МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Кемеровский государственный университет»
Институт цифры
Кафедра цифровых технологий

ОТЧЕТ
по производственной практике.
Научно-исследовательская работа.

студента 4 курса

Искандирова Марата Ринатовича

направление подготовки 02.03.02 Фундаментальная информатика и информационные технологии
направленность (профиль) подготовки «Информатика и компьютерные науки»

Руководитель практики:
канд. физ.-мат. наук, доцент,
доцент кафедры цифровых технологий,
Стуколов С.В.

Работа защищена:
“11” января 2024 г.
с оценкой отлично

КЕМЕРОВО 2024

Оглавление

ВВЕДЕНИЕ

В сфере детских садов современные информационные технологии приобретают все более весомое значение, внедряясь для оптимизации процессов и улучшения качества образования. На сегодняшний день существует неотложная необходимость в создании эффективных информационных систем, направленных на автоматизацию процессов диагностики детей и предоставление рекомендаций для возможности проведения воспитателями адаптационной деятельности, направленной на улучшение учебного взаимодействия с воспитанниками детского сада.

В детском садике 2 раза в год дети проходят тесты по разным критериям(мелкая моторика, письмо, речь и т.д). Далее для ребёнка высчитывается средний балл в каждой категории и на основе этих баллов, дети делятся на 3 категории: высокий, средний или низкий уровень развития. В условиях отсутствия какой-либо информационной системы для расчетов по диагностикам, воспитателям приходится тратить много времени на расчет средних баллов и вычисления категории для каждого ребенка, используя бумажные носители. В связи с использованием бумажных носителей составление рекомендаций по адаптированию подхода работы с детьми также вызывает затруднения ввиду сложности восприятия диагностической информации.

Актуальность работы

Система позволит цифровизовать бумажный учет, избавиться от необходимости ручных вычислений, просматривать статистические и аналитические данные, получать рекомендации по адаптации работы с детьми.

Итог - исключение возможности возникновения ошибки в расчетах (человеческий фактор), ускорение получения аналитических данных. Рекомендации позволят воспитателям лучше видеть общую картину успеваемости детей.

Цели и задачи разработки

Цель работы состоит в разработке информационной системы для автоматизации расчетов результатов диагностики детей с выводом рекомендаций.

Для достижения поставленной цели ставятся следующие задачи:

1. Анализ предметной области: изучение текущего характера подсчета диагностик, анализ формата диагностических документов, разбор специфики рекомендательных сообщений.
2. Анализ существующих решений: изучение существующих методов и подходов к автоматизации расчетов результатов диагностики детей с выводом рекомендаций.
3. Формулирование требований к разрабатываемой системе
4. Разработка архитектуры системы: определение структуры системы, выбор технологий и инструментов для реализации системы, проектирование взаимодействия между компонентами системы.
5. Проектирование базы данных

Исходя из поставленной цели, Искандирову Марату Ринатовичу по производственной практике в научно-исследовательской работе, проходящей с 01 сентября 2023 г. по 11 января 2024 г. были определены следующие задачи, которые необходимо выполнить:

- Анализ предметной области.
- Анализ существующих решений.

- Формулирование требований к разрабатываемой системе.
- Разработка архитектуры системы.
- Проектирование базы данных.

ГЛАВА 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И ФОРМУЛИРОВАНИЕ ТРЕБОВАНИЙ

1.1. Информационная система в детском саду: современные вызовы и потребности

Внедрение информационных технологий в детские сады представляет собой важный этап современного образования, направленный на улучшение процессов диагностики и анализа данных о развитии детей.

Одной из основных задач детских садов является диагностика детей с последующей адаптацией образовательного процесса. Отсутствие автоматизированных систем ведения учета и анализа диагностических данных приводит к необходимости использования ручных расчетов, что приводит к большим временным затратам и потенциальной возможностью возникновения ошибок в выявлении направления изменения воспитательного процесса.

Целью информационной системы для детских садов является автоматизация расчетов результатов диагностики и предоставление рекомендаций для дальнейшей работы с каждым ребенком и/или группой детей. Важно разработать систему, которая не только упростит ввод и редактирование диагностических данных, но и предоставит возможность анализа статистических данных для принятия обоснованных решений в педагогической практике.

1.2. Существующие инструменты автоматизации расчетов результатов диагностики детей с выводом рекомендаций

Для информационной поддержки детских садов уже существуют решения.

Так ООО Мирит [1] предоставляет полномасштабную ИС для детских садиков на государственном уровне АИС “ДОУ”. Однако в их системе полностью отсутствует функционал для работы с диагностиками.

1С:Дошкольное учреждение [2] также предоставляет полномасштабную ИС для детских садиков, но уже на платной основе. Однако в их системе тоже отсутствует функционал для работы с диагностиками.

Существующие продукты не предоставляют нужной для решения проблемы функциональности, вследствие чего было принято решение реализовывать систему.

1.3. Формулирование требований

Требования к разделению доступа определяют структуру прав и обязанностей в системе, гарантируя, что каждый пользователь имеет доступ только к тем ресурсам и функциям, которые соответствуют его роли.

1) **Аутентификация:** система должна предусматривать механизм аутентификации пользователей. Пользователи должны вводить уникальное имя пользователя и пароль для входа в систему. Вся информация, связанная с учетными данными, должна храниться в безопасном виде.

2) **Авторизация:** после успешной аутентификации пользователи получают доступ к функциям системы в соответствии с их ролью - "Воспитатель", "Заведующий", "Аналитик".
3) **Роли:** в системе должны быть предусмотрены следующие роли:

а) Воспитатель:

В числе возможностей воспитателя будет:

1. Занесение/редактирование диагностических данных.
2. Просмотр рекомендаций по всем детям (общая тенденция) и по конкретному ребенку.
3. Просмотр статистических данных.

б) Заведующий:

Дополнительно к возможностям воспитателя заведующий может:

1. создавать новых пользователей (воспитателей).
2. создавать/редактировать группы, добавлять/редактировать данные о детях.
3. создавать категории с подкатегориями для внесения данных по диагностике способностей ребенка. (для одной группы создавать бланк по диагностике в котором будут n категорий, у которых будут m подкатегорий с диапазоном возможных значений [min..max]).

в) Аналитик:

Аналитик может:

1. просматривать данные по садику (видеть у кого все хорошо и у кого все плохо).
2. добавлять/удалять/редактировать рекомендации по работе с детьми.

ГЛАВА 2. ПРОЕКТИРОВАНИЕ И МОДЕЛИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ

2.1. Проектирование архитектуры информационной системы

Для создания информационной системы была выбрана клиент-серверная архитектура, где в качестве клиента выступает веб-клиент, т.к. предполагается, что система может быть интегрирована в разные учреждения, в каждом из которых может быть установлена своя операционная система.

В составе системы предусматриваются следующие основные компоненты:

1. **Клиентская часть:** отображает необходимую информацию, предоставляет интерфейс для ввода данных и отвечает за взаимодействие пользователя с системой.
2. **Серверная часть:** отвечает за обработку запросов от клиентской части, управление базой данных и расчет результатов данных по диагностикам.

Связь между этими компонентами осуществляется по протоколу HTTP с использованием REST API интерфейса и представлена на рисунке 1. [3]

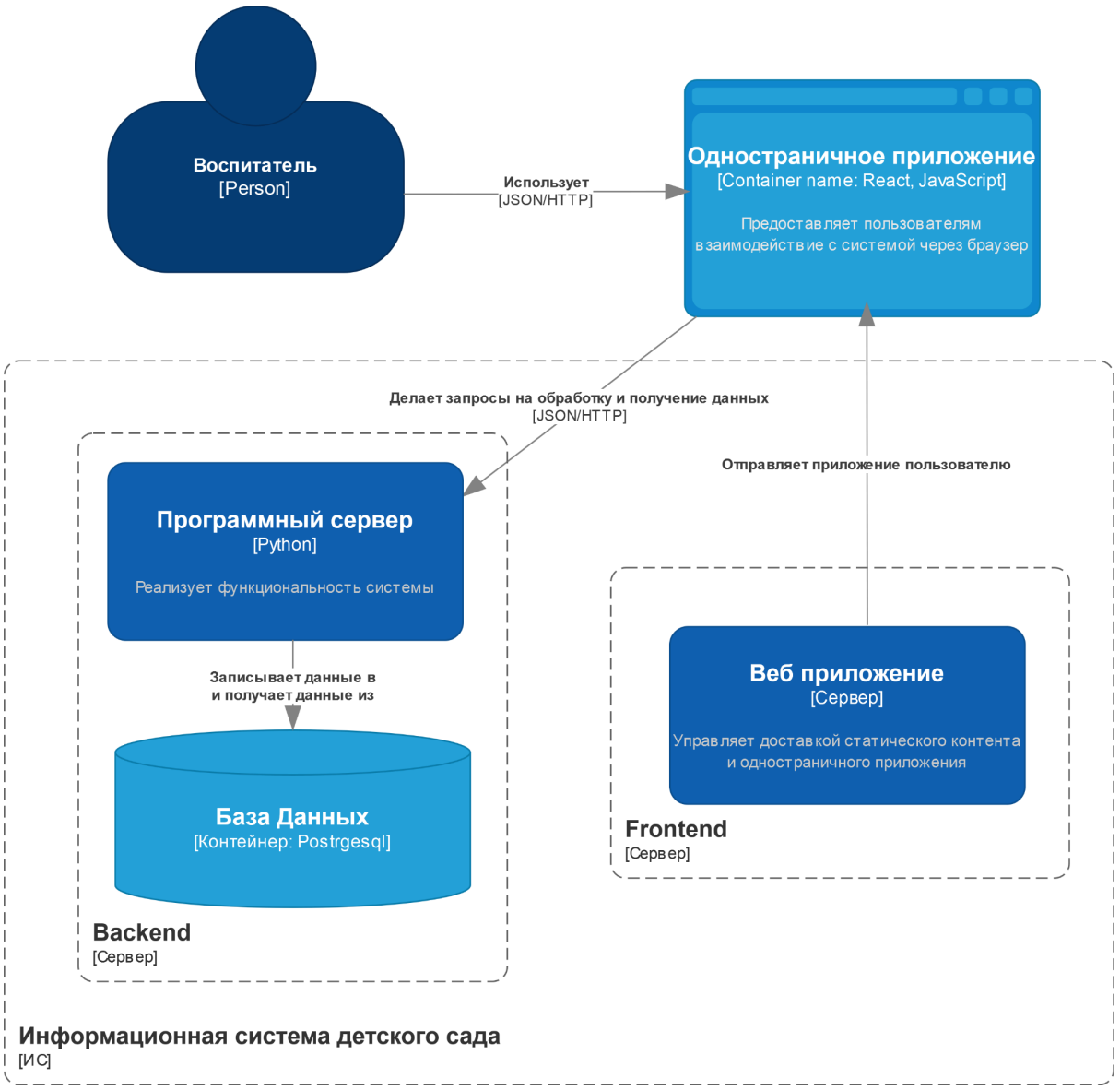

Рисунок 1 - архитектура системы.

Протокол передачи гипертекста (HTTP), составляющий основу современной сетевой коммуникации, устанавливает правила передачи и обработки сообщений, включая структуру и интерпретацию запросов и ответов.

Интерфейс REST API представляет собой архитектурный стиль, принятый для взаимодействия между компонентами системы, определяющий набор ограничений и свойств накладываемый на HTTP. Он позволяет упростить и структурировать взаимодействие между

компонентами, обеспечивает масштабируемость и расширяемость системы и её компонентов, что является важным атрибутом для эффективности и результативности всей системы.

На практике клиентская часть отправляет HTTP-запросы на сервер, используя стандартные REST методы, такие как GET, POST, PUT и DELETE. Серверная часть получает и интерпретирует эти запросы, выполняют необходимые операции и возвращают соответствующие ответы.

Например, клиент может отправить GET-запрос для получения списка групп в детском саду. Получив этот запрос, сервер обрабатывает данные, обращается к базе данных и в зависимости от успеха/неуспеха выполнения запроса отправляет клиенту ответ с данными.

Таким образом, протокол HTTP и интерфейс REST API обеспечивают прочную, эффективную и надежную связь между компонентами информационной системы. Эта архитектура облегчает интеграцию с различными детскими садами и способствует созданию гибкой и масштабируемой информационной системы.

2.1.1. Проектирование архитектуры клиентской части

В целом, архитектура клиентской части представляет собой набор взаимодействующих между собой блоков, каждый из которых имеет свою зону ответственности.

1. **View**: этот блок отвечает за отображение пользовательского интерфейса. Он представляет собой набор страниц и компонентов, отображающих информацию для пользователя и позволяющих взаимодействовать с системой.

2. **ViewModel**: содержит логику обработки взаимодействия пользователя с графическим интерфейсом, такие как обработчики нажатия кнопок, ввода в поля ввода и прочее. Этот блок является связующим звеном между пользовательским интерфейсом и другими частями системы, обеспечивая обработку действий пользователя и вызов соответствующих функций системы. Здесь же находится стейт-менеджмент приложения. Он отвечает за хранение состояния приложения, включая данные пользователя, текущее состояние интерфейса и прочее. Стейт-менеджмент позволяет обеспечить согласованность данных в приложении и упростить взаимодействие между различными частями системы.

3. **Model**: блок Model является связующим звеном между ViewModel и Transport. Здесь находится бизнес-логика приложения и DTO (дата классы).

4. **Transport**: блок Transport отвечает за взаимодействие клиентской и серверной частей. Он формирует соответствующие HTTP-запросы к серверу, обрабатывает ответы и осуществляет их преобразование к DTO классам из Model, что обеспечивает независимость других слоев от необходимости зависеть от текстовый формата обмена данными JSON. Transport также отвечает за обработку ошибок связи.

2.1.2. Проектирование архитектуры серверной части

Многоуровневая архитектура придерживается принципов модульности, что обеспечивает высокую расширяемость системы и надежную структуру с четким разделением задач. Эта архитектура состоит из отдельных блоков, каждый из которых следует независимому паттерну проектирования.

1. Infrastructure: самый внешний слой серверной части приложения. Содержит в себе входную точку и логику взаимодействия с

внешними частями приложения (БД).

Здесь находятся подслои:

A. **Controller:** является точкой входа в систему, управляет входящими запросами. Являясь интерфейсом между клиентом и бизнес логикой, он обрабатывает запросы и направляет их в соответствующие Сервисы. Контроллеры разработаны с акцентом на передачу и маршрутизацию данных, исключая выполнение какой-либо бизнес логики. Такой подход к проектированию обеспечивает четкое разделение обязанностей по отдельным блокам сервиса.

B. **Repository:** следуя паттерну проектирования Repository, выступает в качестве посредника между сервисами и базой данных. Этот блок реализует необходимые операции CRUD (Create, Read, Update, Delete), он абстрагирует взаимодействия с базой данных, благодаря чему обеспечивается отсутствие зависимости бизнес-логики от приложения.

2. Application: является основой логики приложения. Он взаимодействует с полученными от контроллеров данными, обрабатывает их, обращается к доменной бизнес-логике и передает/получает данные из базы данных.

Состоит из мелких классов - интеракторов, которые представляют собой реализацию различных вариантов использования приложения.

3. Domain: является основой любой бизнес-логики приложения. Здесь находится независимая логика, которую при необходимости можно переиспользовать в другом приложении.

Бизнес логика представлена методами проведения расчетов по диагностикам.

Более наглядно рассмотреть разделение слоев можно на рисунке 2

Синий - Infrastructure

Зеленый - Application

Красный - Domain

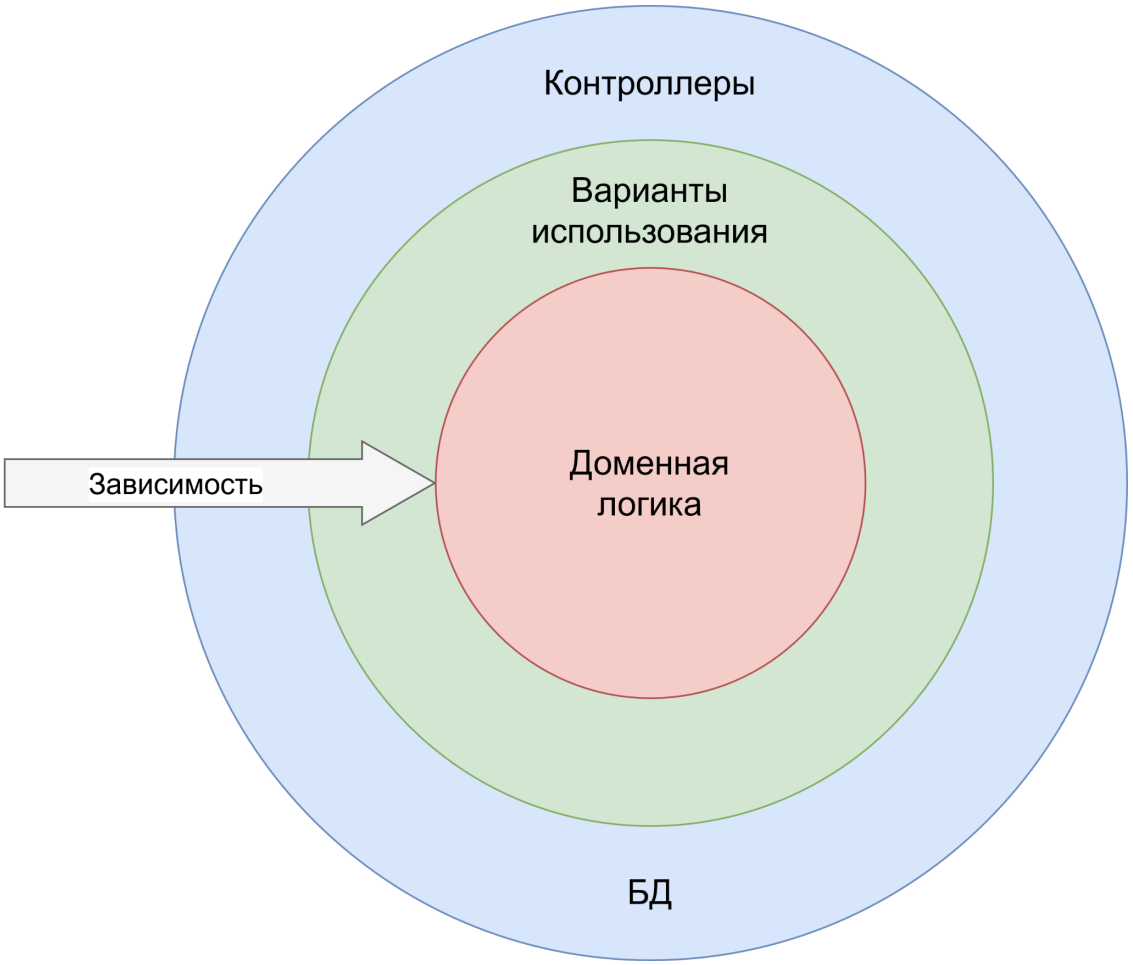

Рисунок 2 - диаграмма распределения слоев.

Приведенная на рисунке 3 диаграмма последовательности дает графическое представление того, как все архитектурные блоки взаимодействуют в системе при обработке стандартного запроса.

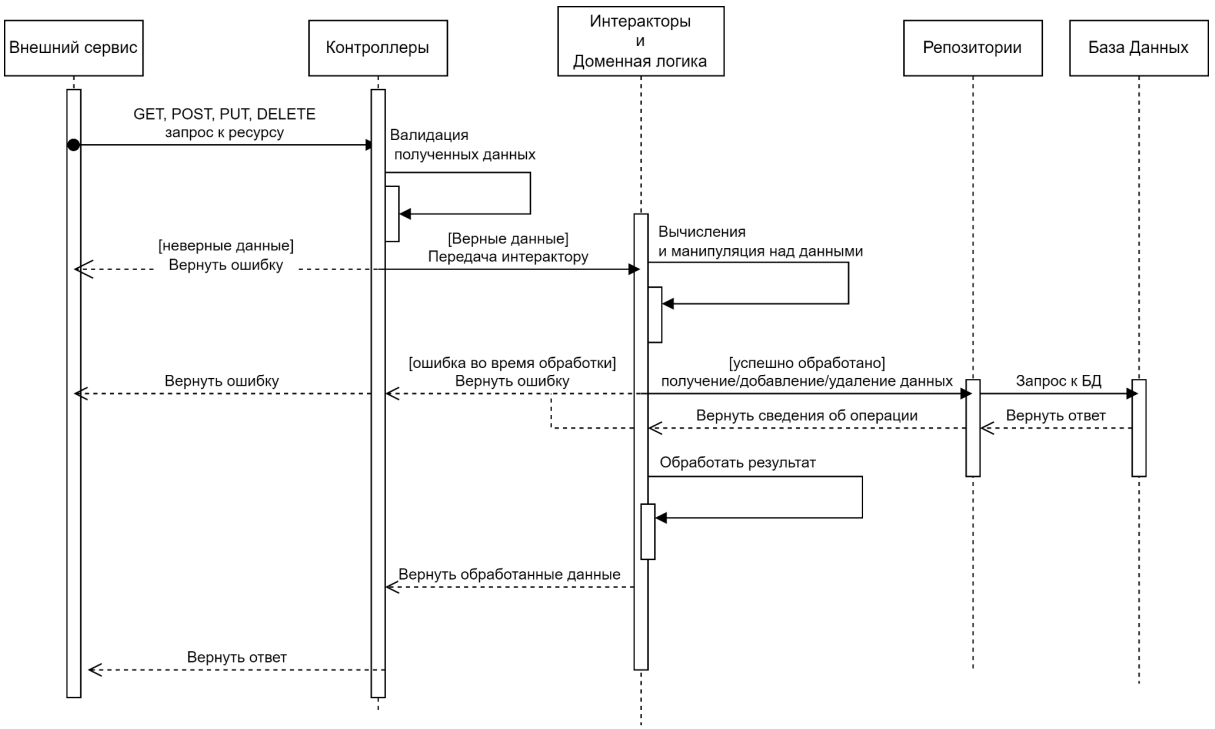

Рисунок 3 - общая диаграмма последовательности, показывающая взаимодействие блоков сервиса.

Взаимодействие начинается с отправки клиентом HTTP-запроса к системе. Этот запрос получает Контроллер, который декодирует запрос и извлекает соответствующие данные.

Затем Контроллер вызывает соответствующий метод в слое Application и передает в него преобразованные данные запроса. Дальнейшая логика (обращение к домену или репозиторию) определена в Интеракторе.

Полученные из Репозитория данные обрабатываются согласно правилам предметной области и возвращаются обратно Контроллеру.

Наконец, Контроллер из обработанных данных формирует HTTP-ответ, который затем отправляется обратно клиенту.

Эта последовательность взаимодействий демонстрирует, как архитектурные блоки работают вместе при обработке запросов. Каждый

блок придерживается определенного паттерна проектирования и имеет четко разграниченные обязанности, что благоприятно сказывается на возможностях масштабирования/изменения системы.

2.1.3. Проектирование базы данных

В базе данных [4] было выделено 7 таблиц:

1. workers - информация о работниках.
    - lastname - фамилия
    - name - имя
    - patronymic - отчество
    - date_birth - дата рождения
    - position - должность
2. groups - группы детей.
    - name - название группы
    - id_worker - id работника, прикпреплённого к этой группу
3. children - ФИО детей.
    - lastname - фамилия
    - name - имя
    - patronymic - отчество
4. groups_children - какой ребёнок в какой группе находится.
    - id_children - id ребёнка
    - id_group - id группы
5. categories - категории с подкатегориями для аналитик.
    - name - название категории
    - id_parent_category - id категории, к которой относится данная категория
6. category_result - результаты тестирования детей за конкретный год и категорию.
    - id_category - id категории для которой записывается результат

- id_children - ребенок который проходит диагностику
- score_start - количество баллов на начало года
- score_end - количество баллов на конец года
- year - год, за который проводится аналитика

7. recommendation - рекомендации для каждой категории.
   id_category - id категории
   - hight_recommendation - рекомендации для детей с высоким уровнем баллов
   - medium_recommendation - рекомендации для детей с средним уровнем баллов
   - low_recommendation - рекомендации для детей с низким уровнем баллов

Диаграмма базы данных представлена на на рисунке 4.

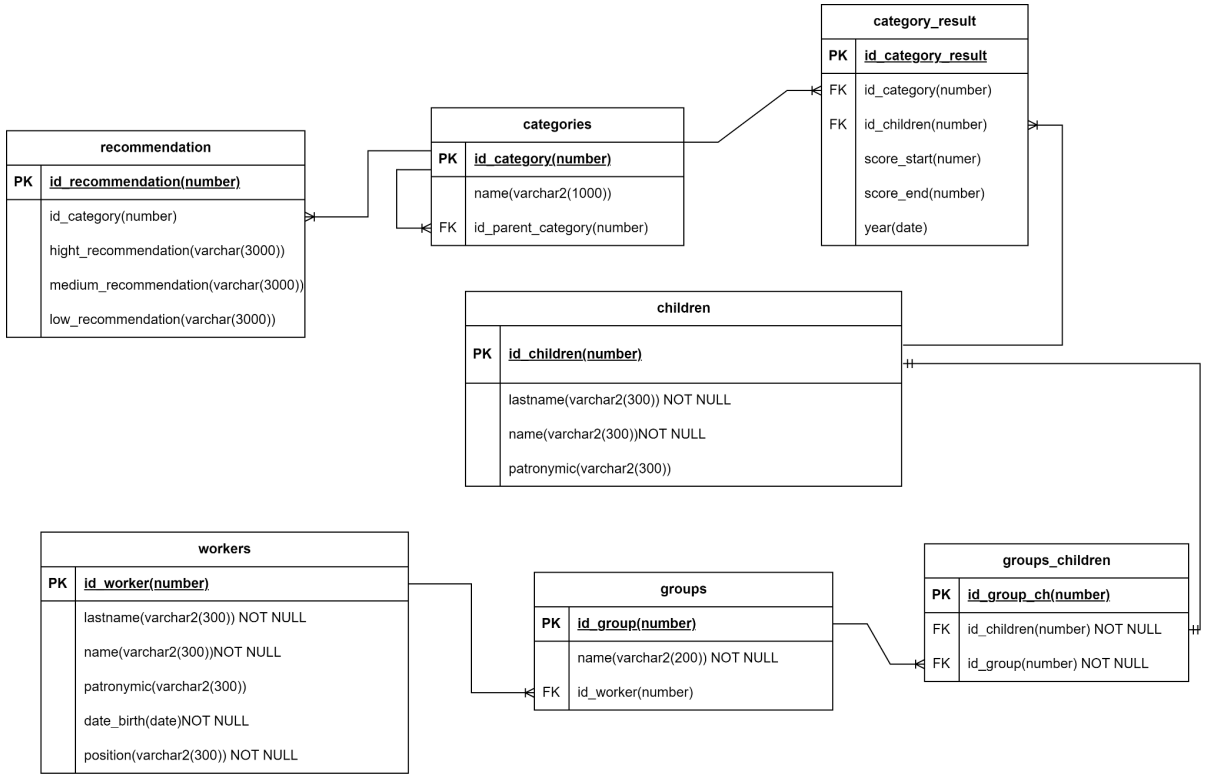

Рисунок 4 - диаграмма базы данных.

Заключение

Исходя из описанной проблематики в области диагностики детей в детском саду и необходимости автоматизации процессов, разработка информационной системы для автоматизации расчетов результатов диагностики и предоставления рекомендаций представляется крайне актуальной и значимой.

Цель разработки данной системы - облегчить и ускорить процесс обработки информации о детском развитии, снизить вероятность ошибок из-за человеческого фактора и позволить воспитателям фокусироваться на более качественном взаимодействии с воспитанниками, имея доступ к более точной и оперативной информации о их развитии.

В ходе выполнения анализа предметной области и изучения существующих решений было выявлено, что для решения поставленной проблемы нет готовых решений.

Следующим этапом были сформулированы требования, полностью удовлетворяющие потребности заказчика.

Была разработана системная архитектура клиентской и серверной частях таким образом, чтобы при реализации оставалась возможность доработки/изменений.

Выполненное проектирование базы данных позволит хранить все нужные для системы данные, предотвращая возможные проблемы целостности данных, такие как дублирование или потеря информации, при этом допускается возможность дальнейшего расширения БД ввиду изменения бизнес-правил.

Список литературы

1) МИРИТ. [Электронный ресурс]. Режим доступа URL: https://mirit42.ru/ (дата обращения 07.01.2023)
2) 1С:Дошкольное учреждение. [Электронный ресурс]. Режим доступа URL: https://solutions.1c.ru/catalog/preschool (дата обращения 07.01.2023)
3) Habr. Как описать большую систему в нотации C4 [Электронный ресурс]. Режим доступа URL: https://habr.com/ru/companies/nspk/articles/679426/ (дата обращения 07.01.2023)
4) Habr. Основы правил проектирования базы данных [Электронный ресурс]. Режим доступа URL: https://habr.com/ru/articles/514364/ (дата обращения 07.01.2023)