МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

|  |  |  |  |
| --- | --- | --- | --- |
| Кафедра | *Вычислительной техники* | | |
|  | (полное название кафедры) | | |
|  |  | Утверждаю | |
| **Зав. кафедрой** | *Якименко А.А.* |
|  | |
| (подпись, инициалы, фамилия) | |
| «16» марта 2024 г. | |

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА**

|  |  |
| --- | --- |
| *Еремеева Виктория Владимировна* | |
| (фамилия, имя, отчество студента – автора работы) | |
| *Разработка приложения "Расписание"* | |
| (тема работы) | |
|  | |
| *Факультет автоматики и вычислительной техники* | |
| (полное название факультета) | |
| Направление подготовки | *Информатика и вычислительная техника 09.03.01* |
|  | (код и наименование направления подготовки бакалавра) |
| *(очная форма)* | |
|  | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Руководитель**  **от НГТУ** |  |  |  | **Автор выпускной квалификационной работы** |
| *Осипенко И.В.* |  |  |  | *Еремеева В.В.* |
| (фамилия, имя, отчество) |  |  |  | (фамилия, имя, отчество) |
|  |  |  |  | *АВТФ, АВТ-009* |
| (ученая степень, ученое звание) |  |  |  | (факультет, группа) |
|  |  |  |  |  |
| (подпись, дата) |  |  |  | (подпись, дата) |

|  |
| --- |
|  |

Новосибирск 2024

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

|  |  |  |  |
| --- | --- | --- | --- |
| Кафедра | *Вычислительной техники* | | |
|  | (полное название кафедры) | | |
|  |  | УТВЕРЖДАЮ | |
| Зав. кафедрой | *Якименко А.А.* |
| (фамилия, имя, отчество) | |
|  |  |
|  |  |
|  | (подпись, дата) |

**ЗАДАНИЕ  
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ БАКАЛАВРА**

|  |  |  |  |
| --- | --- | --- | --- |
| студенту | *Еремеевой Виктории Владимировне* | | |
|  | (фамилия, имя, отчество) | | |
| Направление подготовки | | *Информатика и вычислительная техника 09.03.01* | |
|  | | (код и наименование направления подготовки бакалавра) | |
|  | | | |
|  | | | |
| *Факультет автоматики и вычислительной техники* | | | |
| (полное название факультета) | | | |
| Тема | *Разработка приложения "Расписание"* | | |
|  | (полное название темы выпускной квалификационной работы бакалавра) | | |
|  | | | |
| Исходные данные (или цель работы) | | | *Целью работы является разработка* |
| *веб-приложения "Расписание" для генерации и визуализации расписания уроков* | | | |
| *в школе* | | | |
| *ПО: Python3, PyCharm, PostgreSQL, Docker, Django* | | | |
|  | | | |
|  | | | |
| Структурные части работы | | |  |
| *Введение* | | | |
| 1. *Анализ существующих решений* | | | |
| 1. *Особенности разработки приложения* | | | |
| 1. *Реализация и тестирование* | | | |
| *Заключение* | | | |
|  | | | |

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

Задание согласовано и принято к исполнению.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Руководитель**  **от НГТУ** |  |  |  | **Студент** |
| *Осипенко И.В.* |  |  |  | *Еремеева В.В.* |
| (фамилия, имя, отчество) |  |  |  | (фамилия, имя, отчество) |
|  |  |  |  | *АВТФ, АВТ-009* |
| (ученая степень, ученое звание) |  |  |  | (факультет, группа) |
|  |  |  |  |  |
| (подпись, дата) |  |  |  | (подпись, дата) |

|  |  |  |  |
| --- | --- | --- | --- |
| Тема утверждена | приказом по НГТУ № | ㅤㅤㅤㅤ | от «ㅤㅤ» ㅤㅤㅤㅤㅤㅤ 2024 г. |
| изменена | приказом по НГТУ № | ㅤㅤㅤㅤ | от «ㅤㅤ» ㅤㅤㅤㅤㅤㅤ 2024 г. |

| ВКР сдана в ГЭК №ㅤㅤㅤㅤ, тема сверена с данными приказа | |
| --- | --- |
|  |  |
|  | (подпись секретаря государственной экзаменационной комиссии по защите ВКР, дата) |
|  |  |
|  | (фамилия, имя, отчество секретаря государственной  экзаменационной комиссии по защите ВКР) |

Реферат

В пояснительной записке к выпускной квалификационной работе

бакалавра содержится 61 страница, 17 иллюстраций, 4 таблицы и 17

использованных литературных источников.

В пояснительной записке фигурируют понятия: фонема, мел-кепстральный

коэффициент, нейронная сеть, LSTM, python, keras, web-сервис, REST API,

JSON, Docker.

Целью данной работы является разработка системы для автоматизации

синхронизации движений рта персонажа с озвучкой.

Структура разработанной системы представляет собой клиентское

приложение, а также web-сервис. Оба приложения взаимодействуют между

собой по протоколам http.

Пользователи взаимодействуют с клиентским приложением.

Клиентское приложение самодостаточно и исполняет всю бизнес-логику.

Web-сервис отвечает за распознавание фонем для клиентского приложения из

переданного звукового файла.

Содержание

[Глоссарий 10](#_Toc168411025)

[Введение 11](#_Toc168411026)

[1 Анализ существующих решений 13](#_Toc168411027)

[1.1 Существующие программные аналоги 13](#_Toc168411028)

[1.2 Раскраска графов 16](#_Toc168411029)

[1.3 Генетические алгоритмы 18](#_Toc168411030)

[2 Особенности разработки 20](#_Toc168411031)

[2.1 Общие архитектурные положения 21](#_Toc168411032)

[2.2 Модель прецендентов 21](#_Toc168411033)

[2.3 Структура проекта 22](#_Toc168411034)

[2.4 Диаграмма классов 22](#_Toc168411035)

[2.5 Диаграмма компонент 22](#_Toc168411036)

[2.6 Алгоритмы написания программы 23](#_Toc168411037)

[3 Реализация и тестирование 25](#_Toc168411038)

[3.1 Руководство пользователя 25](#_Toc168411039)

[3.2 Развёртка в контейнере Docker 25](#_Toc168411040)

[Заключение 26](#_Toc168411041)

[Список литературы 27](#_Toc168411042)

[Приложение A. Сводная таблица аналогов 28](#_Toc168411043)

[Приложение B. Схема Rest API 29](#_Toc168411044)

[Приложение C. Код создания модели прецендентов 30](#_Toc168411045)

[Приложение D. Диаграмма прецендентов 32](#_Toc168411046)

[Приложение E. Шкала трудности учебных предметов 33](#_Toc168411047)

[Приложение F. Рейтинговая шкала трудности предметов 34](#_Toc168411048)

**проверь в итоге должно ли нумероваться и что там имелось в виду**

# Глоссарий

желательно таблицей и нумерация

Декартово произведение

Кроссинговер

Окна

Фреймворк

Мутация

Django

MVC

MTV

CMS

API

UI

UX

JSON

XML

AJAX

Middleware

OpenAPI

Планарные графы

Петли

Хроматический многочлен

# Введение

В современном мире информация играет ключевую роль в эффективности любой организации. Растёт количество информациии. Процессы её обработки становится невозможно выполнять вручную. Системы требуют компьютеризации для повышения скорости и недопущения ошибок.

В образовательной сфере, создание информационной системы становится критически важным. Образовательная среда требует инноваций, включая использование общей базы данных для диспетчеризации режимной части школы.

Сегодняшние школы используют информационные системы более интенсивно и постоянно. Они активно заняты не только в урочное время, но и во внеучебное, проводя различные кружки и сотрудничая с внешними организациями, что способствует развитию учебного заведения, повышению его эффективности. Увеличилась численность обучающихся и преподавателей, при этом учителя теперь меньше загружены.

Из вышесказанного следует, что одной из первостепенных задач современной системы образования является автоматизация процессов расписания.

В обычной практике, данные для составления расписания заносятся разными людьми в таблицу Excel (учебный план, нагрузка), которая затем пересылается по электронной почте друг другу. Расписание обычно создается на миллиметровке с учителями в строках, днями и номерами уроков в столбцах. Затем это расписание переносится в таблицу Excel с классами в столбцах и днями, номерами уроков в строках. В процессе этого переноса часто обнаруживаются наложения и «окна», которые затем требуется исправлять. Процесс составления расписаний является очень трудоёмким.

Изложенные проблемы подчеркивают актуальность создания приложения «Расписание занятий» для образовательных учреждений. Автоматизация этого процесса поможет сократить трудозатраты и уменьшить вероятность ошибок, повышая эффективность учебного процесса. А также может помочь множество форм представления информации с возможностью выбора удобной конкретным пользователем.

**Целью** моей работы является проектирование пользовательского интерфейса приложения «Расписание». Для этого я поставила перед собой следующие **задачи**:

* Ознакомиться с организационной структурой СКБ, ознакомиться с организацией образовательного и технологического процесса, направлениями научной и научно-исследовательской работы.
* Провести анализ существующих решений.
* Разработать архитектуру приложения; создать модели; написать шаблоны страниц; оформить стили проекта.
* Определить функционала.
* Произвести тестирование интерфейса и функциональное тестирование.

аннотация

актуальность

проверить нумерацию

номера из глоссария проставить по работе

расставить ссылки на литературу

приложение нумеровать иначе и всё подписывать

# Анализ существующих решений

## Существующие программные аналоги

Существует множество программных решений для автоматизации составления расписания в образовательных учреждениях, каждое из которых обладает своими особенностями и преимуществами. Программные решения для управления расписанием в образовательных учреждениях представлены в виде десктопных приложений, предназначенных, как правило, для установки на одну машину. Все перечисленные программы способны составлять расписание, поддерживать его корректировки, работать с несколькими сменами и различными предметами у учителей. Многие поддерживают отчётность или экспорт информации в различные удобные форматы типа Word, Excel. Различия могут быть как в функциональности, так и в цене и удобстве использования. При выборе программы следует учитывать индивидуальные потребности и особенности образовательного учреждения. Сводную таблицу можно посмотреть в приложении.

1. Экспресс-расписание Колледж [3]

* Есть проверка аудиторий на вместимость в связи с возникшей проблемой COVID-19.
* Цена: 2 500 – 300 000 руб.

1. 1С:Автоматизированное составление расписания.Колледж [2]

* Преимуществом является учёт времени переходов между корпусами.
* Решение из Новосибирска.
* 70 заведений.
* Цена: не указана на сайте.

1. Система «АВТОРасписание» [10]

* 300 заведений.
* Развивается с 1996 года.
* Наличие подробной статистики и объективной оценки качества любого варианта расписания.
* Возможность поддержки любого национального языка (по желанию клиента).
* Цена: 20 000 – 150 000.

1. Хронограф [4]

* Трудно заполнять таблицы и вносить данные.
* Не интуитивно-понятный интерфейс.
* Сомнительное качество и мало возможностей.
* Цена: бесплатно

1. «НИКА-Колледж» [5]

* Возможности сдваивания и страивания.
* Экспорт в Exel.
* Учёт переходов между корпусами.
* Учёт пожеланий учителей и совместителей.
* Лицензия на 5 лет и помесячно (1, 3, 6).
* Возможность докупать отдельные модули.
* Цены: 11 000 – 25 000 на 5 лет. Демо ~2 500.

1. aSc [6]

* Зарубежная программа.
* По описанию имеет хорошее качество.
* Цена: 150 – 2 000 € в год ~ 15 000 – 200 000 рублей.

Рассмотренные выше сервисы обладают рядом достоинств и недостатков.

Достоинства:

* позволяют быстро создавать расписания с минимальными усилиями;
* могут учитывать различные ограничения и требования, что позволяет создавать более точные расписания;
* позволяют легко изменять расписания при изменении условий и требований;
* могут повысить эффективность работы организации;

Недостатки:

* правильная настройка может потребовать времени и усилий;
* могут не учитывать все особенности конкретной ситуации, что может привести к некачественному расписанию;
* если система перестанет работать, это может привести к проблемам с расписанием;
* некоторые системы могут быть довольно дорогими, особенно если они требуют постоянного обновления и поддержки.
* существующие решения не имеют доступа в интернет и не привязаны к образовательным платформам.

Естественно стремление каждого разработчика устранить существующие недостатки при реализации своей системы с сохранением достоинств других. И я не являюсь исключением. Особенно меня привлекла идея, чтобы моя программа учитывала возможность автоматического переноса на сайт школы и встраиваемость в образовательные платформы типа «Дневник.ру», «Электронная школа» и другие. Не нравятся местоимения

## Раскраска графов

Раскраску графов применяют для многих типов задач, таких задач как раскраска географических карт, оптимизация каких-то процессов, имеющие общие ресурсы, в том числе формирование расписания. При раскраске графов важно, чтобы никакие две смежные вершины не были раскрашены в один и тот же цвет. При этом количество цветов должно быть минимальным.

Рассмотрим применение теории графов для решения задачи расписания на примере. Пусть нам нужно составить расписание для двух классов на понедельник. Для этого способа нужно распределить предметы по дням по классам. Приведём такой пример в Таблице 1:

Таблица – Набор предметов для составления расписания

|  |  |  |  |
| --- | --- | --- | --- |
| № п.п |  | 5А | 5Б |
| 1 | Русский | + | + |
| 2 | Математика | + | + |
| 3 | Английский |  | + |
| 4 | Физкультура | + | + |
| 5 | ОБЖ | + | + |
| 6 | Биология | + |  |
| 7 | Литература |  | + |
| 8 | Обществознание | + |  |

Предметы должны распределяться исходя из пожеланий учителей по дням работы. Также нужно учитывать время для выполнения домашнего задания в классе. Можно поставить в один день целую параллель классов, например для учителя география в целях экономии времени учителя для подготовки к урокам на следующий день.

Далее необходимо составить граф. В целях экономии места на Рисунке 1.1будем отмечать предметы порядковым номером из Таблицы 1.

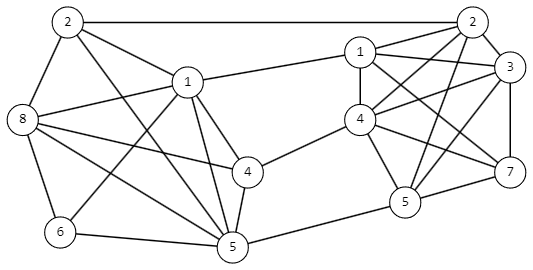


Рисунок .. Граф, составленный по таблице 1

На этом рисунке мы можем заметить 2 многоугольника вершин, расположенные в левой и правой части картинки. Они обозначают 2 разных класса: 5А и 5Б. Цифры в многоугольниках обозначают предметы в классах в понедельник.

Далее мы соединяем вершины рёбрами в местах пересечений. Внутри многоугольников все цифры соединены друг с другом. Это означает, что мы не можем вести две дисциплины в классе одновременно. Это невозможно для детей. Но мы можем, не соединять предметы, на которых класс делится на подгруппы. Например, технология, английский, физкультура.

Далее соединяются вершины между классами. Это означает, что мы не хотим, чтобы учитель преподавал в двух классах одновременно. Можно не соединять вершины, если их ведут разные учителя. Также дополнительно можно соединить русский и литературу, например, в разных классах, если их должен вести один и тот же учитель. Но мы этого делать не будем, потому что примем, что в русский и литературу в школе ведут разные учителя.

Затем происходит раскраска полученного графа. Она представлена на Рисунке 1.2:

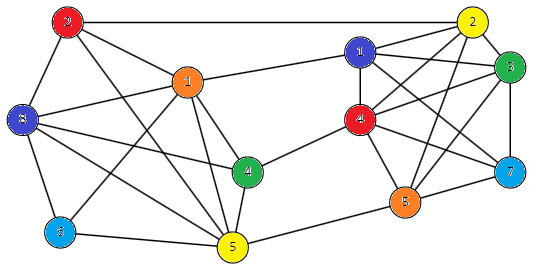


Рисунок 1.2. Раскрашенный граф

Граф можно раскрасить разными способами, причём с помощью всего четырёх цветов. Но на данном примере целесообразнее пользоваться шестью цветами по числу уроков в день. Введём обозначения, красный – первый урок, оранжевый – второй и далее по цветам радуги. Таким образом покажем получившееся расписание в Таблице 2:

Таблица 2 – Получившееся расписание

|  |  |  |
| --- | --- | --- |
|  | 5А | 5Б |
| 1 | Математика | Физкультура |
| 2 | Русский | ОБЖ |
| 3 | ОБЖ | Математика |
| 4 | Физкультура | Английский |
| 5 | Биология | Литература |
| 6 | Обществознание | Русский |

Полученное расписание не имеет наложений уроков у классов и классов у учителей, что соответствует требованиям. Рекомендации о желательном порядковом номере урока также следует учитывать. Для этого выбираются те варианты, где используются соответствующие цвета для урока. Либо же можно выставить цвета для уроков, имеющих константные номера.

При составлении настоящего расписания придётся строить гораздо большие графы, что является очень трудоёмким и долгим процессом.

## Генетические алгоритмы

про документацию пописать везде! а какую?) я уже забыла

# Особенности разработки

Составление расписания является мультиоптимизационной задачей. Требуется учитывать следующие требования:

* отсутствие окон у учеников;
* минимизация числа окон у учителей;
* отсутствие «наслоений» предметов у детей и классов у учителей;
* пожелания учителей;
* соответствие графику трудоспособности детей для дня и недели: учёт времени начала учёбы, окончания, отдыха;
* пожелания учителей;
* не допущение ситуаций, когда предметы из одной группы (например, технические, математика и физика) следуют друг за другом;
* равномерное распределение предметов в течение дня и недели.

Конкретные условия качества составления прописаны в СанПин. В Приложении E приведён пример шкалы трудности предметов для 5-9 классов. В неё по строкам размещены предметы, а по столбцам трудность этой дисциплины в каждом классе.

На рисунке ниже (Рисунок 2.1) приведён пример диаграммы работоспособности в баллах по дням для шестидневной недели.

Рисунок .. Шкала трудности предметов

## Общие архитектурные положения

Проект будет разрабатываться с помощью фреймворка Django. Хотя этот фреймворк и имеет свои минусы (синхронность, встроенные запросы в БД), ими можно пренебречь. В Django используется шаблон программирования MTV, что очень похоже на модель MVC.

Модуль модели часто представлен в виде таблиц БД. Код моделей пишется в файлы models. Django использует ORM в качестве абстракции и переноса из классов моделей в БД.

Модуль templates представлен в виде html-файлов. Часто в них находится специфический код, похожий на язык Jinja. В Django этот язык называется Django template language. Шаблоны можно вкладывать друг в друга.

Модуль View отвечают за логику маршрутизации, заголовки, post, get. Являются связующим звеном между model и templates, отображают ответы и обрабатывают запросы.

## Модель прецендентов

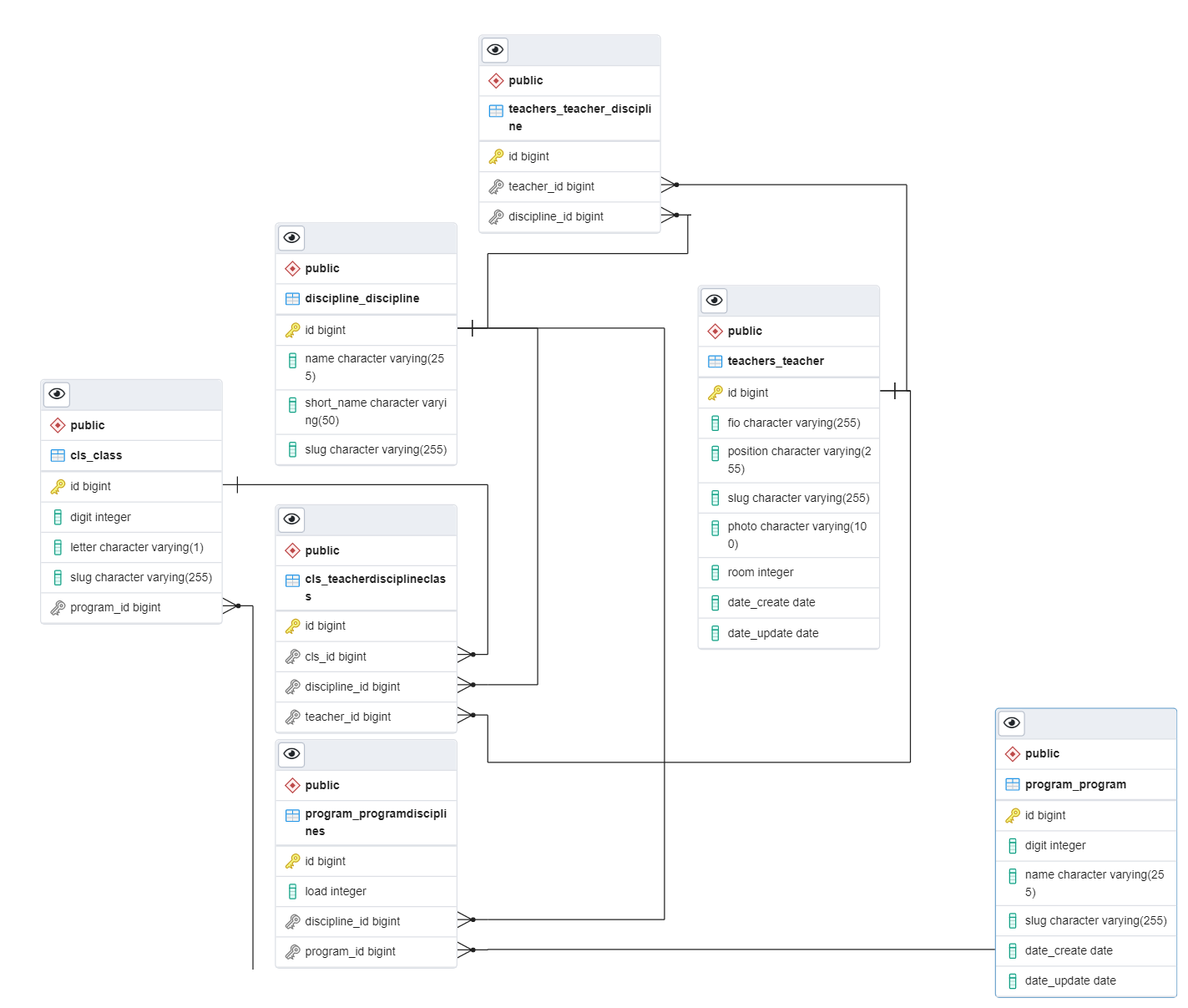
Код для модели прецендентов был написан на языке PlantUML и представлен в Приложении C. Диаграмма приведена в Приложении D.

## Структура проекта

Проект будет состоять из 5 модулей: дисциплины, программы, классы, учителя. Модули перечислены в том порядке, в котором их рекомендуется разрабатывать и в котором удобно работать на сайте.

Схема Rest API представлена в приложении B.

## Диаграмма классов



## Диаграмма компонент

Django

На рисунке ниже представлена стандартная диаграмма компонент django-приложение. Наше приложение целиком соответствует этой диаграмме. При запросе в веб-браузере происходит запрос на сервер в django. Адрес сопоставляется с известными адресами, которое наше приложение умеет обрабатывать. В случае, если приложение не знает такого адреса, то отправит ответ вида 404. Файл **urls** ищет функции в файле **views**. Он в свою очередь оперирует БД (при необходимости), отрисовывает шаблоны из файлов **templates**, передаёт туда данные. **Templates** могут обращаться к статическим файлам типа стилей и скриптов.

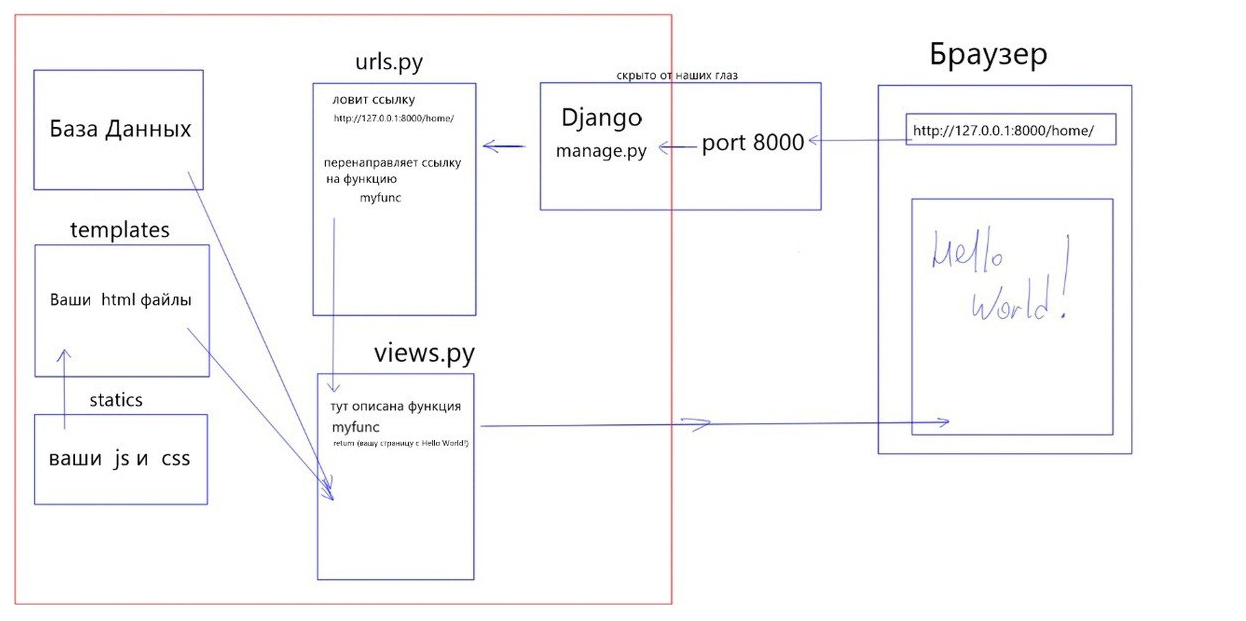


Рисунок .. Диаграмма компонент

## Алгоритмы написания программы

Для генерации расписаний обычно используются алгоритмы полного перебора, которые перебирают все возможные варианты расписаний и выбирают оптимальный вариант. Однако, при большом количестве вариантов расписаний, полный перебор становится вычислительно затратным и неэффективным.

Трудно сказать, какие алгоритмы используются в аналогах моего приложения генерации расписания из-за коммерческой тайны и высокой цены на продукты. Обычно

В комбинаторике, Декартово произведение двух множеств - это множество всех упорядоченных пар элементов, где первый элемент принадлежит первому множеству, а второй элемент - второму множеству. Это понятие широко используется в анализе задач по составлению расписаний, где необходимо комбинировать различные элементы (например, учителей, классы, уроки) для создания оптимального расписания.

Плавно переходя к генетическим алгоритмам, они являются оптимизирующими алгоритмами перебора, которые моделируют процесс естественного отбора и эволюции. Генетические алгоритмы используют понятия генов, хромосом и популяций для поиска оптимального решения. Они позволяют находить более быстрые и приближенные к оптимальному решения в сравнении с полным перебором при работе с большими объемами данных.

Функция качества (fitness function) в генетических алгоритмах определяет, насколько хорошо решение соответствует поставленной задаче. Она оценивает каждое индивидуальное решение в популяции и помогает определить, какие решения следует сохранить и использовать для создания новых поколений. Чем выше значение функции качества, тем более пригодным считается решение. Функция качества играет ключевую роль в оптимизации и выборе лучших расписаний при использовании генетических алгоритмов.

Первичный сбор материалов (литература, интернет-источники) по всем этапам разработки, ключевым моментам, используемым средствам, существующим решениям.

Написать про генетические алгоритмы сюда

архитектура про wsgi и asgi и про миграции

Развёртывания всякое что докер запрещён и что хочется на платный сервак запустить

# Реализация и тестирование

## Руководство пользователя

## Развёртка в контейнере Docker

# Заключение

# Список литературы

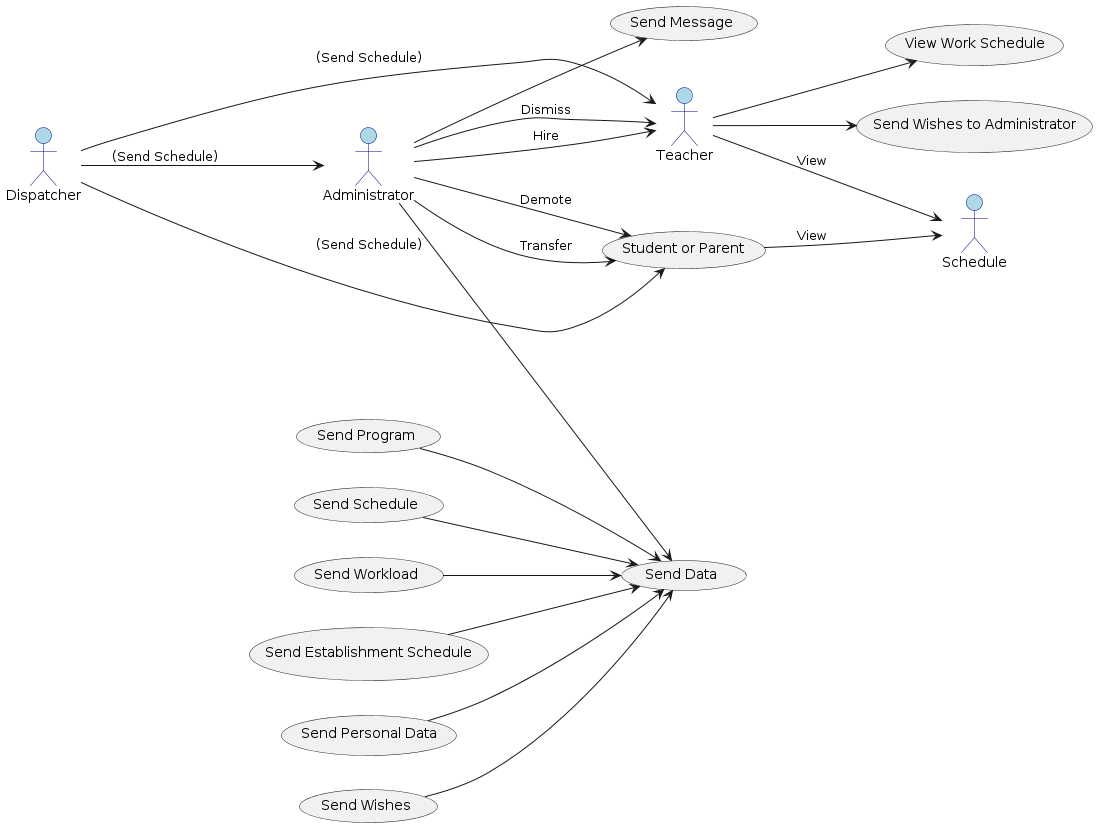
1. Сводная таблица аналогов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Наименование программы | Ссылка | Цена, руб. | Качество по 5-балльной шкале | Особенности |
| Экспресс-расписание Колледж | [https://pbprog.ru/products/programs.php?ELEMENT\_ID=376](https://vk.com/away.php?to=https%3A%2F%2Fpbprog.ru%2Fproducts%2Fprograms.php%3FELEMENT_ID%3D376&cc_key=) | 2 500 - 300 000 руб. | 3 | Есть проверка аудиторий на вместимость. |
| 1С: Автоматизированное составление расписания. Колледж | [https://solutions.1c.ru/catalog/asp\_spo](https://vk.com/away.php?to=https%3A%2F%2Fsolutions.1c.ru%2Fcatalog%2Fasp_spo&cc_key=) | 22 000 – 26 000 | 4 | Решение прорабатывается под отдельную организацию. |
| Система «АВТОРасписание» | [https://www.mmis.ru/programs/avtor](https://vk.com/away.php?to=https%3A%2F%2Fwww.mmis.ru%2Fprograms%2Favtor&cc_key=) | 20 000 - 150 000 | 5 | Развивается с 1996 года.  Наличие подробной статистики и объективной оценки качества любого варианта расписания.  Возможность поддержки любого национального языка (по желанию клиента). |
| Хронограф | [http://softrare.ru/windows/hronograf-30-master](https://vk.com/away.php?to=http%3A%2F%2Fsoftrare.ru%2Fwindows%2Fhronograf-30-master&cc_key=) | бесплатно | 1 | Разрешается указывать данные о кабинетах, инвентаре, расписание звонков. |
| «Ника-Колледж» | [http://nikasoft.ru/nika-college/](https://vk.com/away.php?to=http%3A%2F%2Fnikasoft.ru%2Fnika-college%2F&cc_key=) | 11 000 – 25 000 на 5 лет. Демо – 2500. | 5 | Возможности сдваивания и страивания.  Экспорт в Exel.  Учёт переходов между корпусами.  Учёт пожеланий учителей и совместителей.  Лицензия на 5 лет и помесячно (1, 3, 6).  Возможность докупать отдельные модули. |
| aSc | [https://www.asctimetables.com/](https://vk.com/away.php?to=https%3A%2F%2Fwww.asctimetables.com%2F%23%21%2Fhome&cc_key=) | 150 000 – 200 000 | 3 | Зарубежная программа с удобным интерфейсом имеет хорошие отзывы. |

1. Схема Rest API

|  |  |  |
| --- | --- | --- |
| Метод | URL | Действие |
| Классы | | |
| get | /classes/create\_class/class\_list/ | Список всех классов |
| get | /classes/create\_class/ | Переход на страницу создания класса |
| post | /classes/create\_class/ | Отправка формы, создание класса |
| get | /classes/<slug:slug>/update/ | Переход на страницу обновления класса |
| post | /classes/<slug:slug>/update/ | Отправка формы, обновление класса |
| post | /classes/getTeachersFromDB/ | Вызывает одноимённую функцию, пердаёт список доступных и закреплённых за классом учителей |
| post | /classes/teachers\_field\_form/ | Сохраняет в БД несвязанные с формой поля |
| get | /classes/<slug:slug>/delete/ | Удаляет класс |
| post | /classes/change\_disciplines/ | В зависимости от выбираемой программы изменяет отображения дисциплин и нагрузки |
| Дисциплины | | |
| get | /disciplines/ | Отображает мультиформу создания и изменения дисциплин |
| post | /disciplines/ | Сохраняет мультиформу |
| get | /disciplines/<slug:slug>/delete/ | Удаляет дисциплины |
| Программы | | |
| get | /programs/ | Отображает список всех программ |
| get | /programs/create\_program/ | Форма создания программы |
| post | /programs/create\_program/ | Сохранение данных формы |
| get | /programs/<slug:slug>/update/ | Отображение формы изменения |
| post | /programs/<slug:slug>/update/ | Сохранение формы |
| post | /programs/getHoursFromDB/ | Отправляет данные выбранных дисциплин |
| post | /programs/load\_field\_form/ | Отправка дополнительных полей и создание несвязанных с моделью записей |
| Учителя | | |
| get | /teachers/ | Список всех учителей |
| get | /teachers/<slug:slug>/update/ | Отображение формы обновления учителя |
| post | /teachers/<slug:slug>/update/ | Отправка формы и её сохранение |
| get | /teachers/create/ | Форма создания учителя |
| post | /teachers/create/ | Сохранение учителя |
| get | /teachers/<slug:slug>/delete/ | Удаление учителя |
| post | /teachers/getDataFromDB/ | Отображает дисциплины и классы учителя |
| post | /teachers/classes\_field\_form/ | Отправляет и сохраняет классы учителя |
|  | | |
| get | /admin/\* | Страницы управления администратора |
| get | / | Перенаправление на главную страницу |
| get | /\_\_debug\_\_/ | Вкладка отладки проекта |
| get | /api/schema/ | Статическая схема проекта для формирования документации |
| get | /api/schema/swagger-ui/ | Представление UI интерфейса для просмотра документации |

1. Код создания модели прецендентов
2. @startuml
3. left to right direction
4. skinparam actor {
5. BackgroundColor LightBlue
6. BorderColor DarkBlue
7. }
8. actor Учитель
9. actor Клиент
10. actor Учащийся
11. actor Родитель
12. actor Диспетчер
13. actor Администратор
14. actor Директор
15. actor Заместитель
16. Директор-->Администратор
17. Заместитель-->Администратор
18. Родитель-->Клиент
19. Учащийся-->Клиент
20. Учитель --> (Отправить пожелания)
21. (Отправить пожелания) --> Администратор
22. Учитель --> (Просмотреть график работы)
23. Администратор --> (Отправить сообщение)
24. Администратор --> (Отправить данные)
25. Администратор --> (Перевести)
26. Администратор --> (Понизить)
27. Перевести --> Клиент
28. Понизить --> Клиент
29. Администратор --> (Уволить)
30. Администратор --> (Нанять)
31. Уволить --> Учитель
32. Нанять --> Учитель
33. (Отправить нагрузку) --> (Отправить данные) <<extends>>
34. (Отправить график заведения) --> (Отправить данные) <<extends>>
35. (Отправить личные данные) --> (Отправить данные) <<extends>>
36. (Отправить пожелания) --> (Отправить данные) <<extends>>
37. (Отправить программу) --> (Отправить данные) <<extends>>
38. Диспетчер --> (Отправить расписание)
39. (Отправить расписание)--> Учитель
40. (Отправить расписание)--> Клиент
41. (Отправить расписание)--> Администратор
42. Клиент -->(Просмотреть расписание)
43. Учитель -->(Просмотреть расписание)
44. @enduml
45. Диаграмма прецендентов



1. Шкала трудности учебных предметов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Учебные предметы | Количество баллов (по классам) | | | | |
| 5 класс | 6 класс | 7 класс | 8 класс | 9 класс |
| Физика | - | - | 8 | 9 | 13 |
| Химия | - | - | - | 10 | 12 |
| История | 5 | 8 | 6 | 8 | 10 |
| Иностранный язык | 9 | 11 | 10 | 8 | 9 |
| Математика | 10 | 13 | - | - | - |
| Геометрия | - | - | 12 | 10 | 8 |
| Алгебра | - | - | 10 | 9 | 7 |
| Природоведение | 7 | 8 | - | - | - |
| Биология | 10 | 8 | 7 | 7 | 7 |
| Липература | 4 | 6 | 4 | 4 | 7 |
| Информатика и ИКТ | 4 | 10 | 4 | 7 | 7 |
| Русский язык/Родной язык | 8 | 12 | 11 | 7 | 6 |
| География | - | 7 | 6 | 6 | 5 |
| Изобразительное искусство | 3 | 3 | l | - | - |
| Мировая художественная культура (МХК) | - | - | 8 | 5 | 5 |
| Музыка | 2 | 1 | 1 | 1 | - |
| Обществознание (включая экономику и право) | 6 | 9 | 9 | 5 | 5 |
| Технология | 4 | 3 | 2 | 1 | 4 |
| Основы безопасности жизнедеятельности | 1 | 2 | 3 | 3 | 3 |
| Физическая культура | 3 | 4 | 2 | 2 | 2 |

1. Рейтинговая шкала трудности предметов