложении users файл forms.py. В нем и начнем создание форм регистрации и авторизации. Там пропишем два класса, которые отвечают за соответствующие формы (Рисунок 45). В файле views пропишем представления страниц, а также дополнительную функцию выхода из аккаунта (рисунок 46). И теперь в html файле выведем эти формы (Рисунок 47 – 48). Получаем такие функциональные страницы (Рисунок 49 - 50)

Рисунок 45

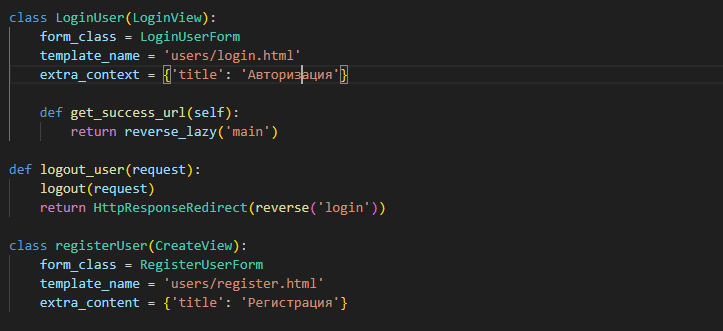
****

Рисунок 46

****

Рисунок 47

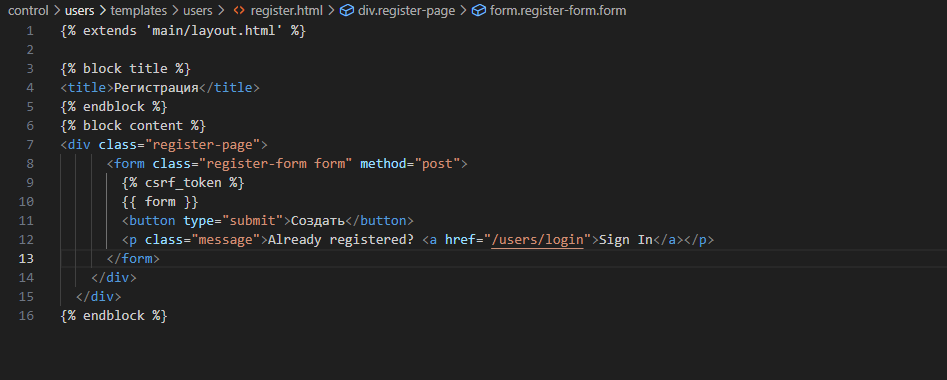
****

Рисунок 48

# 

Рисунок 49

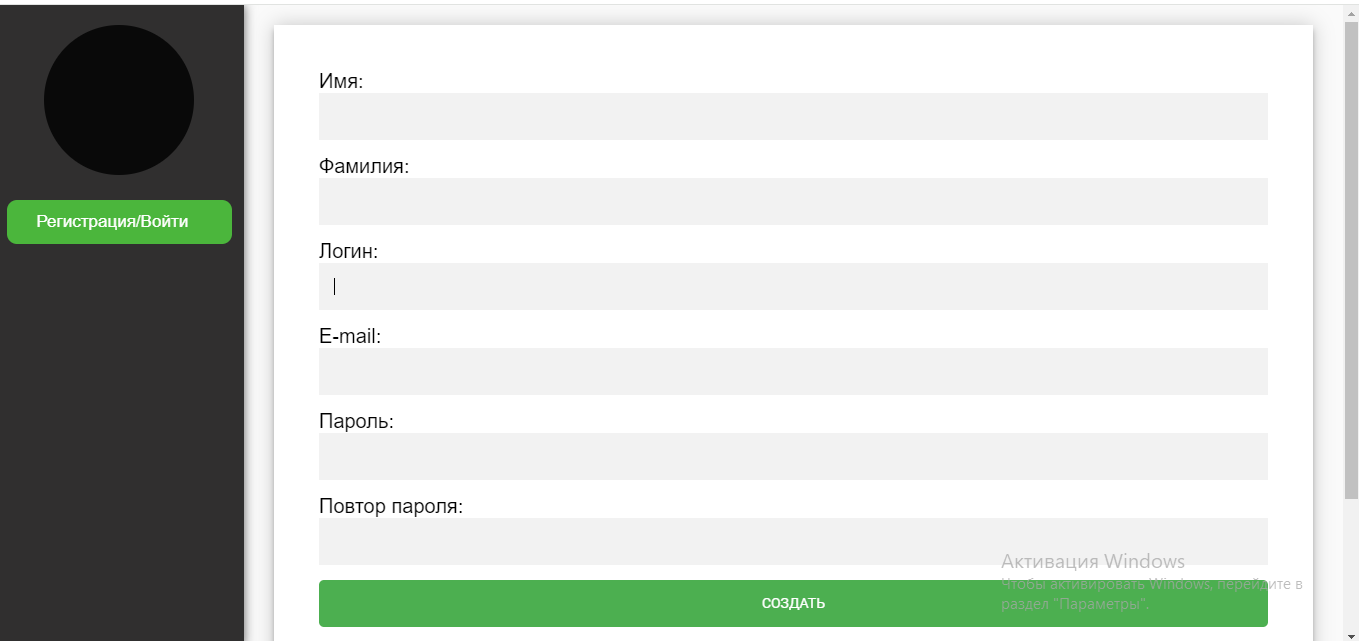


Рисунок 50

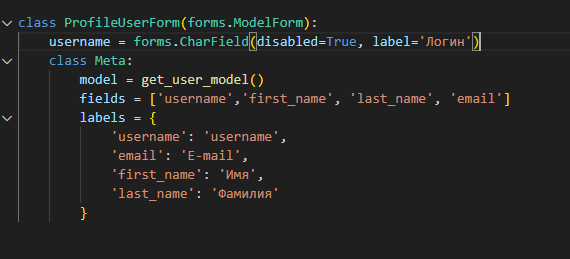
И теперь сделаем страницу просмотра и изменения профиля (Рисунок 51 – 54)

Рисунок 51

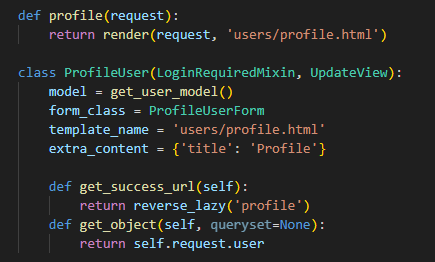


Рисунок 52

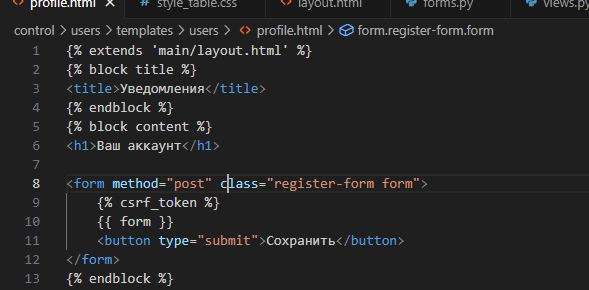


Рисунок 53

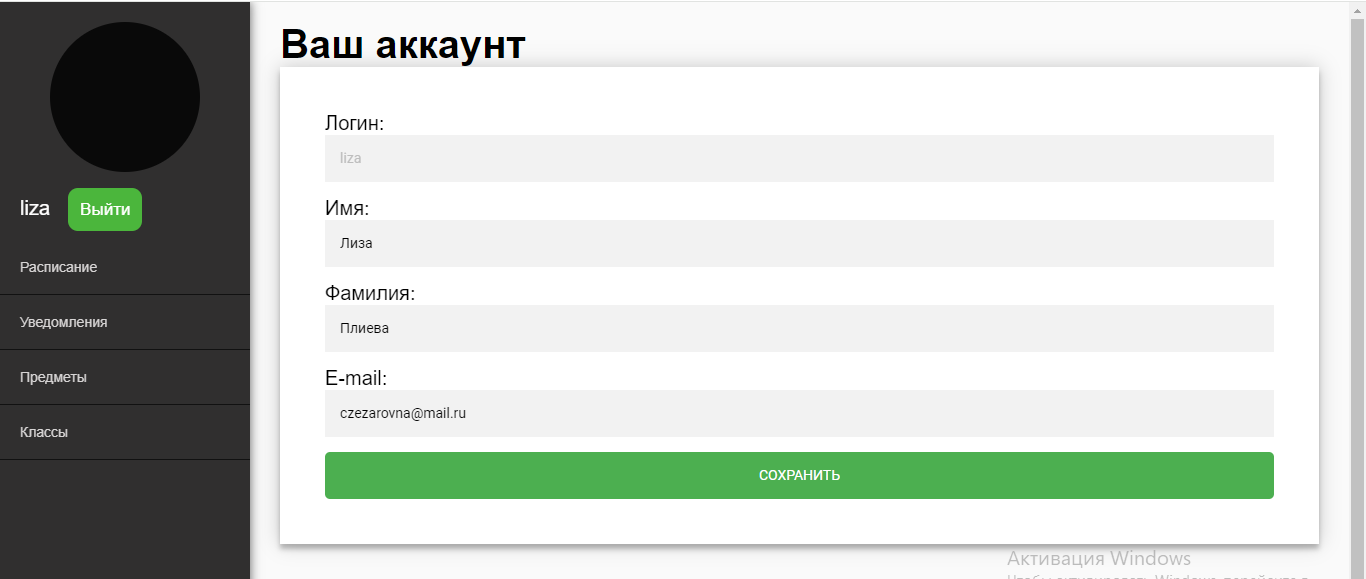


Рисунок 54

И теперь приступим к главной функции – посещаемость. К каждой кнопке на страницах расписание, предметы и классы, мы прикрепим ссылку, которая будет ввести к странице с отметкой посещаемости. Создадим urls маршрут, в views представление и html шаблон нашей страницы под названием ‘checking’. (Рисунок 55). Теперь создадим модель, в которой будет храниться информация о посещении. Поля таблицы следующие: Дата, Ученики, Класс, Урок. При чем поля класс и урок будут соединены с соответствующими таблицами с помощью ForeignKey (связь один-ко-многим). А вот поле ученики соединим с таблицей с учениками связью многие-ко-многим. Это нужно чтобы за один урок мы могли отметить сразу несколько учеников. (Рисунок 56-57). Теперь в папке с формами пропишем новую, причем связанную с нашей моделью. (Рисунок 58). В файле views пропишем обработку нашей формы (Рисунок 59) и можем перенести ее на саму страницу (рисунок 60-61)

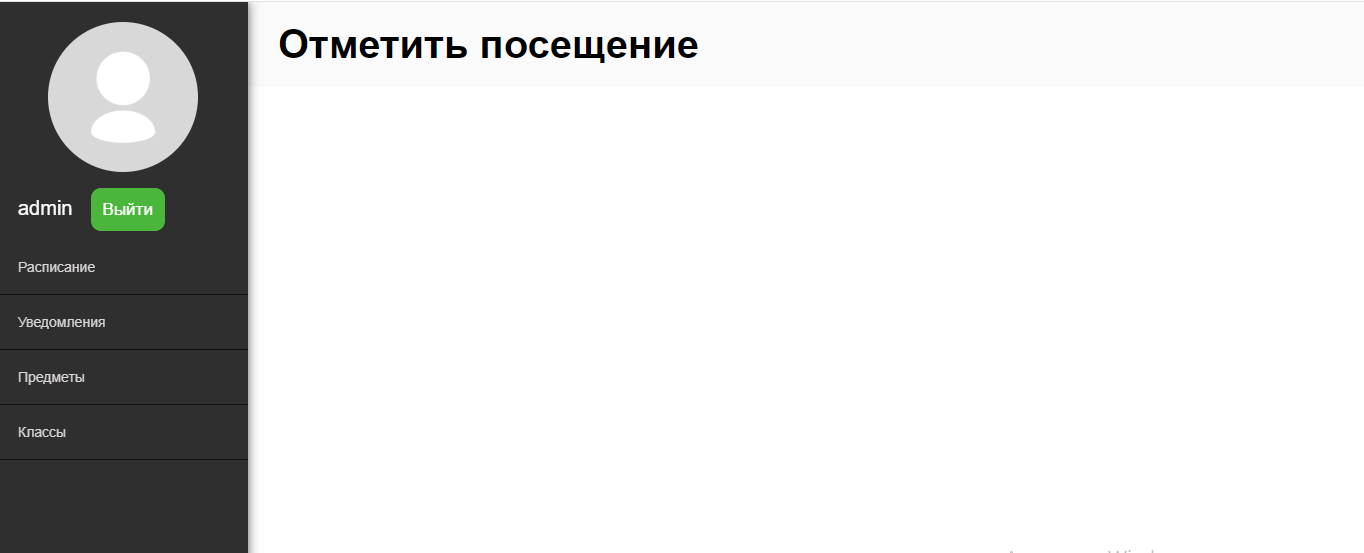


Рисунок 55

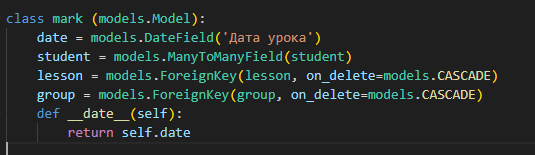
****

Рисунок 56

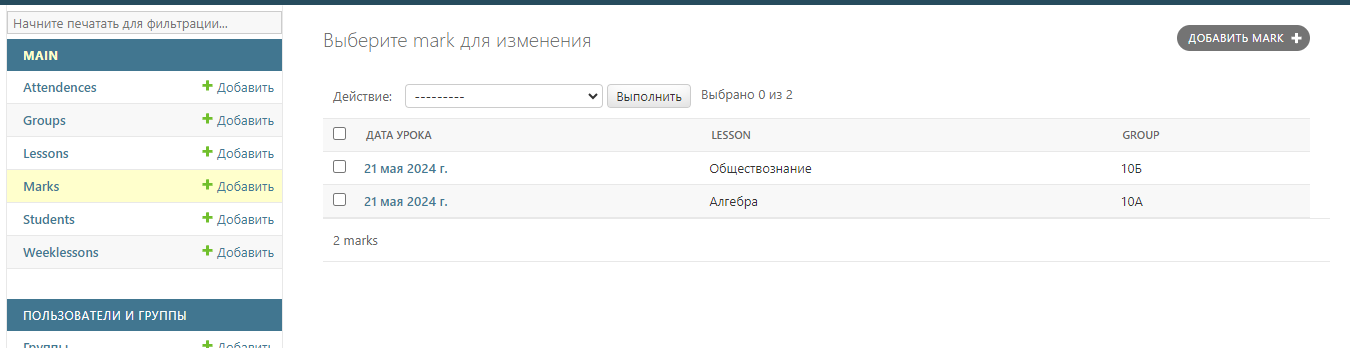
****

Рисунок 57

# 

Рисунок 58

# 

Рисунок 59

# 

Рисунок 60

# 

Рисунок 61

И теперь наш сайт функционально готов! По желанию можно добавить больше стилей, поменять оформление под свой вкус. Вот такой сайт мы получим в итоге (Рисунок 62 – 70)

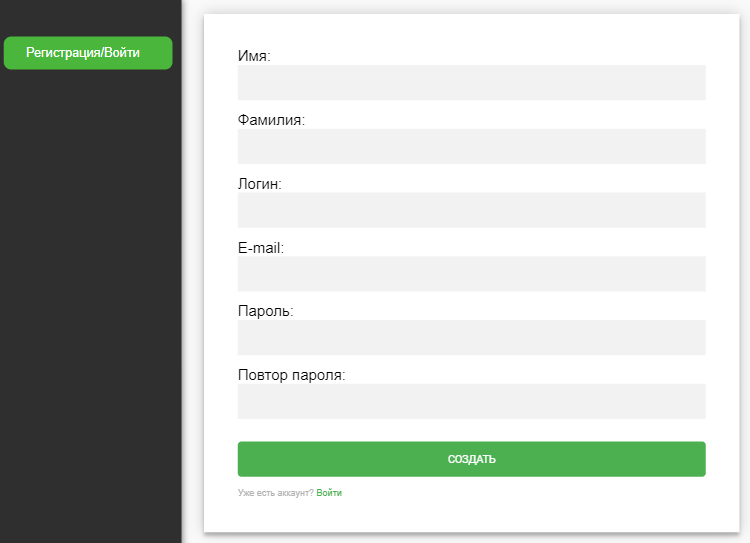


Рисунок 62

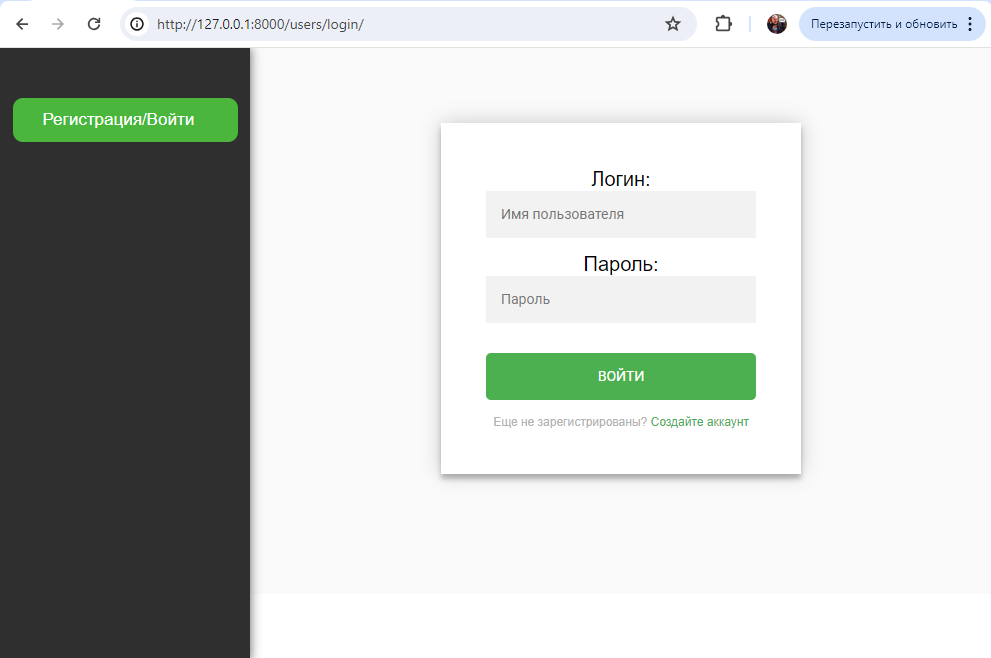


Рисунок 63

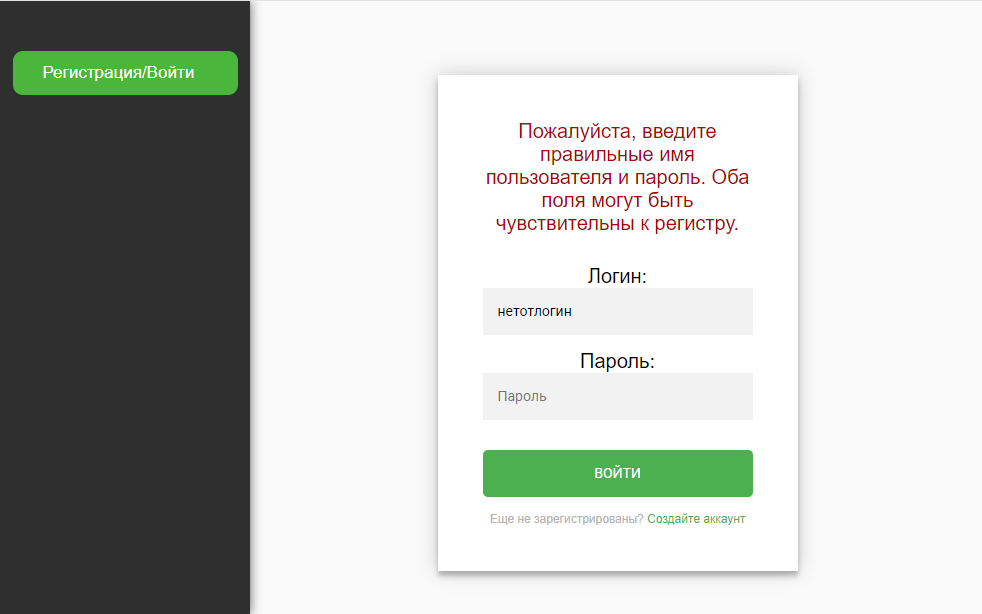


Рисунок 64

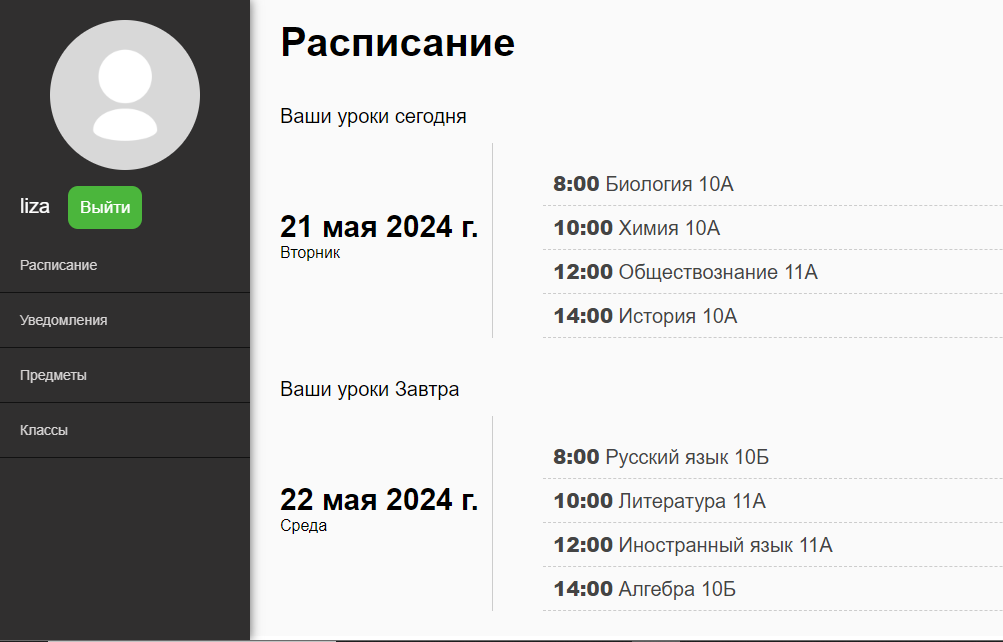


Рисунок 65

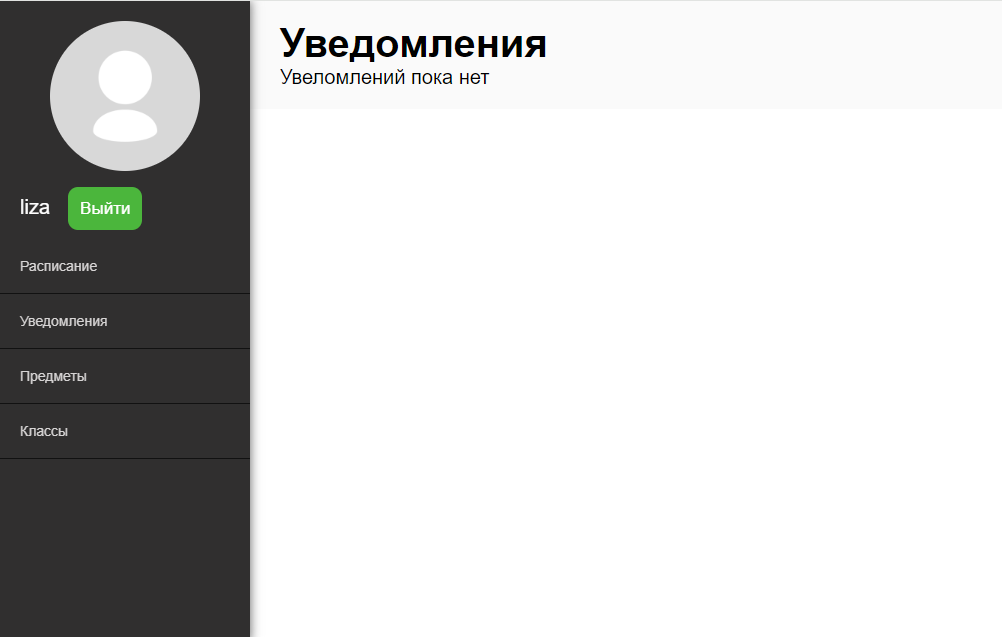


Рисунок 66

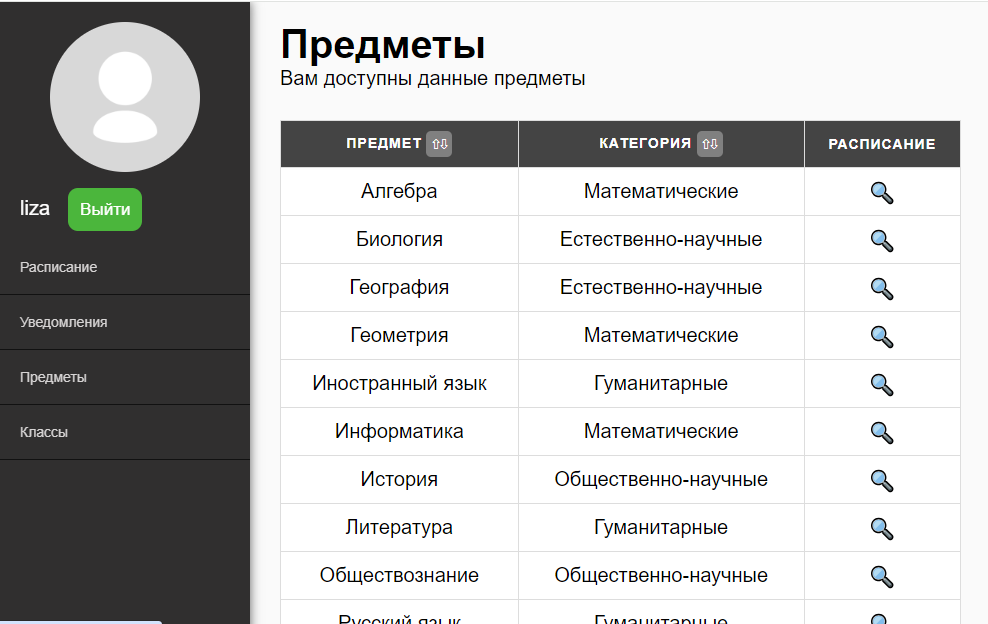


Рисунок 67

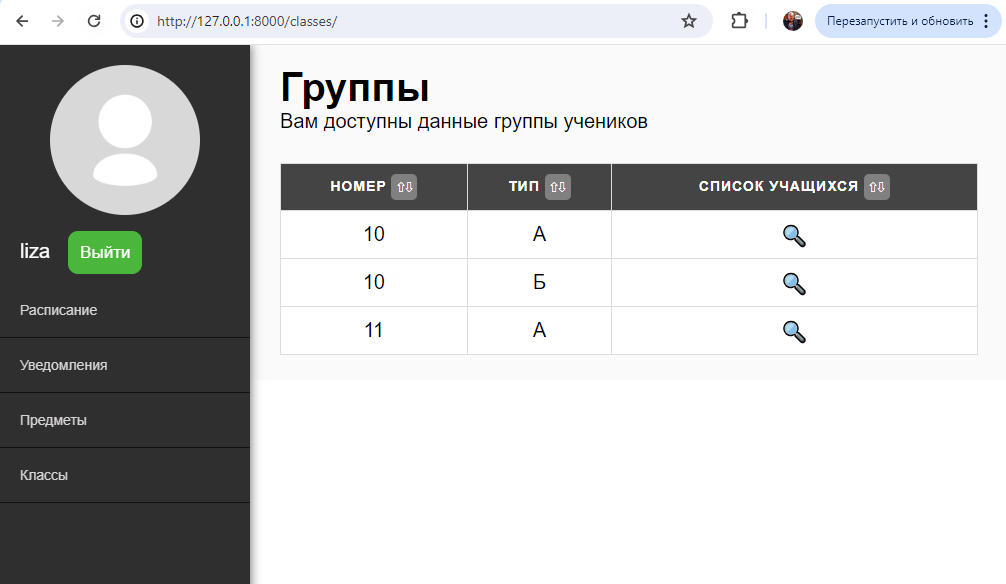


Рисунок 68

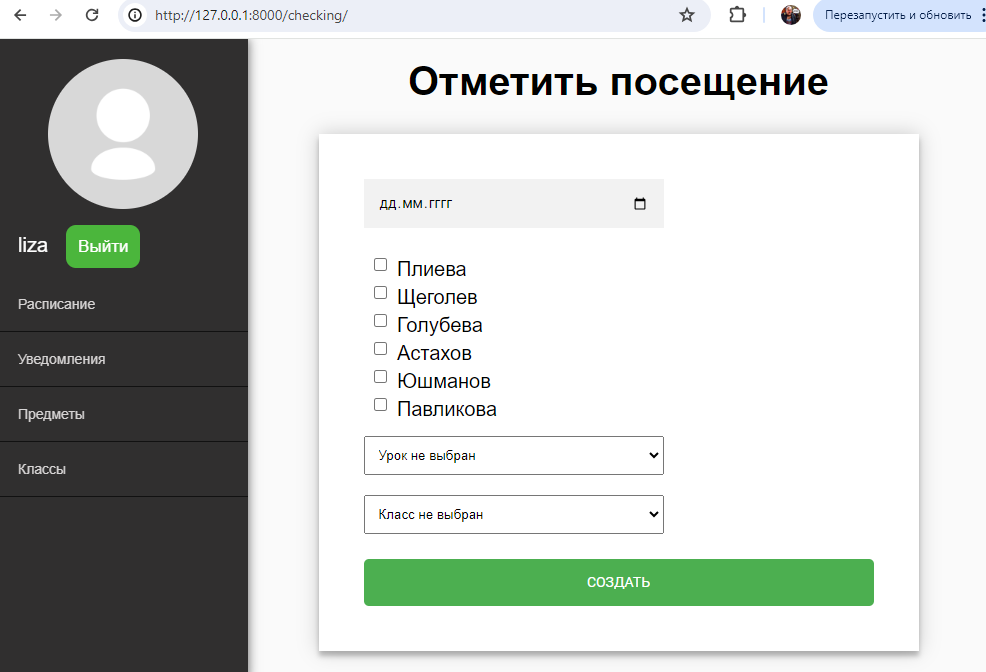


Рисунок 69

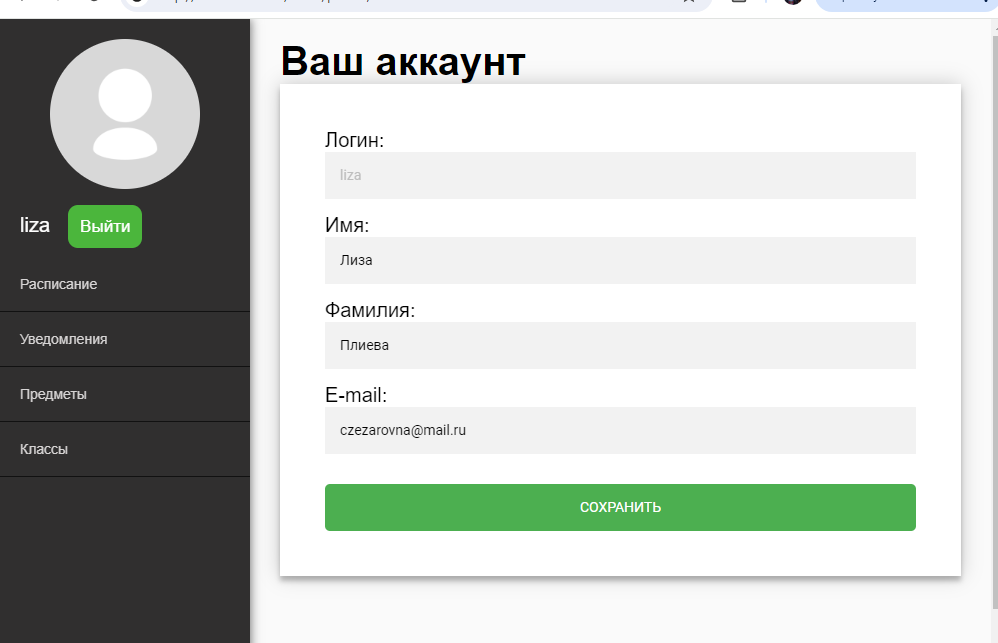


Рисунок 70

# **Заключение**

В заключение, разработка информационной системы по контролю посещаемости учащихся является важным шагом в современном образовании. Такая система позволит эффективно отслеживать посещаемость студентов, повышая общую дисциплину и ответственность. Кроме того, она способствует более точному анализу данных о посещаемости, что поможет улучшить качество образования и своевременно выявлять проблемные моменты. Внедрение данной системы не только оптимизирует учебный процесс, но и способствует развитию цифровизации в образовании.

Результаты работы:

1. Изучили основные понятий баз данных: В рамках этого этапа мы изучали основные концепции баз данных, такие как таблицы, столбцы, строки, ключи (первичные, внешние), основные типы баз данных (реляционные, иерархические и т.д.), свойства и особенности
2. Изучили системы управления базами данных (СУБД): Здесь мы ознакомились с различными системами управления базами данных. Изучили их функциональности, возможностей, особенностей и специфики использования. Изучили язык SQL: SQL (Structured Query Language) — это язык запросов, используемый для работы с данными в реляционных базах данных.
3. Освоение HTML, CSS и JavaScript: HTML (HyperText Markup Language), CSS (Cascading Style Sheets) и JavaScript являются основными технологиями веб-разработки. Мы изучили создание структуры веб-страниц с помощью HTML, стилизацию элементов с помощью CSS и добавление интерактивности с помощью JavaScript.
4. Использование Visual Studio Code: Visual Studio Code — это популярная среда разработки, которая предоставляет широкие возможности для написания кода, отладки, управления расширениями и интеграции с различными инструментами разработки.
5. Создание сайта с функцией контроля посещаемости: Наш проект по созданию сайта с функцией контроля посещаемости вероятно включал в себя разработку интерфейса для пользователей, функционал для отметки посещений или регистрации пользователей, а также использование базы данных для хранения информации о посещениях.

# **Используемая литература**

**Нормативные документы:**

Конституция Российской Федерации (с учетом поправок, внесен­ных за­конами Российской Федерации «О поправках к Конституции Россий­ской Федерации» от 30 декабря 2008 г. № 6-ФКЗ и от 30 декабря 2008г. № 7-ФКЗ) // Российская газета № 4831 от 21 января 2009 г.

Федеральный закон Российской Федерации от 7 февраля 2011 № 3-ФЗ «О полиции»// Россий­ская газета от 8 февраля 2011 № 5401.]

**Основные источники:**

Безруков В.А. Базы данных и SQL. Учебное пособие. — М.: Книга по Требованию, 2020. — 240 с.

Гриценко А.Ю. Django для профессионалов: полное руководство по веб-разработке на Python. — СПб: БХВ-Петербург, 2019. — 672 с.

Дмитриев А.А. HTML и CSS. Путь к совершенству. — М.: ДМК Пресс, 2018. — 384 с.

Закас Н. JavaScript. Подробное руководство. — М.: Вильямс, 2016. — 1008 с

Иваненко О.Ю. Python 3 и Flask. Создание веб-приложений. — СПб: Питер, 2018. — 464 с.

Кэрнигэ Ч. Базы данных. Проектирование и реализация. — М.: ДМК Пресс, 2019. — 336 с.

Локкетт Э., Эткинс Дж. SQL: Справочник. — СПб: Питер, 2017. — 448 с.

Маккинли Дж. Python и машинное обучение: изучение Python, NumPy, pandas, Matplotlib и другие библиотеки для обучения с учителем. — М.: ДМК Пресс, 2019. — 224 с.

Николенко С.И. HTML и дизайн. Подробное руководство. — М.: ДМК Пресс, 2017. — 320 с.

Палетски Д. Книга о Django. Подлинное руководство для профессионалов. — М.: Издательский дом Диалектика, 2019. — 512 с.

Рубио Х., Хесс К. Базы данных SQL и реляционная теория. — М.: Бином, 2019. — 424 с.

Страуструп Дж. Python 3: Программирование на Python. — СПб: Символ-Плюс, 2017. — 656 с.

Уэльш А., Диксон Л. CSS. Рецепты программирования. — СПб: БХВ-Петербург, 2016. — 448

Фланаган Д. JavaScript. Подробное руководство. — М.: Вильямс, 2019. — 832 с.  
 Чэнг В. Visual Studio Code. Создание приложений. – СПб: БХВ-Петербург, 2018. – 288 с.

**Электронные ресурсы:**

[https://www.oracle.com/cis/database/what-is-database](https://www.oracle.com/cis/database/what-is-database/#:~:text=%D0%91%D0%B0%D0%B7%D0%B0%20%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85%20%E2%80%94%20%D1%8D%D1%82%D0%BE%20%D1%83%D0%BF%D0%BE%D1%80%D1%8F%D0%B4%D0%BE%D1%87%D0%B5%D0%BD%D0%BD%D1%8B%D0%B9%20%D0%BD%D0%B0%D0%B1%D0%BE%D1%80,%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F%20%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8%20%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85%20(%D0%A1%D0%A3%D0%91%D0%94).).