Министерство науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский государственный технический университет имени Н.Э. Баумана»

(национальный исследовательский университет)

Московский техникум космического приборостроения

Код, специальность 09.02.07 Информационные системы и программирование

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

по учебной практике по профессиональному модулю

ПМ. 02 Осуществление интеграции программных модулей

Выполнил студент \_\_\_\_\_\_\_Симонян Павел Русланович\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилия, имя, отчество)

Курс \_\_\_\_\_3\_\_\_\_\_\_ Группа \_\_\_ТИП-61\_\_\_

Подпись студента \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель практики \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (\_\_\_Митрошенкова Е.А\_\_\_)

(подпись) (фамилия, имя, отчество)

Москва 2022 г

СОДЕРЖАНИЕ

[Введение 3](#_Toc93389740)

[1 Постановка задачи 4](#_Toc93389743)

[2 Проектирование информационной системы 5](#_Toc93389751)

[3 Описание выбора средств разработки 1](#_Toc93389758)2

[4 Схема данных 1](#_Toc93389761)3

[5 Структура приложения 1](#_Toc93389768)6

[6 Тестирование программы 1](#_Toc93389769)9

[Заключение 2](#_Toc93389770)1

# ВВЕДЕНИЕ

У студента возникает ежедневная нужда изучить своё расписание. Для этого предлагаются различные инструменты. Предлагаемая им альтернатива призвана не только исправить некоторые их недостатки, но и стать новой расширяемой и расширяемой информационной платформой для студентов, преподавателей и администрации техникума.

# 1 ПОСТАНОВКА ЗАДАЧИ

Приложение позволит автоматически структурировать информацию о расписании, предоставляя её для каждой группы отдельно и сохраняя на устройстве для доступа оффлайн. Для заполнения базы данных приложение должно иметь настроенный парсер электронных таблиц, что позволит в процессе эксплуатации легко и быстро обновлять информацию о расписании.

К функциональным требованиям относятся:

* Парсинг электронных таблиц с расписанием;
* Сохранение информации о расписании в базу данных;
* Выдача расписания пользователям;
* Авторизация для администрации и регистрация новых пользователей;
* Возможность редактировать информацию о преподавателях, дисциплинах и графике проведения занятий, чтобы нивелировать неточность парсера;

# 2 ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ

Для разработки востребованной системы необходимо отследить процессы, которые можно оптимизировать и ускорить, сохраняя или дополняя функциональность. С точки зрения администратора - это моментальное донесение информации до студента, используя для этого уже имеющиеся данные, но новые средства.

На рисунке 2.1 представлена диаграмма IDEF0, описывающая бизнес - процесс, выполняемый администратором приложения, который информационная система может оптимизировать.

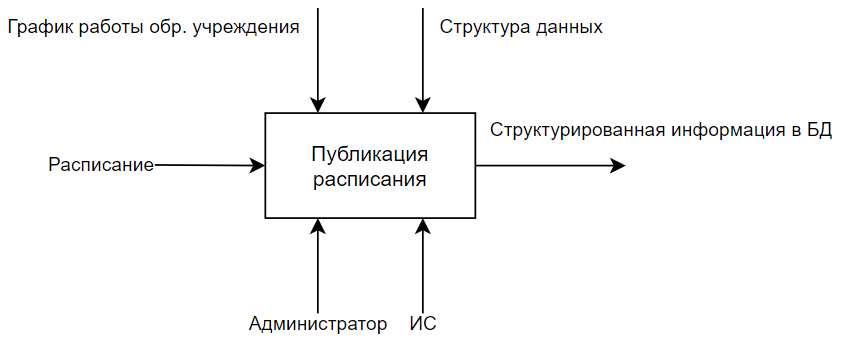


Рисунок 2.1 – Диаграмма IDEF0

На рисунке 2.2 представлена декомпозиция представленной модели.

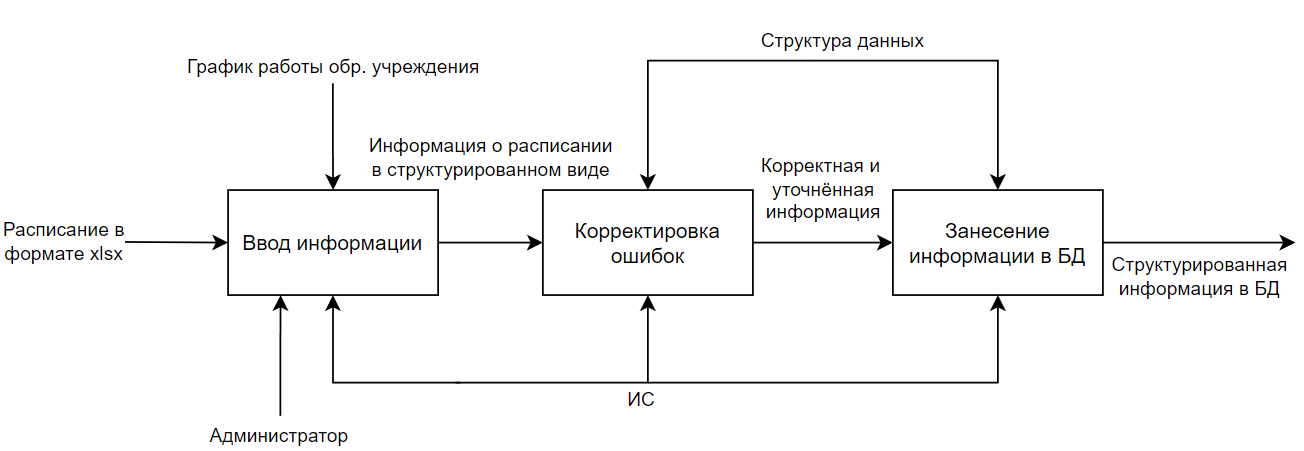


Рисунок 2.2 – Декомпозированная диаграмма IDEF0

Главной проблемой при анализе данных о расписании становится человеческий фактор. Поскольку приложение не предназначается для ввода данных с нуля, но для их парсинга из электронной таблицы, программа не может гарантировать идеальную ссылочную целостность и отсутствие дублирования данных. Впоследствии данная проблема легко может решиться функцией объединения дисциплин под общим именем. Даже с учётом неизбежных ошибок, их корректировка позволяет структурировать данные, полученные из файла, и заполнить ими базу данных для последующих редакций и выборок.

В определении структуры ролей и пользователей поможет построение диаграммы вариантов использования (USE CASE), представленной на рисунке 2.3.

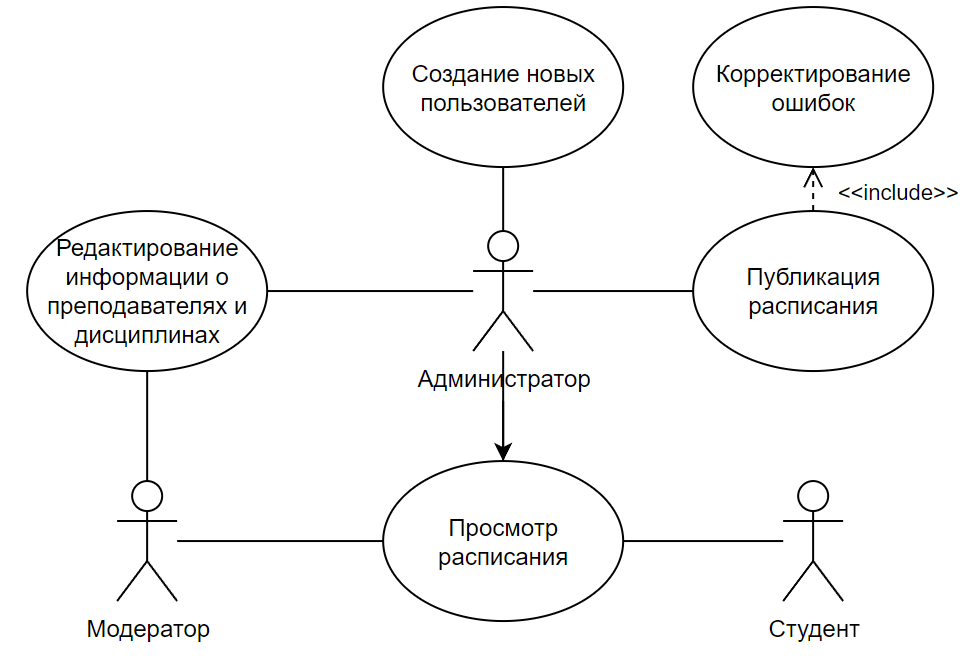


Рисунок 2.3 *–* Диаграмма USE CASE

Хотя при таком размере и функциональности информационной системы ролевая система не проявляет все свои положительные стороны, она позволяет немного разграничить ответственность администраторов приложения и сделать его безопаснее. Изучая диаграмму, можно выделить три действующих лица:

1. Студент или другой незарегистрированный пользователь. Может просматривать расписание для любой группы;
2. Модератор. Дополнительно обладает возможностью редактировать названия предметов и имена преподавателей;
3. Администратор. Обладает всеми привилегиями, может создавать новых пользователей, размещать новое расписание и график проведения занятий;

Мы можем рассмотреть простые варианты использования приложения в контексте действующих лиц, чтобы лучше понять, какие процессы приложения имеют ключевую роль и какие при этом взаимодействуют сущности. Эти данные моделируют диаграммы последовательности на рисунках 2.4 - 2.6.

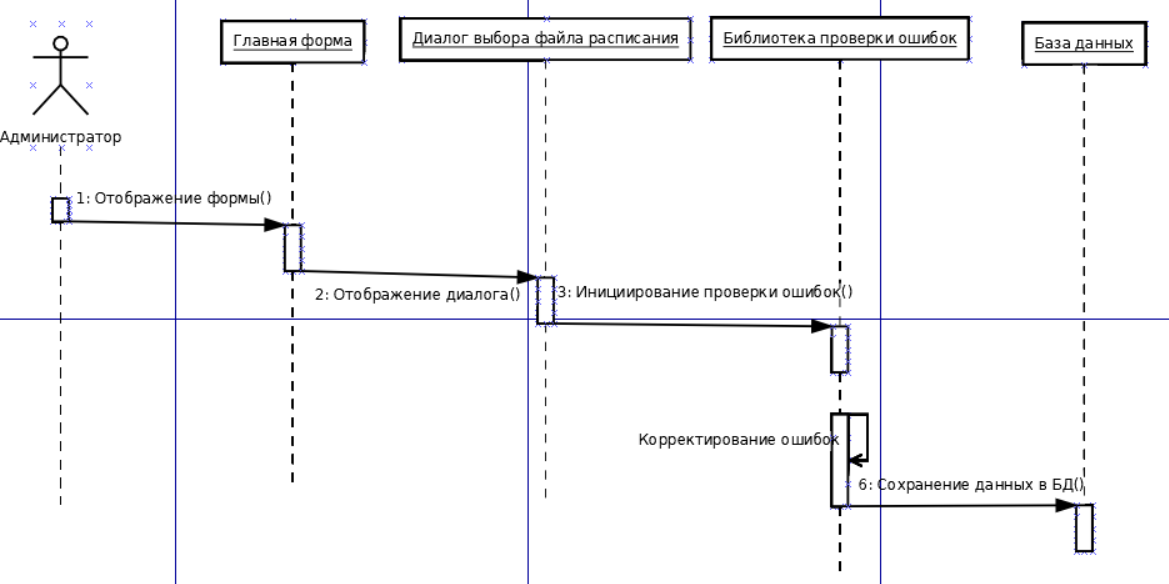


Рисунок 2.4 *–* Диаграмма последовательности для процесса заполнения базы данных

Как мы видим, данный процесс подразумевает минимальное участие со стороны пользователя. Дополнительные корректировки могут быть внесены позже, но большая часть данных уже имеет логическую значимость для студентов и не требуют немедленного редактирования.

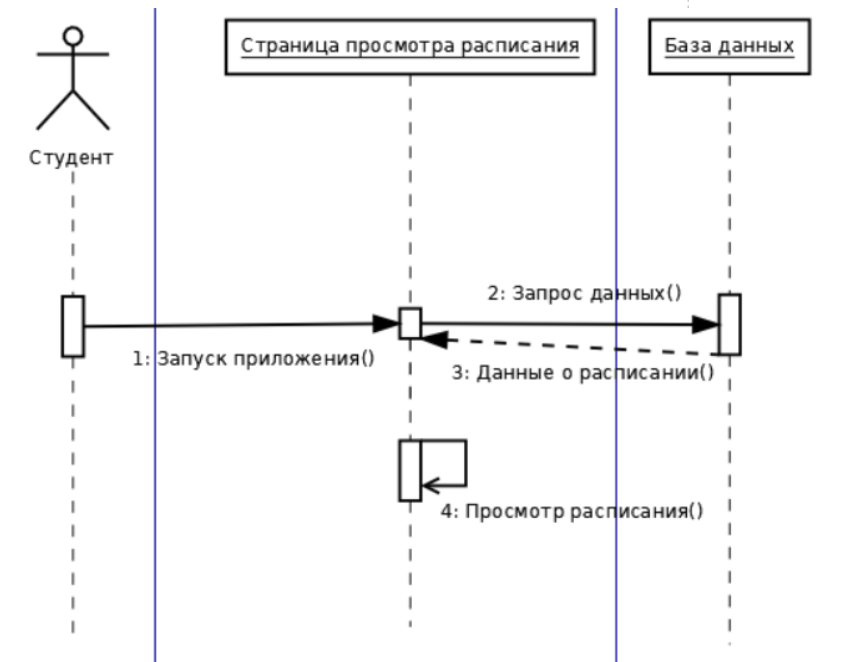


Рисунок 2.5 *–* Диаграмма последовательности просмотра расписания

Данная диаграмма даёт нам не так много, но позволяет обратить внимание на то, как скачивание файла или поиск картинки с расписанием заменяется на автоматическую выдачу нужной информации. И хотя электронный дневник уже предложен учебным заведением, данное приложение всё ещё способно предложить новый опыт использования.

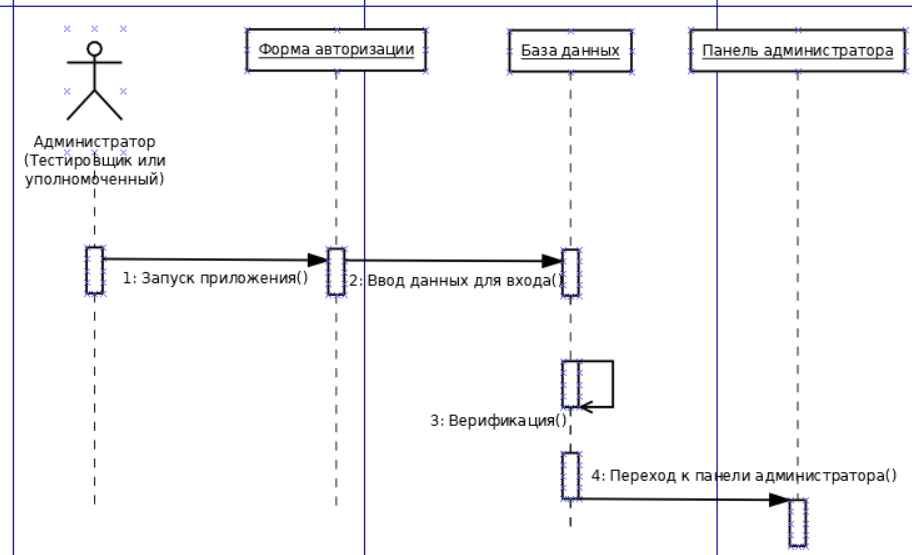


Рисунок 2.6 *–* Диаграмма последовательности авторизации администратора

Данная диаграмма показывает, что для использования дополнительных функций администрирования требуется авторизация со стороны сервера.

На рисунках 2.7 *–* 2.9 представлены диаграммы деятельности для выше представленных процессов. Они позволят пронаблюдать за процессом ветвления в ключевых процессах программы.

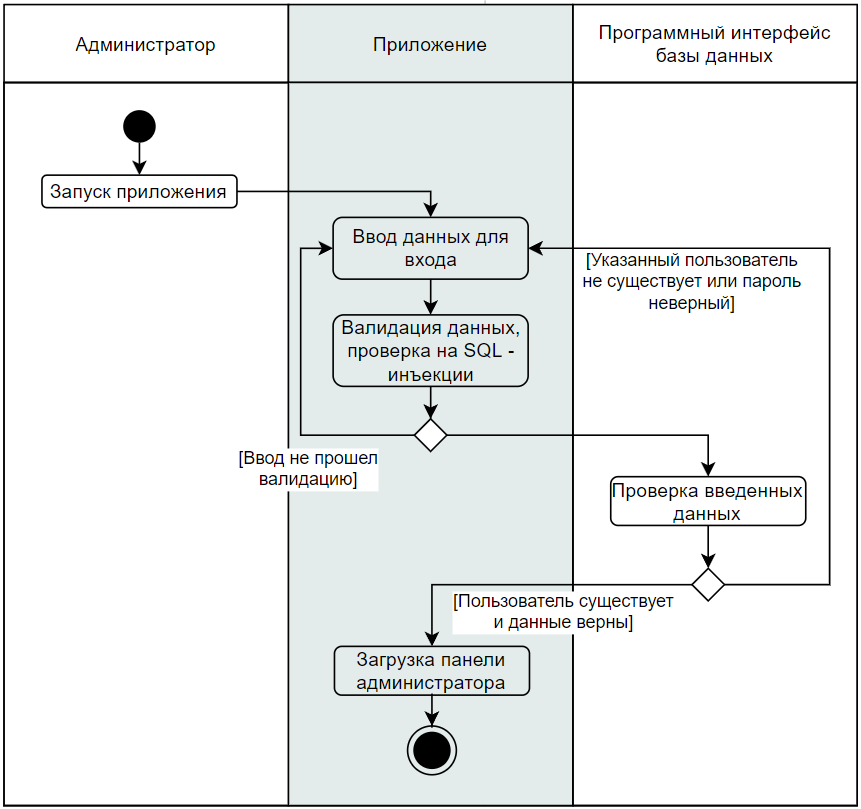


Рисунок 2.7 *–* Диаграмма деятельности процесса регистрации

Диаграмма ниже показывает, что парсер расписаний должен принимать на вход только файлы определённого типа.

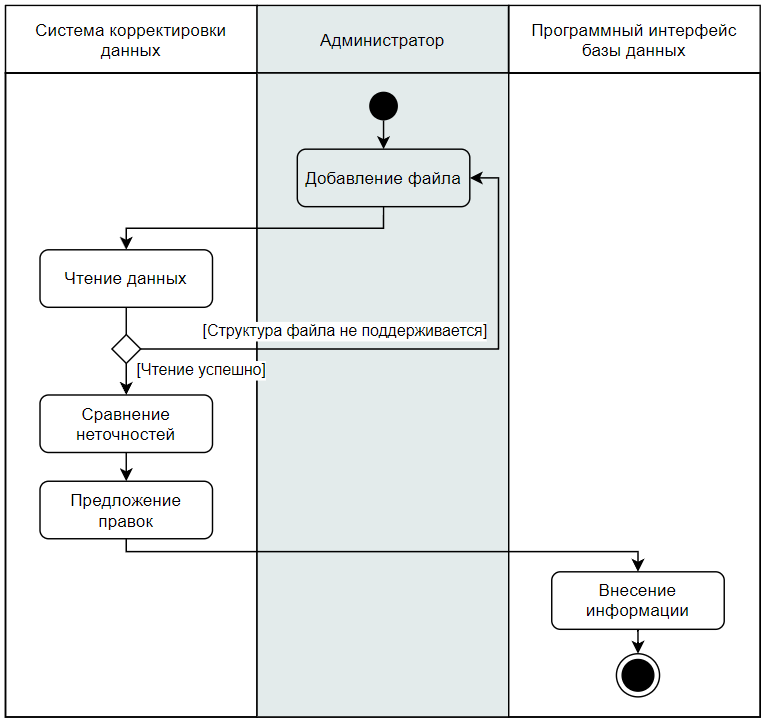


Рисунок 2.8 *–* Диаграмма деятельности для замена расписания

Следующая диаграмма изображает процесс проверки наличия в кэше расписания. Если оно имеется, пользователь сможет просматривать расписание без доступа к Интернету.

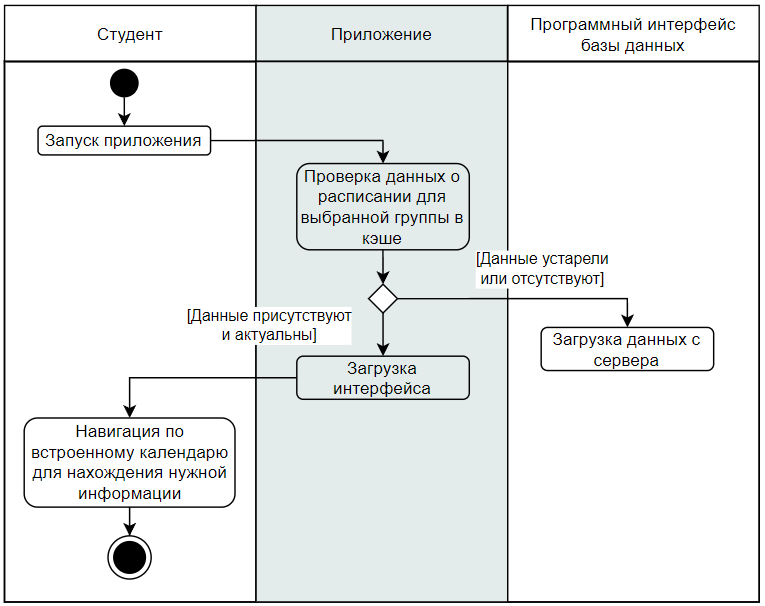


Рисунок 2.9 *–* Диаграмма деятельности процесса проверки кэша

Проанализировав предметную область, мы можем выделить следующие её сущности (изображение 2.10):

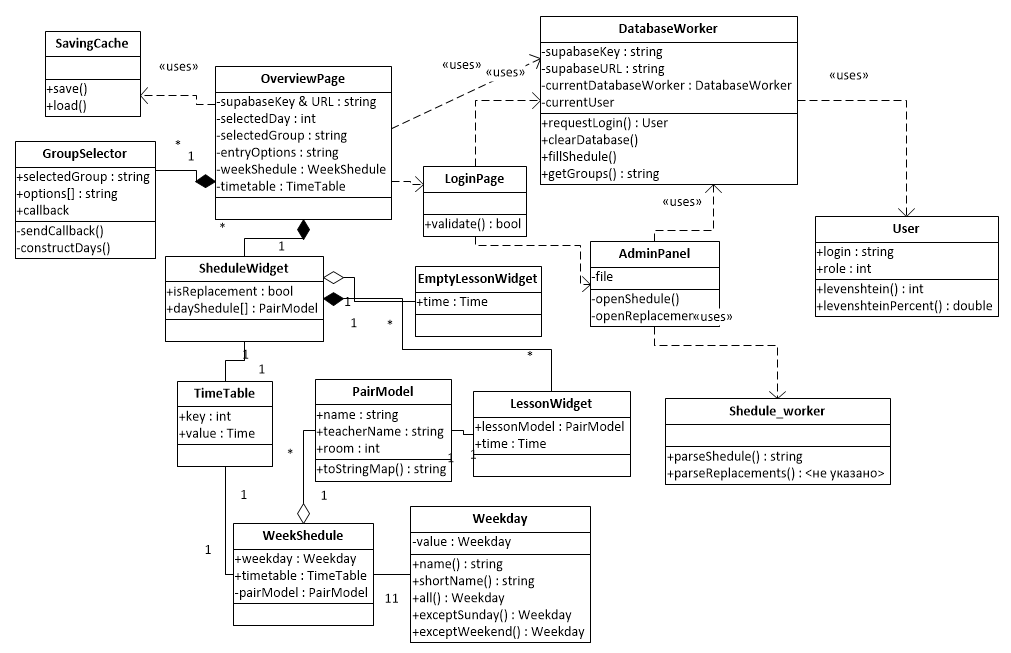


Рисунок 2.10 *–* Диаграмма классов

На диаграмме можно отметить 4 группы элементов.

Страницы: OverviewPage - главная форма с отображением расписания; LoginPage - форма авторизации; AdminPanel - панель администратора для редактирования данных базы;

Вёрстка: SheduleWidget - элемент, отображающий нужный для выбранного дня список элементов LessonWidget; LessonWidget - визуальный элемент, отображающий информацию о паре; EmptyLessonWidget - визуальный элемент, показывающий, что пара пустая (нет занятий в этот час); GroupSelector - визуальный элемент, позволяющий выбрать группу, для которой надо отображать расписания;

«Рабочие»: Классы, берущие на себя конкретную логическую изолированную задачу. SheduleWorker - класс, отвечающий за парсинг расписаний; SavingCache - класс, отвечающий за сохранение и загрузку кэша; DatabaseWorker - класс, отвечающий за работу с базой данных;

Модели: Классы, определяющие модели предметной области и упрощающие взаимодействие с ними в программе. TimeTable - дневной график занятий; PairModel - хранит информацию о паре (кабинет, название, преподаватель, время); Weekday - дни недели; Month - месяцы; User - информация об авторизованном пользователе;

На рисунке 2.11 представлена диаграмма компонентов, показывающая структуру приложения в упрощённом виде:

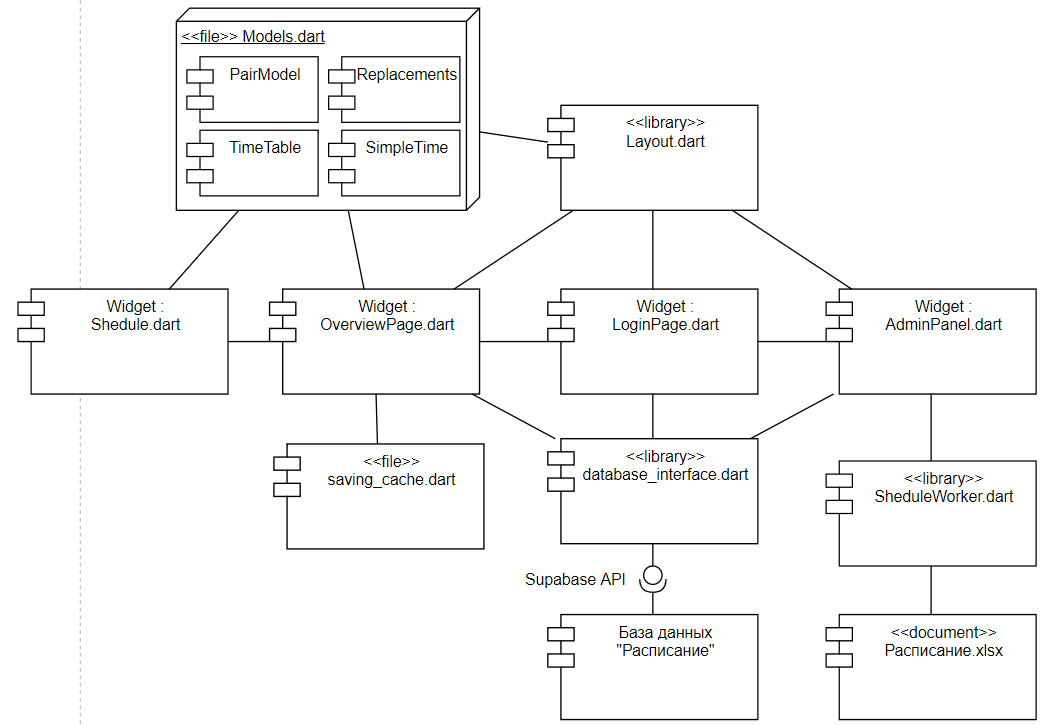


Рисунок 2.11 *–* Диаграмма компонентов

Здесь лучше видно, что некоторые группы классов объединены в один файл исходного кода (Layout.dart, Models.dart). По связям видно, что классы моделей позволяют проще анализировать и итерировать различные взодящие данные. Также видно, что все взаимодействие с базой данных осуществляется только через database\_worker.dart.

Функциональные и системные требования к ИС:

1. Парсинг данных
2. Извлечение данных из БД
3. Хранение данных в памяти устройства
4. Организация защиты данных

# 3 ОПИСАНИЕ ВЫБОРА СРЕДСТВ РАЗРАБОТКИ

Было решено разрабатывать приложение на языке Dart и фреймворке Flutter из-за удобства отладки на компьютере без потери производительности, мультиплатформенности, скорости и относительно большой среди других подобных решений популярности.

Для хранения данных был использован хостинг Supabase и СУБД PostgreSQL, так как эта связка - одна из совсем немногих, позволяющих бесплатное использование с большим лимитом на память, и имеющих API для Dart.

# 4 СХЕМА ДАННЫХ

Структура сущностей базы данных представлена на диаграмме IDEF1X, на рисунке 4.1.

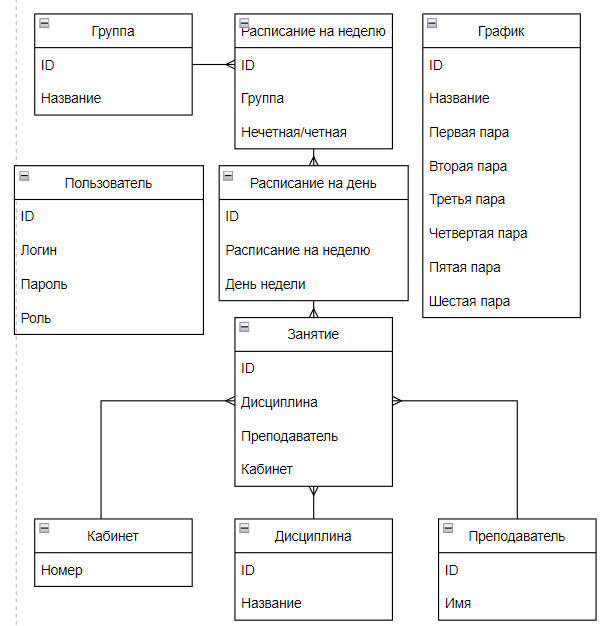


Рисунок 4.1 *–* Диаграмма IDEF1X

Ниже, в таблицах 4.1 *–* 4.9 представлены структуры каждой из сущностей

Таблица 4.1 – Структура таблицы «Пользователь»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | Тип данных | Размер | Описание |
| ID | Числовой | 11 | ID |
| Логин | Текстовый | 10 | Имя пользователя |
| Пароль | Текстовый | 128 | SHA512 Хэш-код пароля |
| Роль | Числовой | 3 | Роль пользователя (0 - владелец, 1 - администратор, 2 - модератор) |

Таблица 4.2 – Структура таблицы «Группа»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | Тип данных | Размер | Описание |
| ID | Числовой | 11 | ID |
| Название | Текстовый | 10 | Название группы |

Таблица 4.3 – Структура таблицы «Дисциплина»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | Тип данных | Размер | Описание |
| ID | Числовой | 11 | ID |
| Название | Текстовый | 100 | Название дисциплины |

Таблица 4.4 – Структура таблицы «Преподаватель»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | Тип данных | Размер | Описание |
| ID | Числовой | 11 | ID |
| Имя | Текстовый | 100 | Имя преподавателя |

Таблица 4.5 – Структура таблицы «Кабинет»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | Тип данных | Размер | Описание |
| Номер | Текстовый | 15 | Номер (иногда название) кабинета |

Таблица 4.6 – Структура таблицы «Расписание на неделю»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | Тип данных | Размер | Описание |
| ID | Числовой | 11 | ID сотрудника, первичный ключ |
| Группа | Текстовый | 50 | ФИО сотрудника |
| Нижняя | Логический | 1 | Является нижней? В противном случае - верхняя. |

Таблица 4.7 – Структура таблицы «Расписание на день»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | Тип данных | Размер | Описание |
| ID | Числовой | 11 | ID |
| Расписание на неделю | Числовой | 11 | Расписание на неделю, внешний ключ |
| День недели | Числовой | 1 | День недели, 1-7 (1-6 без воскресенья) |

Таблица 4.8 – Структура таблицы «График»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | Тип данных | Размер | Описание |
| ID | Числовой | 11 | ID |
| Название | Текстовый | 10 | Название графика для простоты ориентирования |
| Первая пара | Текстовый | 11 | Время первой пары (‘09:00-10:30’) |
| ... | … | … | … |
| Шестая пара | Текстовый | 11 | Время шестой пары |

Таблица 4.9 – Структура таблицы «Занятие»

|  |  |  |  |
| --- | --- | --- | --- |
| Идентификатор | Тип данных | Размер | Описание |
| ID | Числовой | 11 | ID |
| Дисциплина | Числовой | 11 | Дисциплина |
| Преподаватель | Числовой | 11 | Преподаватель, у которого эта пара |
| Кабинет | Текстовый | 15 | Кабинет, в котором проводится занятие |

Физическая диаграмма базы данных представлена на следующем изображении.

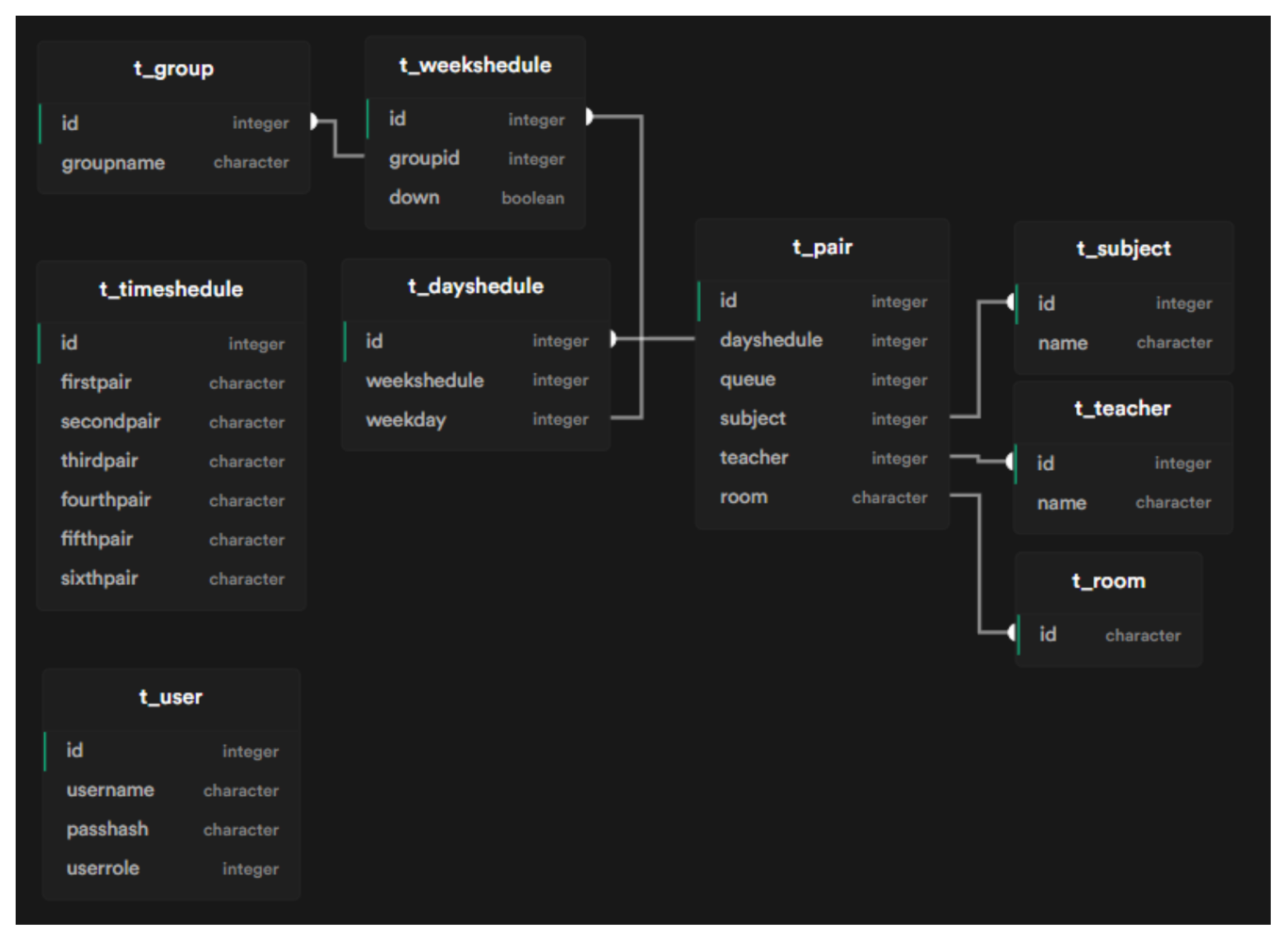


Рисунок 4.2 *–* Физическая модель базы данных

# 5 СТРУКТУРА ПРИЛОЖЕНИЯ

Наиболее чётко, на мой взгляд, структура приложения изображена на рисунке 2.11 (диаграмма компонентов). В Flutter все визуальные элементы называются «виджетами». Самые распространённые из них помещены в библиотеку layout.dart и унифицированы для повторного использования.

Навигация по приложению линейная с возможностью сразу вернуться к расписанию (при этом пользователь не выйдет из аккаунта).

Ниже представлены схемы одних из самых важных алгоритмов в программе.

На рисунке 5.2 представлена схема работы формы авторизации.



Рисунок 5.1 *–* Схема алгоритма формы авторизации

Как можно увидеть, пароль хэшируется, а имя пользователя защищается от SQL - инъекций.

На рисунке 5.2 представлена схема алгоритма работы парсера.

. 

Рисунок 5.2 *–* Схема работы парсера

Как можно увидеть из диаграммы, программа проверяет значения, которые уже были распознаны. Если найдено значение, которое по каким-то критериям считается лучшим, мы избегаем дублирования информации и ссылаемся на старую запись, в противном случае либо перезаписываем ее, либо создаём новую, если значения совершенно разные.

6 ТЕСТИРОВАНИЕ ПРОГРАММЫ

Для начала можно попробовать зайти в приложение без подключения к интернету. Тогда оно уведомит нас об этом:

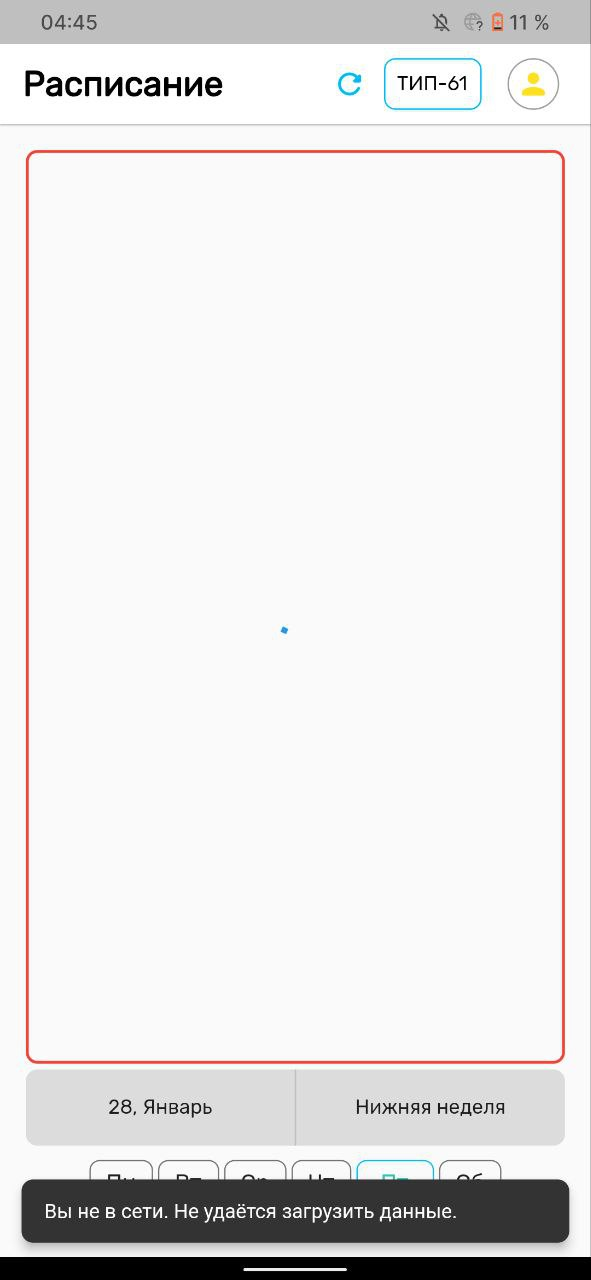


Рисунок 6.1 *–* Уведомление об отсутствии Интернета

Далее попробуем авторизоваться и увидим несколько проверок для формы (изображения 6.2 - 6.4):

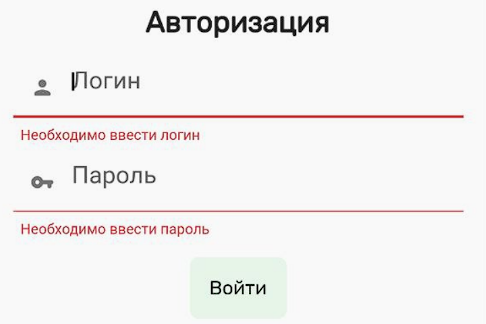


Рисунок 6.2 *–* Проверка на пустой ввод

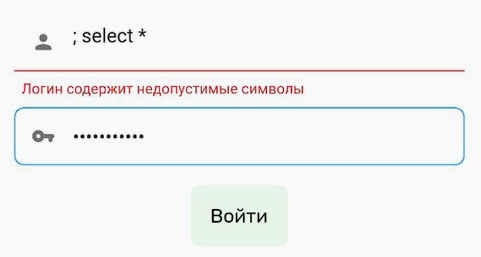


Рисунок 6.3 *–* Проверка на недопустимые символы

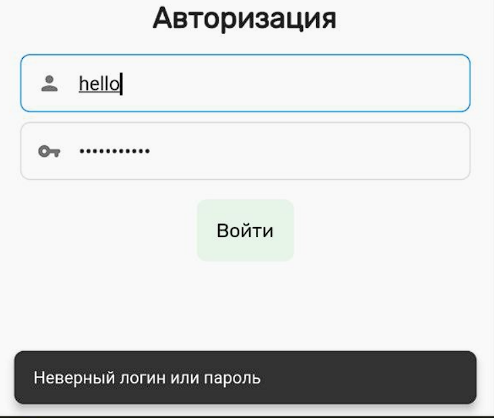


Рисунок 6.4 *–* Проверка на неверные данные

Если приложение не находит для пары преподавателя или кабинет, значения в расписании меняются на прочерки или «Не указан»

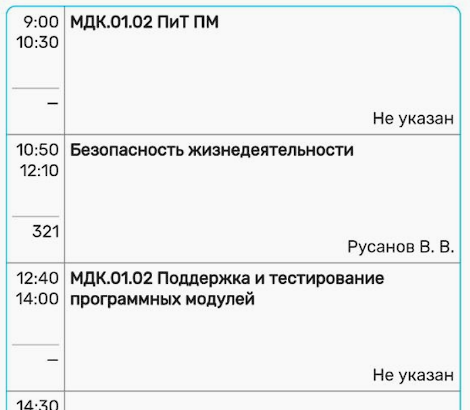


Рисунок 6.5 *–* Замена отсутствующих данных

Далее, в таблицах 6.1 и 6.2 представлены позитивные и негативные кейс тесты соответственно. Рассматривается процесс аутентификации пользователя.

Таблица 6.1 – Позитивный кейс тест аутентификации

|  |  |  |
| --- | --- | --- |
| Действие | Ожидаемый результат | Результат теста |
| Загрузка расписания из файла с расписанием | * Парсинг занимает не более 15 секунд на телефоне * Данные успешно добавлены в базу данных * Пользователь оповещён о удачной попытке | * Парсинг занимает около 10 секунд * Данные успешно добавлены * Пользователю показано всплывающее сообщение об успехе |

Таблица 6.2 – Негативный кейс тест аутентификации

|  |  |  |
| --- | --- | --- |
| Действие | Ожидаемый результат | Результат теста |
| Обновление имени преподавателя - ввод SQL - инъекции | * Форма сообщит о некорректном вводе | * Успешно |
| Обновление расписания сторонней xlsx таблицей | * Приложение не упадёт * Файл не распарсится, база данных не нарушится * Пользователя уведомят о неудачной попытке обновления расписания | * Успешно * Успешно * Успешно |

# ЗАКЛЮЧЕНИЕ

В рамках данной учебной практики было разработано мультиплатформенное приложение для работы с удалённой базой данных, а также была подготовлена вся документация для этого приложения, которую мы на данный момент изучили. Я закрепил навык постановки задачи и планирования разработки программ.