МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

**«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Факультет информационных систем и технологий

Кафедра «Информатика и вычислительная техника»

**Отчет**

по индивидуальному заданию в рамках дисциплины

«Управление потоками работ в САПР»

Выполнил:

П.С.Кондратьев

Приняли:

Ю. А. Лапшов

В.Н.Негода

Ульяновск – 2020

1. **Цели и содержание индивидуального задания**

Целями индивидуального задания являются:

* изучение основ учебной дисциплины в контексте проекта создания средств автоматизации формирования индивидуальных планов преподавателей;
* развитие средств автоматизации организационного управления кафедры ВТ и УлГТУ.

Содержание индивидуального задания приведено в следующей таблице:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Учебная дисциплина | Отчетность | Вид и объем учебной работы | Индивидуальное задание | Кто принимает результат |
| Управление потоками работ в САПР | Экз | Лек.: 8 Лаб.: 16 | Анализ методов управление потоками работ при проектировании автоматизированных систем. Организация управления потоками работ при проектировании системы формирования индивидуальных планов преподавателей. | В. Н. Негода  Ю. А. Лапшов |

1. **Анализ методов и средств автоматизации проектирования автоматизированных систем организационного управления**

Суть проектирования заключается в реализации 2 систем: серверной (Back-end на Фреймворке NestJs) и пользовательского интерфейса (Front-end на Фреймворке Angular) для автоматизированной системы поддержки рабочих программ.

Начать необходимо с понятия “Автоматизированная система”.

Автоматизированная система (АС) — человеко-машинная система для выполнения ежедневных, часто рутинных, профессионально выполняемых на рабочем месте сотрудника работ с целью сокращения времени, ошибок и обеспечения оперативной связи с другими работниками.

Задачей же автоматизированной системы поддержки рабочих программ является автоматизированное рабочее место, целью которого является снижения нагрузки сотрудника, отвечающего за обработку информации и снижению затрачиваемого времени на создание рабочей программы, что позволит преподавателю меньше заниматься рутинной работой с документацией и уделять больше внимания студентам и процессу обучения.

* 1. **Анализ проблематики проектирования АС ОУ**

Анализ предметной области проводится с целью получения необходимой информации о процессах и объектах, относящихся к сфере действия информационной системы. Сферу действия информационной системы можно представить в виде области, границы которой отделяют процессы и объекты, относящиеся к информационной системе, от тех процессов и объектов предметной области, которые в данной информационной системе не будут представлены.

Разработка архитектуры системы же основывается на построении архитектуры системы, которая является определение автоматизируемых функций. Для этого для каждой функции ОМБП (Обобщенная модель бизнес-процессов) необходимо определить целесообразность ее автоматизации. Для каждой автоматизируемой функции необходимо определить степень автоматизации – будет ли функция автоматизирована полностью или функция будет разделена на автоматизируемую часть и часть, которая будет выполняться без средств автоматизации. В последнем случае необходимо преобразовать ОМБП для выделения функций, которые должны быть полностью автоматизированы. Следующим шагом является формирование программных комплексов на основе выделенных операций. Эта процедура производится на основе организационной структуры, в рамках которой функционирует ИС.

Далее идет разделение программных модулей по типам (универсальные программные модули, программные модули слоя правил бизнеса, программные модули слоя документов, программные модули управления бизнес-процессом и интерфейсные программные модули) производится на этапе разработки программной системы.

Завершает проектирование отдельной подсистемы, которая в свою очередь представляет собой процесс определения детального состава программных модулей подсистемы и уточнение таблиц баз данных, с которыми будут работать эти программные модули. Основной задачей проектирования отдельной подсистемы ИС является детальная проработка модели бизнес-процессов и информационной модели для отдельной подсистемы ИС.

Обеспечение целостности больших информационных систем требует поддержания моделей системы в актуальном состоянии на всех этапах разработки. Предлагается разделить проектирование информационных систем на два этапа: этап общего проектирования и этап проектирования отдельных подсистем.

На этапе общего проектирования формируется архитектура системы как логическая модель информационной системы, объединяющая модель бизнес-процессов, информационную модель и модель организационной структуры.

На этапе детального проектирования отдельных подсистем происходит уточнение архитектуры системы, что обеспечивает поддержание моделей системы в актуальном состоянии.

* 1. **Спецификация потока работ, связанных с выполнением ВКР**

Разрабатываемая подсистема интеграции с внешними ресурсами является частью платформы, которая является частью автоматизированной системы поддержки рабочих программ.

Назначением данной подсистемы является работа с внешними данными, которые необходимы для создания «Рабочей программы». Внешние данные включают в себя рабочие планы дисциплин, расчеты штатов, нормы времени.

Являясь частью комплексной платформы, данная подсистема имеет общую цель системы – создание автоматизированной системы поддержки рабочих программ с возможностью использования готового шаблона и исходных данных по необходимому предмету. Главная задача данной системы - снижение нагрузки сотрудника, отвечающего за обработку информации, а также минимизация возможности совершения ошибки при оформлении рабочей программы, так как имеется стандартизованный шаблон.

Таблица 1. Специфицирующая потоки работ

|  |  |  |  |
| --- | --- | --- | --- |
| № | название работы | ожидаемые результаты | планируемая трудоемкость |
| 1 | Разработка диаграмм компонентов | Диаграмма деятельности, Use-case диаграммы, ER диаграммы | 3 дня |
| 2 | Анализ представление учебных планов образовательных программ и спецификация разработка механизмов извлечения данных из них | UML диаграмма, диаграмма связей, Получение данных в виде json объектов | 3 недели |
| 3 | Анализ представления расчета штатов и разработка механизмов извлечения данных из них | UML диаграмма, Система отношений | 1 неделя |
| 4 | Анализ представления индивидуальных планов в части учебно-методической, организационно-методической и научно-исследовательской работы и разработка механизмов автоматизации формирования этой части | Парсинг данных,  Сохранение в базе данных,  Создание шаблона документа,  Заполнение шаблона,  Запись данных в шаблон,  Создание общей схемы получения данных | 4 недели |

1. **Прототипирование проектных решений сформированного потока работ**
   1. **Анализ основных видов проектных решений и инструментальных средств поддержки их формирования.**

Модель автоматизированного рабочего места можно разделить на 2 части: обеспечивающую и функциональную.

Отличительные особенности функций деятельности пользователя, которые необходимо автоматизировать, сущность определенного АРМ (Автоматизированное рабочее место) и представление совокупности взаимной связи задач определяет функциональная часть. Для разработки функционального обеспечения необходимо знание входных и выходных данных, пользовательских требований к АРМ, а также средств и методов достижения достоверности и качества информации. Это может включать в себя способы защиты от несанкционированного доступа, систему авторизации и доступа для пользователя, логгирование изменений, а также возможность сохранения состояния системы в чрезвычайных ситуациях (сбой, нестандартные ситуации).

Обеспечивающая часть включает традиционные виды обеспечения:

* информационное (описание организации, базы данных, структур связей подсистем),
* программное (подразделяется на общее – операционные системы, вспомогательные программы т.п., и функциональное – универсальные программы и пакеты),
* техническое (совокупность технических средств обработки информации),
* технологическое (программное обеспечение, предназначенное для организации технологического процесса),
* лингвистическое (программное обеспечение для организации совместных условий работы специалистов).

Основным модулем программного продукта является автоматизированное рабочее место преподавателя, которое выполняет задачу по созданию «рабочих программ» по закрепленному за ним предметом. Разработка базовых проектных решений системы поддержки формирования индивидуальных планов преподавателей

* 1. **Разработка диаграмм компонентов**

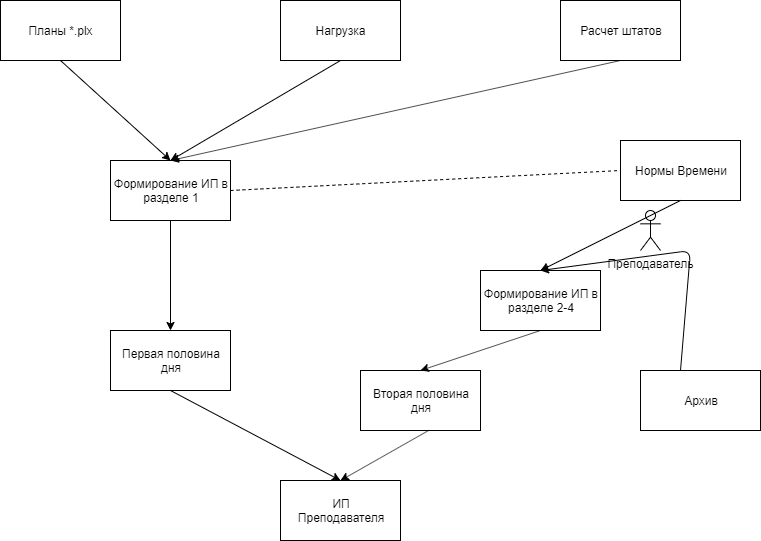


Рис. 1. Концепция автоматизированной системы поддержки рабочих программ (версия 1)

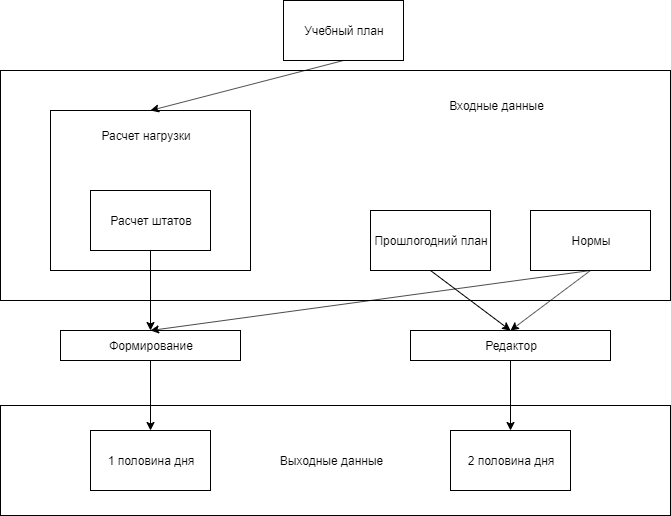


Рис. 2. Концепция автоматизированной системы поддержки рабочих программ (версия 2)

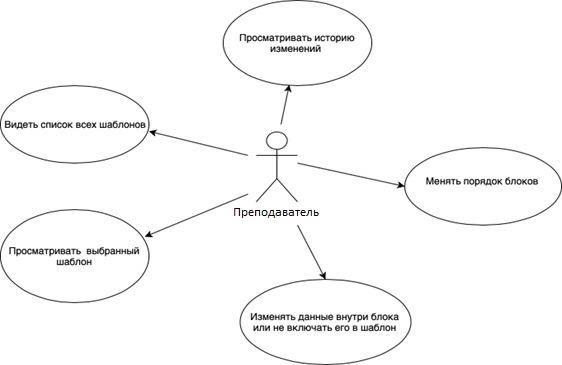


Рис. 3. Пример Use-case диаграммы в системы поддержки рабочих программ

* 1. **Анализ представление учебных планов образовательных программ и спецификация разработка механизмов извлечения данных из них.**

Учебный план – это документ, устанавливающий список предметов, объем рабочего времени, порядок следования и отдельные этапы подготовки учебных дисциплин, стажировок, факультативных занятий, учебных модулей и других видов практической образовательной деятельности. Также в нем указываются положения о промежуточной оценке студентов и учеников, если этого не определяет Федеральный закон об образовании.

Учебный план представлен в формате .plx и содержит в себе XML разметку, содержащую в себе необходимую информацию для создания шаблона рабочей программы в автоматическом режиме.

Благодаря тому, что XML является универсальным форматом для обмена информацией, взаимодействие с данным языком разметки имеет простую структуру. Для работы с ним используются так называемые XML-анализаторы, которые работают с xml-разметкой и позволяют находить по тегам необходимую информацию, которую нужно выделить из файла.

Чтобы работа с xml файлом была удобней, принято то что его нужно распарить в формате json, для удобства чтения и простоты использования данных внутри проекта. Парсинг происходит с помощью библиотеки xml2js, взятой из менеджера пакетов JS.

Листинг 1. Загрузка и парсинг xml-файла.

let { parseString, Builder} = require('xml2js');  
let xml = fs.readFileSync("09.03.01\_02-2019.plx", "utf8").toString();

parseString(xml, { explicitArray : false }, (error, result) => {

if(error) throw error;

fs.writeFileSync("json", JSON.stringify(result, null, 4));

console.log(result.Документ);

});



Рис. 4 Файл json

* 1. **Анализ представления расчета штатов и разработка механизмов извлечения данных из них.**

Расчет штатов (нагрузка преподавателя) представляет собой файл расширения .xlsm, но также часы по какому-либо предмету можно найти в в файле учебного плана с расширением .plx (был рассмотрен ранее в П.3.2). Так как заведомо в автоматизированной системе входные данные неявно заданы (могут храниться в xlm или excel), то разработка будет происходить не сверху вниз, а снизу-вверх (отталкиваемся от выходного файла, который должен получиться после использования расчетов штатов – 1 половина дня).



Рис. 5. Пример объекта дисциплины Высокопроизводительные вычисления

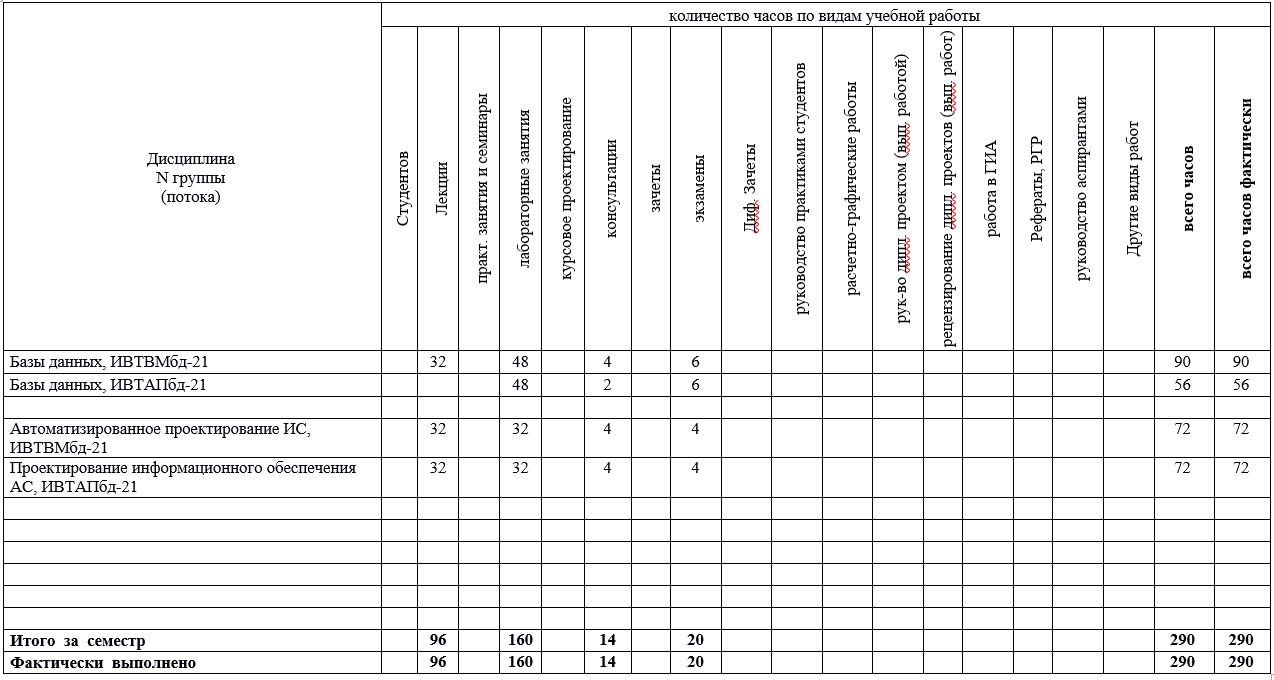


Рис. 6. Нагрузка преподавателя по программам высшего образования

* 1. **Анализ представления индивидуальных планов в части учебно-методической, организационно-методической и научно-исследовательской работы и разработка механизмов автоматизации формирования этой части.**

Прототип индивидуальных планов в части учебно-методической, организационно-методической и научно-исследовательской работы и разработка механизмов автоматизации формирования будет представлять собой:

* Парсинг данных
* Сохранение в базе данных
* Создание шаблона документа
* Заполнение шаблона
* Запись данных в шаблон

Пример прототипа, создание документа:

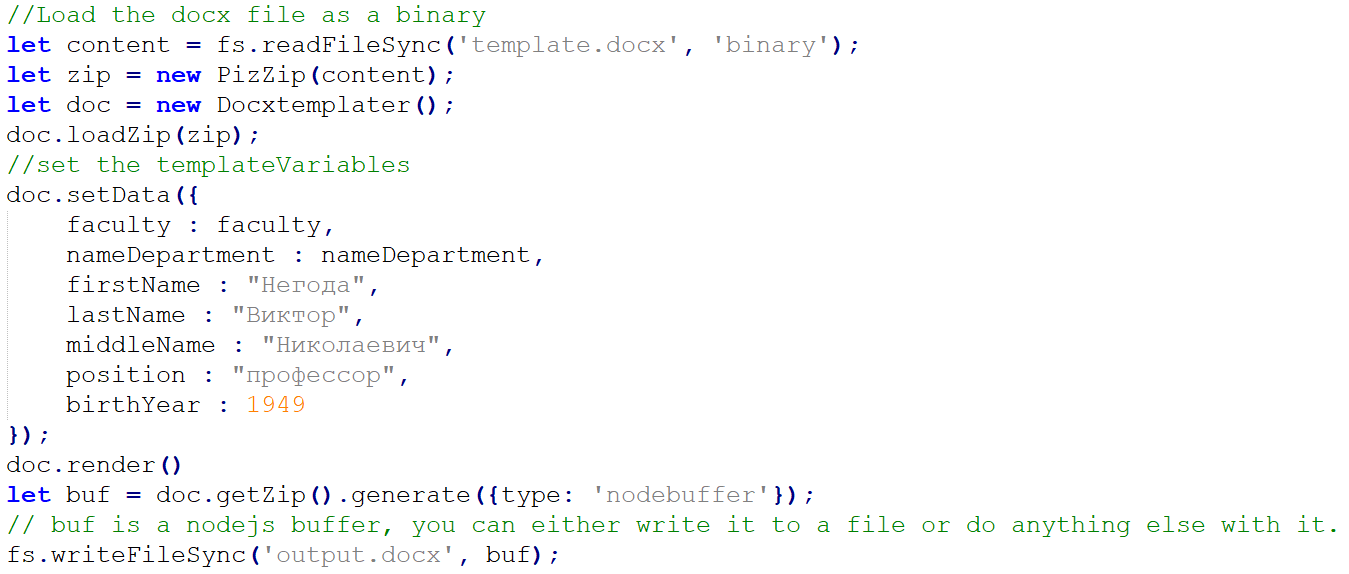


Рис. 7. Создание и сохранение данных в документ по шаблону

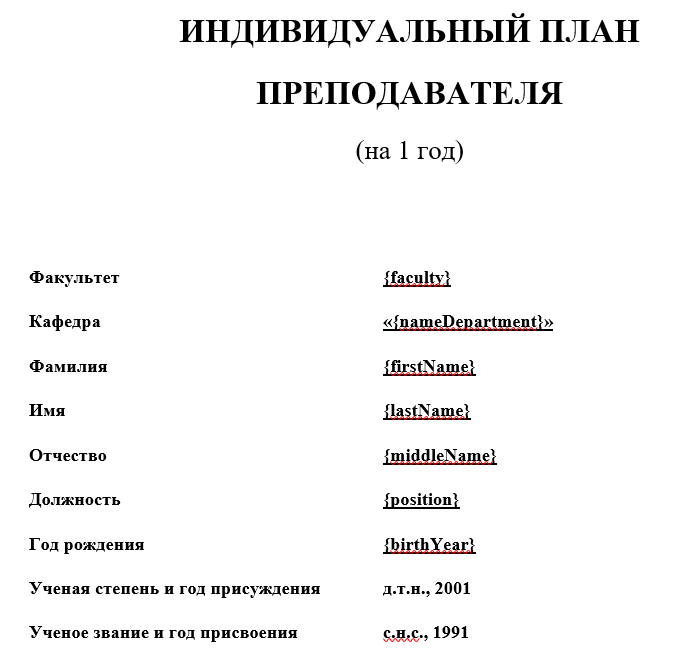


Рис. 8. Шаблон документа template.docx

1. **Анализ методов управление потоками работ с использованием API**

В проект был включен GraphQL - язык запросов и манипуляций данными с открытым исходным кодом для API и среда выполнения для выполнения запросов с существующими данными.

В двух словах, GraphQL это синтаксис, который описывает как запрашивать данные, и, в основном, используется клиентом для загрузки данных с сервера. GraphQL имеет три основные характеристики:

* Позволяет клиенту точно указать, какие данные ему нужны.
* Облегчает агрегацию данных из нескольких источников.
* Использует систему типов для описания данных.

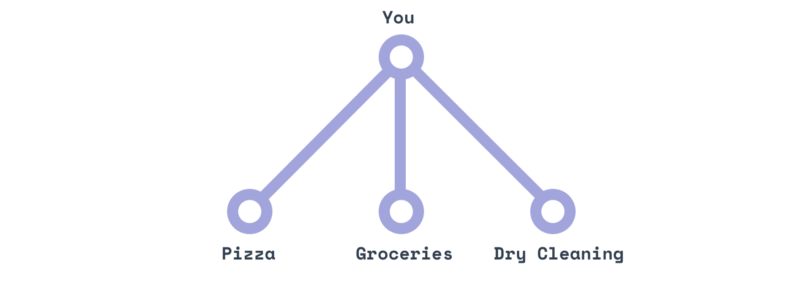
GraphQL позволяет добиться:

* Высокой степени готовности инструментальных средств использования в проекте, возможность избежать дублирования кода;
* Небольшой семантический разрыв между языками спецификациями и манипуляции, с одной стороны, и логической организации данных с другой стороны;

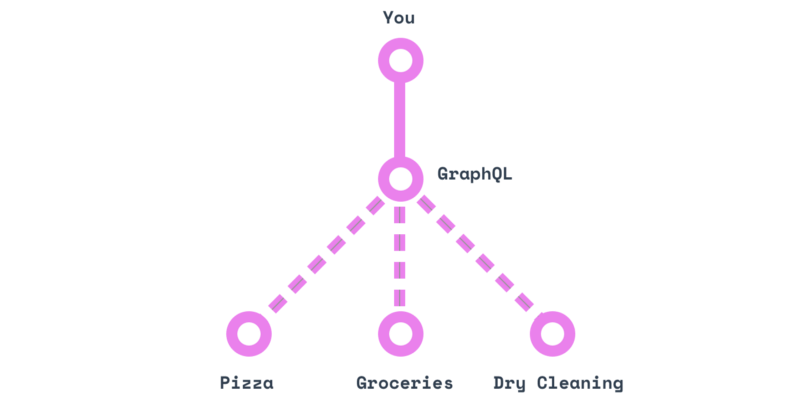
Фактически, слой GraphQL находится между клиентом и одним или несколькими источниками данных; он принимает запросы клиентов и возвращает необходимые данные в соответствии с переданными инструкциями.

**Простой пример:**

Пользоваться старой REST-моделью это как заказывать пиццу, затем заказывать доставку продуктов, а затем звонить в химчистку, чтобы забрать одежду. Три магазина – три телефонных звонка.



GraphQL похож на личного помощника: вы можете передать ему адреса всех трех мест, а затем просто запрашивать то, что вам нужно («принеси мне мою одежду, большую пиццу и два десятка яиц») и ждать их получения.



На практике GraphQL API построен на трёх основных строительных блоках: на схеме (schema), запросах (queries) и распознавателях (resolvers).

**Запросы (queries)**

(query и request одинаково переводится как "запрос". Далее будет подразумеваться query, если не указано иначе – прим. пер.)

Когда вы о чём-то просите вашего персонального помощника, вы выполняете запрос. Это выглядит примерно так:

query {

stuff

}

Мы объявляем новый запрос при помощи ключевого слова query, также спрашивая про поле stuff. Самое замечательное в запросах GraphQL является то, что они поддерживают вложенные поля, так что мы можем пойти на один уровень глубже:

query {

users {

firstName, lastName, middleName, age

}

}

Как можно заметить, клиенту при формировании запроса не нужно знать откуда поступают данные. Он просто спрашивает о них, а сервер GraphQL заботится об остальном.

К примеру получение тех же самых полей через запрос SELECT:

SELECT firstName, lastName, middleName, age FROM users;



Рис. 9. SCHEMA QL

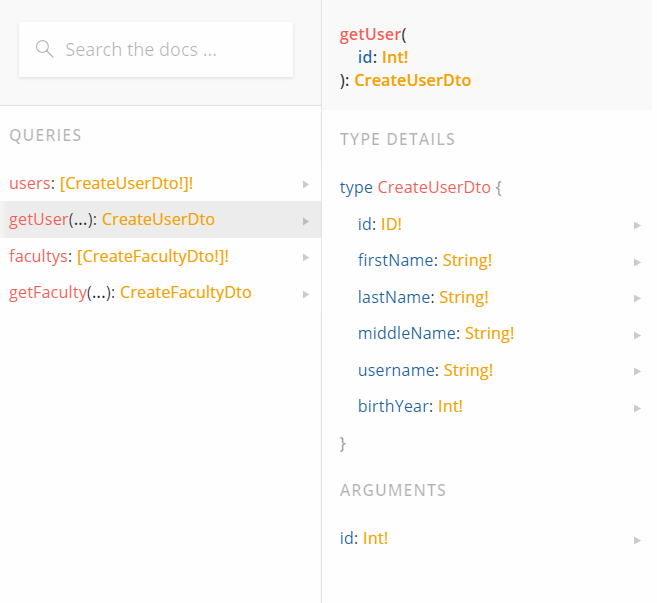


Рис. 10. QUERIES QL