МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий (ИИТ)
Кафедра практической и прикладной информатики (ППИ)

ИТОГОВЫЙ ОТЧЕТ

по дисциплине «Проектирование баз данных»

Студент группы ИКБО-33-22 Шило Ю. С. _____
(подпись)

Ассистент Чучаева С. М. _____
(подпись)

Отчет представлен «___»_____2024г.

Москва 2024 г

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Проектирование - важный этап в разработке любого проекта, который определяет архитектуру и функционал будущей системы. Оно позволяет предотвратить ошибки, оптимизировать работу системы и уменьшить затраты на ее разработку и поддержку. В данном отчете буду представлены различные модели и диаграммы, используемые в процессе проектирования, а также рассмотрим логическую и физическую модели базы данных. Цель проектирования - создать систему, которая эффективно работает и помогает достичь поставленных целей. Важность проектирования заключается в создании надежной и качественной системы, что способствует снижению рисков, улучшению ее работы и развитию в будущем.

ПРЕДМЕТНАЯ ОБЛАСТЬ

Предметная область данного проектирования — организация работы календаря игровых новинок.

Целью проектирования является создание системы, которая позволит эффективно организовать системы работы календаря. С продумывание разных возможный ситуаций, встречающихся в системах такого формата.

МОДЕЛЬ НОТАЦИИ IDEF0

IDEF0 (Integrated Definition for Function Modeling) — интегрированное определение для моделирования функций, представляет собой стандартный метод моделирования функциональных процессов, который широко применяется в области бизнес-анализа и проектирования информационных систем. С помощью модели IDEF0 можно описать структуру и взаимосвязи между функциональными процессами, а также выявить необходимые входные

Модель IDEF0 включает в себя следующие элементы:

- Функция - основной элемент модели, представляющий собой процесс преобразования входных данных в выходные.
- Входные данные - информация, необходимая для выполнения функции.
- Выходные данные - результат выполнения функции.
- Механизмы - ресурсы, необходимые для выполнения функции, такие как люди, оборудование, программное обеспечение и т.д.
- Управление - внешние факторы, влияющие на выполнение функции, такие как правила, стандарты, политики и т.д.

С помощью модели IDEF0 можно создать графическое представление функциональных процессов в виде диаграмм, которые помогают визуализировать структуру и взаимосвязи между процессами, а также идентифицировать необходимые входные и выходные данные.

В процессе проектирования системы календаря игровых новинок, модель IDEF0 может быть использована для описания процессов как регистрация игры или человека и отправка уведомления пользователю. Диаграммы IDEF0 помогут визуализировать структуру и взаимосвязи между этими процессами, а также идентифицировать необходимые входные и выходные данные.

В рамках данной предметной области была разработана модель в

нотации IDEF0. На Рисунке 1 представлена контекстная диаграмма данного процесса.

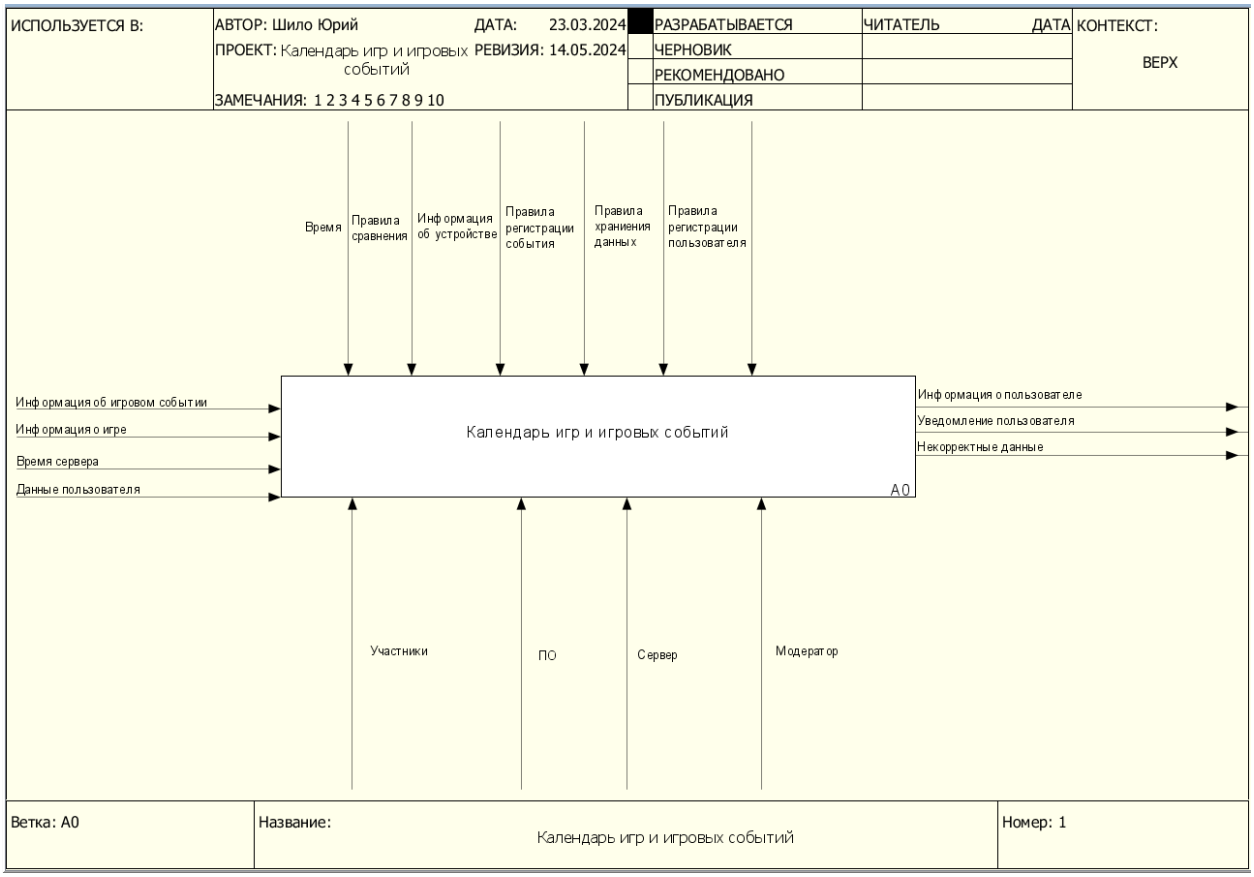

Рисунок 1 - Контекстная диаграмма «Календарь игровых событий» в методологии IDEF0

Основной блок:

- Календарь игр и игровых событий

Входная информация:

- Информация об игровом событии
- Информация о игре
- Время сервера
- Данные пользователя

Выходная информация:

- Информация о пользователе
- Уведомление пользователя
- Некорректные данные

Управляющие:

- Время
- Правила сравнения
- Информация об устройстве
- Правила хранения данных
- Правила регистрации пользователя

Механизмы:

- Участники
- ПО
- Модератор
- Сервер

Декомпозируем общий блок «Календарь игр и игровых событий» на связанные между собой элементы. Получим три основных этапа (Рисунок 2):

- Регистрация пользователя
- Регистрация игры
- Представление данных о событии

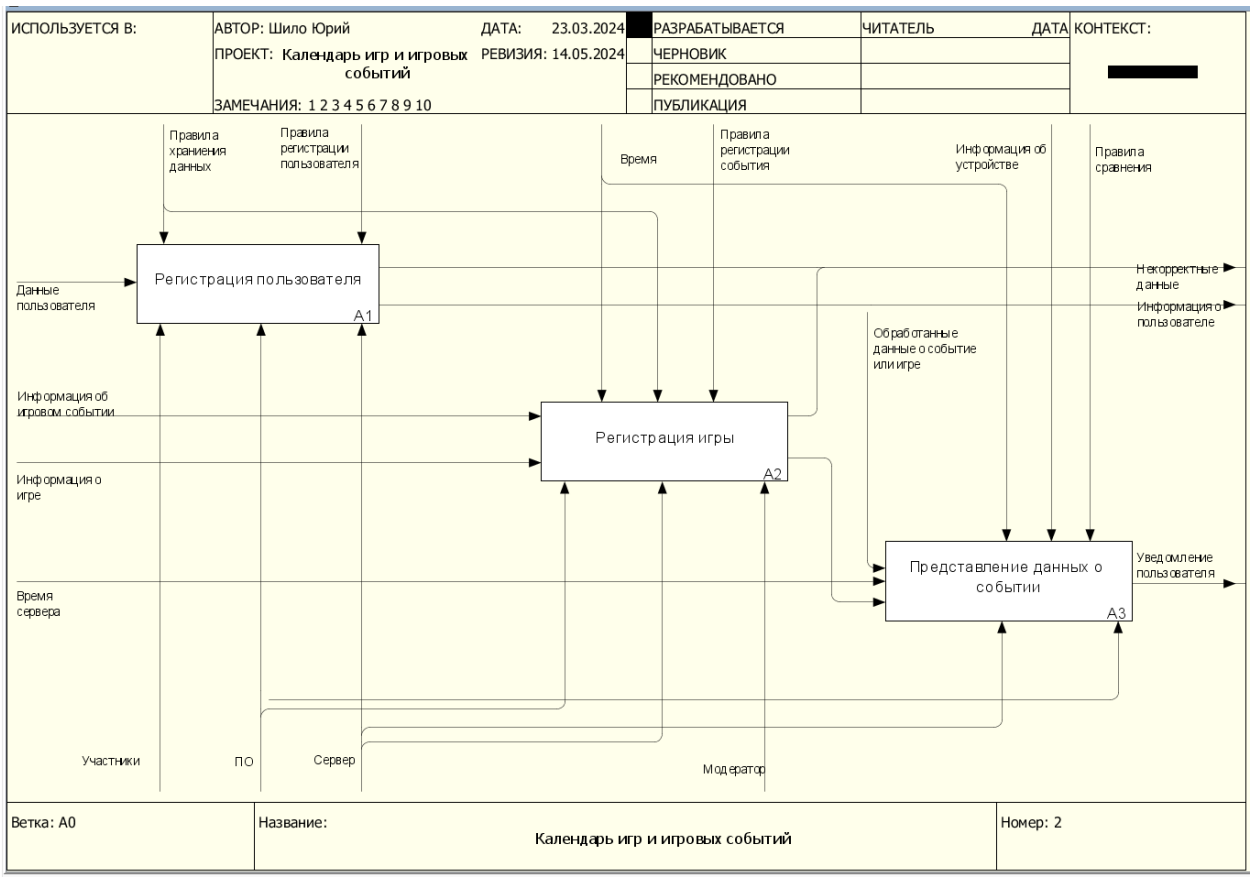

Рисунок 2 - Декомпозиция контекстной диаграммы в методологии IDEF0

Блок «Регистрация пользователя» мы декомпозируем на 3 этапа (Рисунок 3):

- Ввод данных пользователем
- Проверка корректности данных
- Добавление данных на сервер

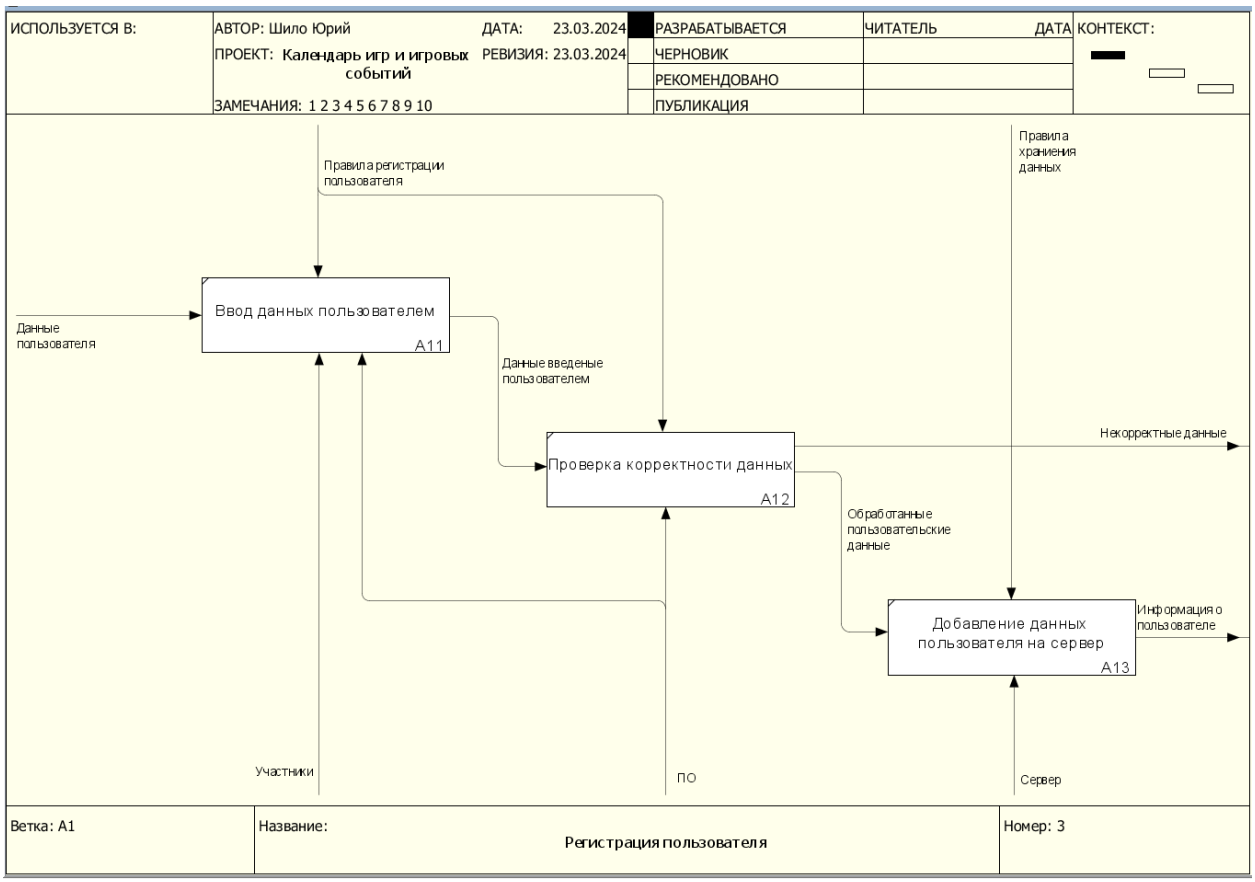

Рисунок 3 – Декомпозиция блока «Регистрация пользователя» в методологии IDEF0

Блок «Регистрация игры» мы декомпозируем на 3 этапа (Рисунок 4):

- Ввод данных о событии или игре
- Проверка корректности данных
- Добавление обработанных данных на сервер

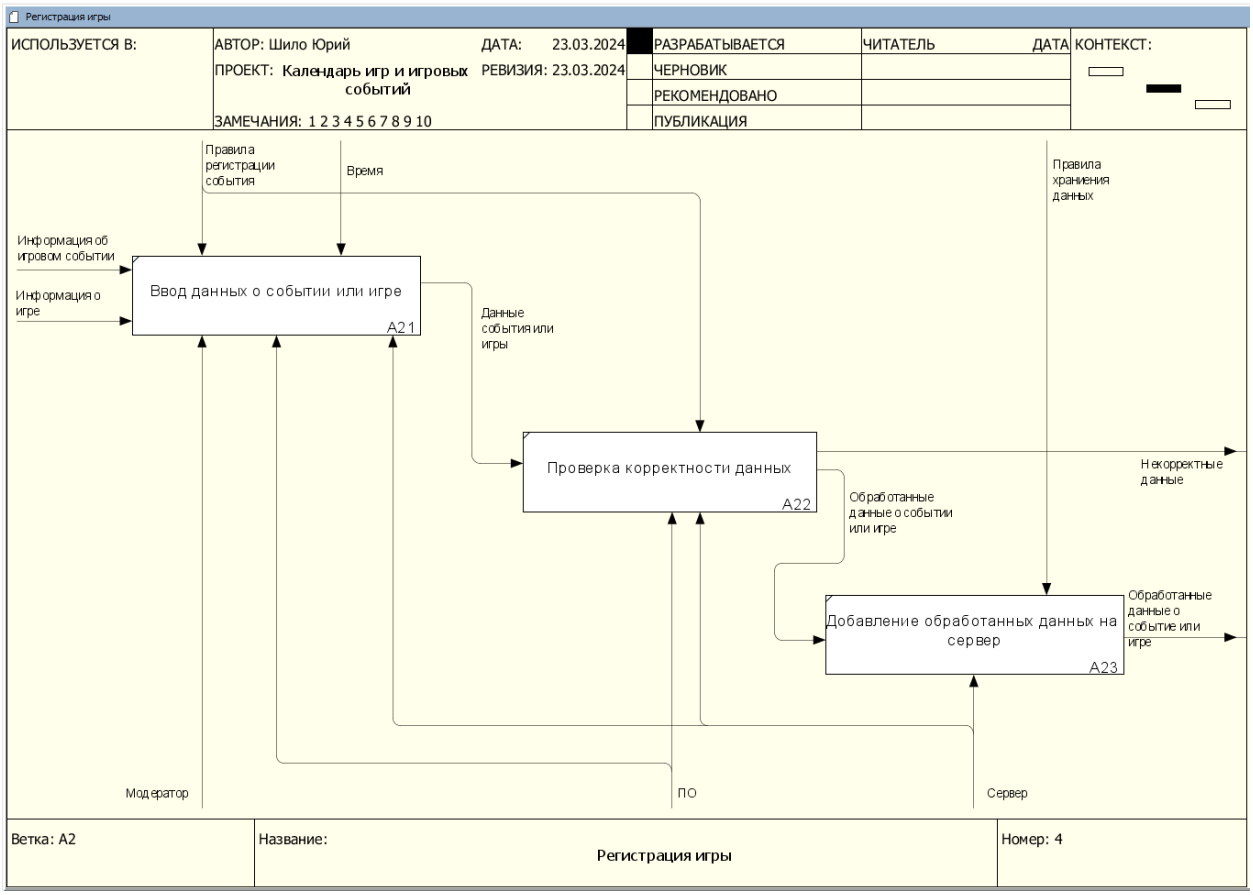

Рисунок 4 – Декомпозиция блока «Регистрация игры» в методологии IDEF0

Блок «Предоставление данных о событии» мы декомпозируем на 3 этапа (Рисунок 5):

- Сопоставление данных
- Проверка времени
- Отправка уведомления

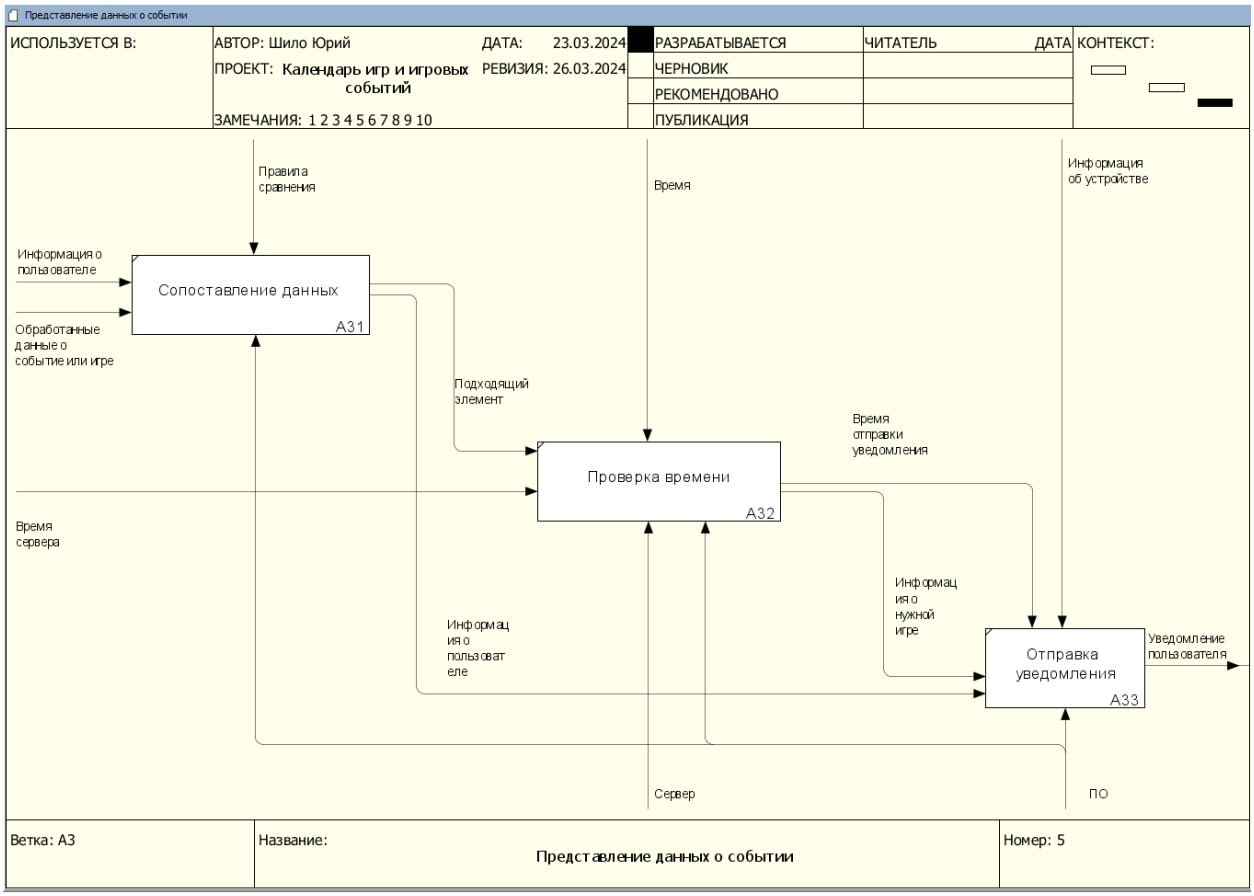

Рисунок 5 – Декомпозиция блока «Предоставление данных о событии» в методологии IDEF0

МОДЕЛЬ НОТАЦИИ DFD

Модель DFD (Data Flow Diagram) – это один из распространенных стандартов моделирования процессов, широко применяемый в бизнес-анализе и разработке информационных систем. Она позволяет наглядно представить передачу данных между различными процессами и внешними источниками, а также изменения данных внутри этих процессов.

Модель DFD включает в себя следующие элементы:

- Процесс - основной элемент модели, представляющий собой преобразование входных данных в выходные.
- Поток данных - перемещение данных между процессами или между процессом и внешним источником.
- Хранилище данных - места хранения данных, необходимых для процессов.
- Внешний источник - внешние сущности, которые взаимодействуют с системой, такие как люди, другие системы, устройства и т.д.

С помощью модели DFD можно создать графическое представление процессов в виде диаграмм, которые помогают визуализировать потоки данных между процессами и внешними источниками, а также преобразования данных внутри процессов.

Была разработана модель DFD по предметной области «Календарь игровых событий» (Рисунки 6-9).

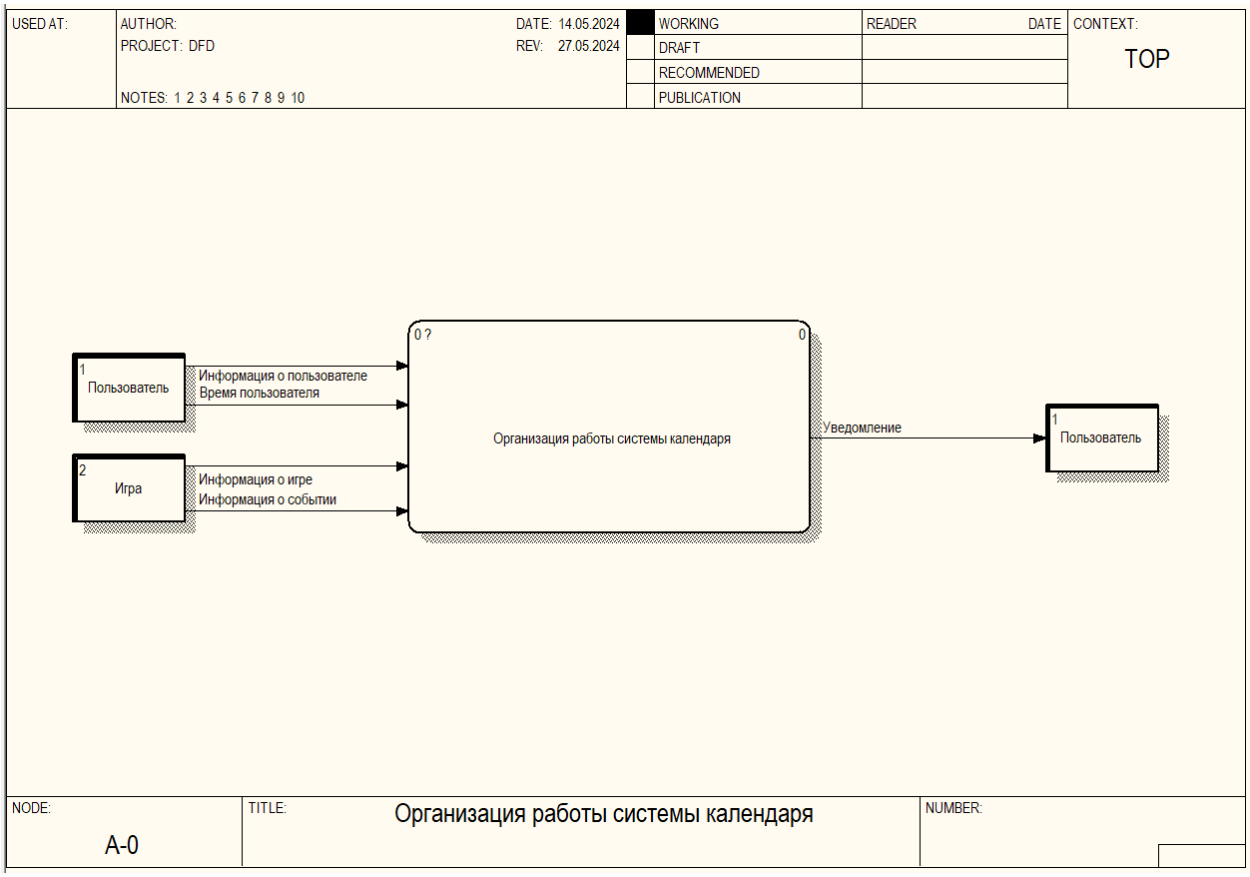

Рисунок 6 - Контекстная диаграмма «Организация работы системы календаря» в нотации DFD

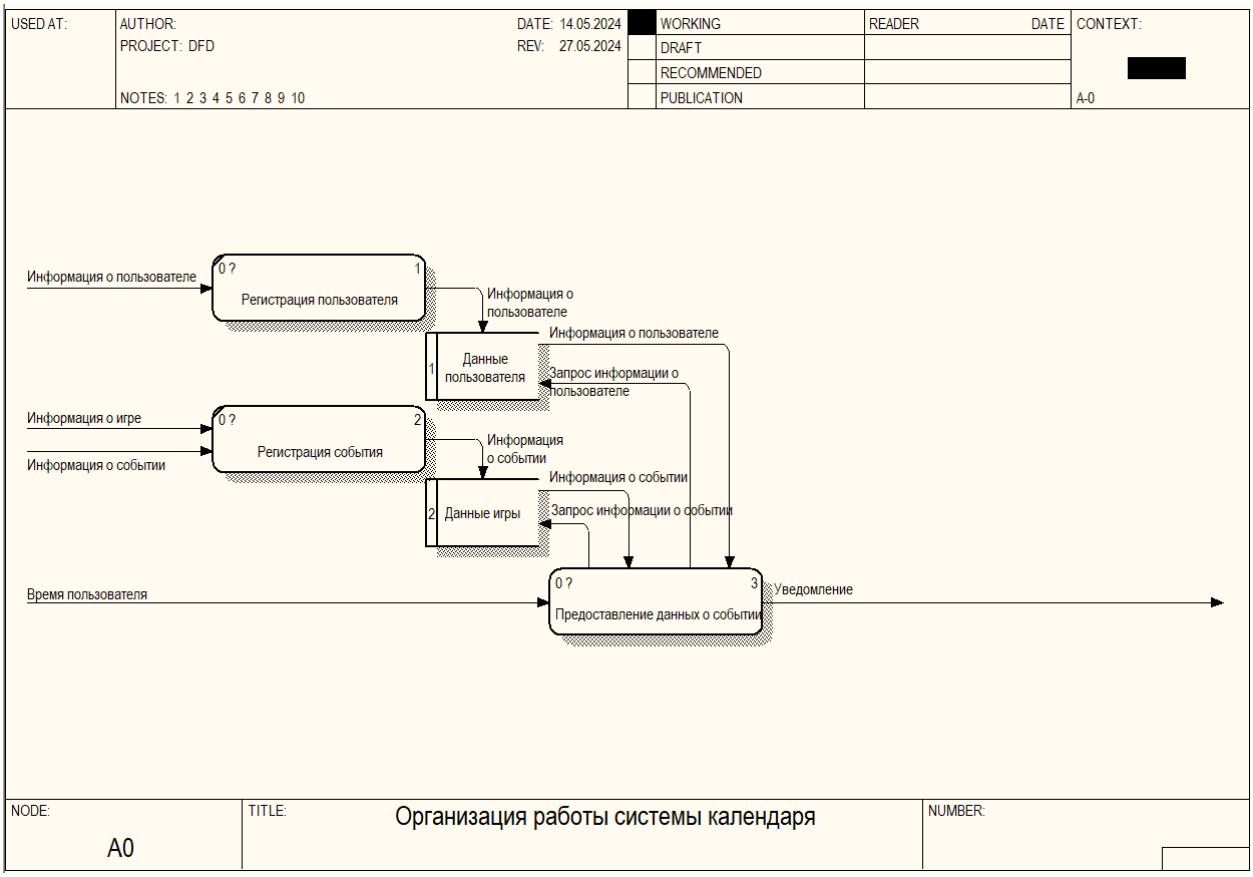

Рисунок 7 - Декомпозиция контекстной диаграммы «Организация работы

системы календаря» в методологии DFD

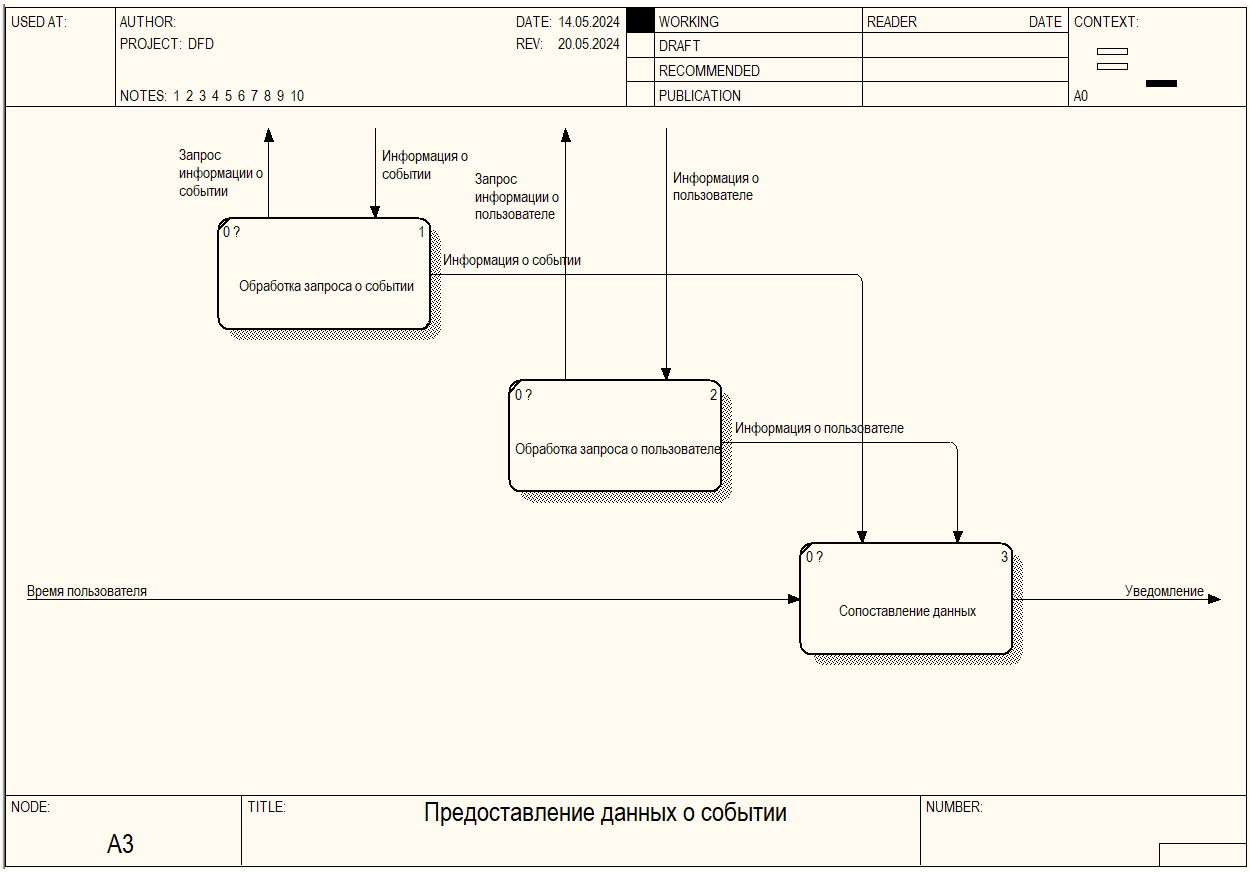

Рисунок 8 - Декомпозиция блока «Предоставление данных о событии» в методологии DFD

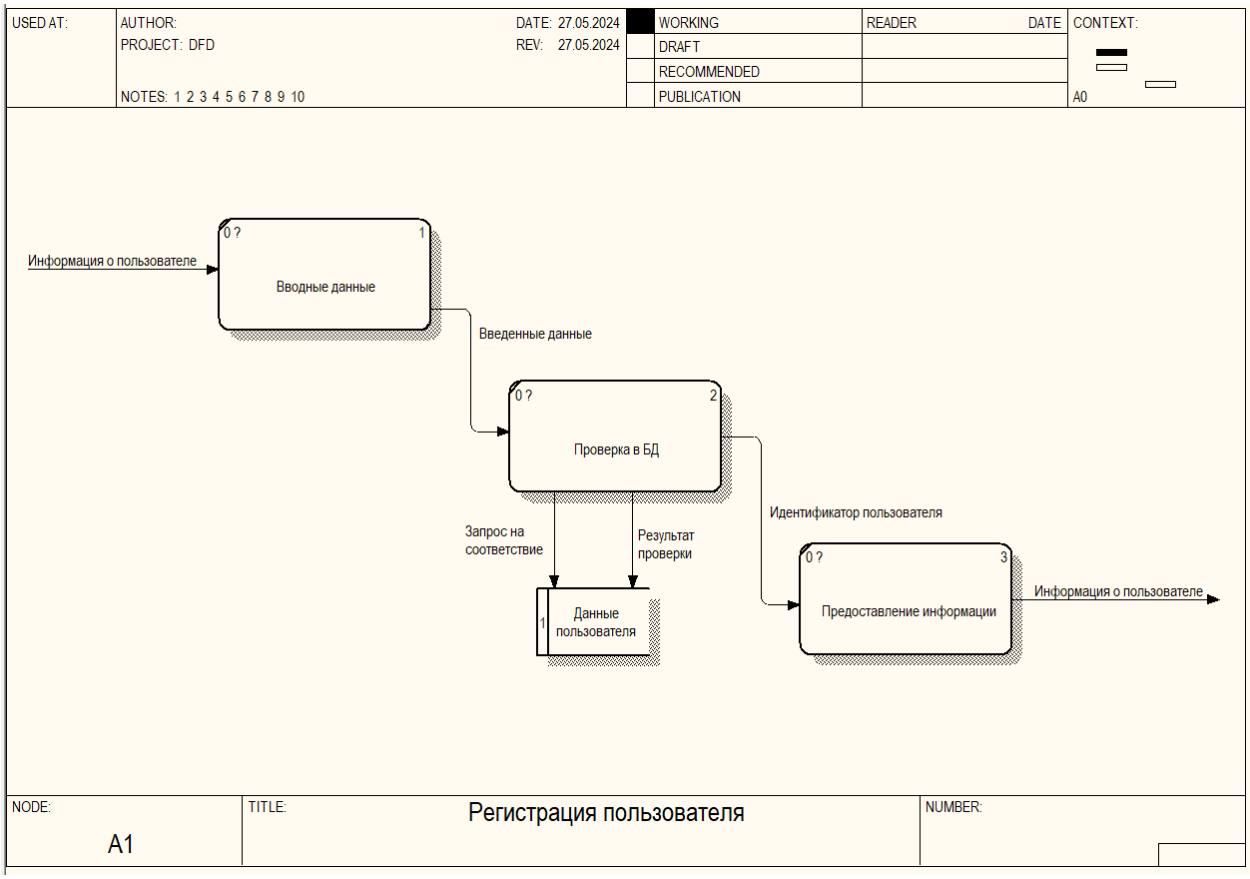

Рисунок 9 - Декомпозиция блока «Регистрация пользователя» в методологии DFD

На данной модели отображается основной процесс (сама система в целом) и ее связи с внешней средой (внешними сущностями). Это взаимодействие показывается через потоки данных.

Внешние сущности изображают входы в систему и/или выходы из нее. Для процесса «Взаимодействие с системой календаря» были выделены такие внешние сущности, как:

- Пользователь

Стрелки (потоки данных) описывают движение объектов из одной части системы в другую.

Хранилище данных. В отличие от стрелок, описывающих объекты в движении, хранилища данных изображают объекты в покое. Для рассматриваемого процесса были выделено такое хранилище как: «Данные игры» и «Данные пользователя»

ПРОЕКТИРОВАНИЕ НА ЯЗЫКЕ UML

ДИАГРАММА ВАРИАНТОВ ИСПОЛЬЗОВАНИЯ

Диаграмма вариантов использования (Use Case Diagram) является одним из стандартов моделирования требований в разработке программного обеспечения. Диаграмма вариантов использования позволяет описать функциональные требования к системе, представляя ее в виде набора вариантов использования (use cases), которые описывают взаимодействие пользователей с системой.

Диаграмма вариантов использования включает в себя следующие элементы:

1. Актёры (Actors) - внешние сущности, которые взаимодействуют с системой, такие как пользователи, другие системы, устройства и т.д.
2. Варианты использования (Use Cases) - описание набора действий, которые выполняет система для достижения определенной цели.
3. Отношения между актёрами и вариантами использования - связи между актёрами и вариантами использования, которые показывают, какие варианты использования выполняются каждым актёром.

С помощью диаграммы вариантов использования можно создать графическое представление функциональных требований к системе, которое помогает визуализировать взаимодействие пользователей с системой и определить основные функции, которые должна выполнять система.

На Рисунке 10 представлена диаграмма для предметной области «Календарь игровых новинок».

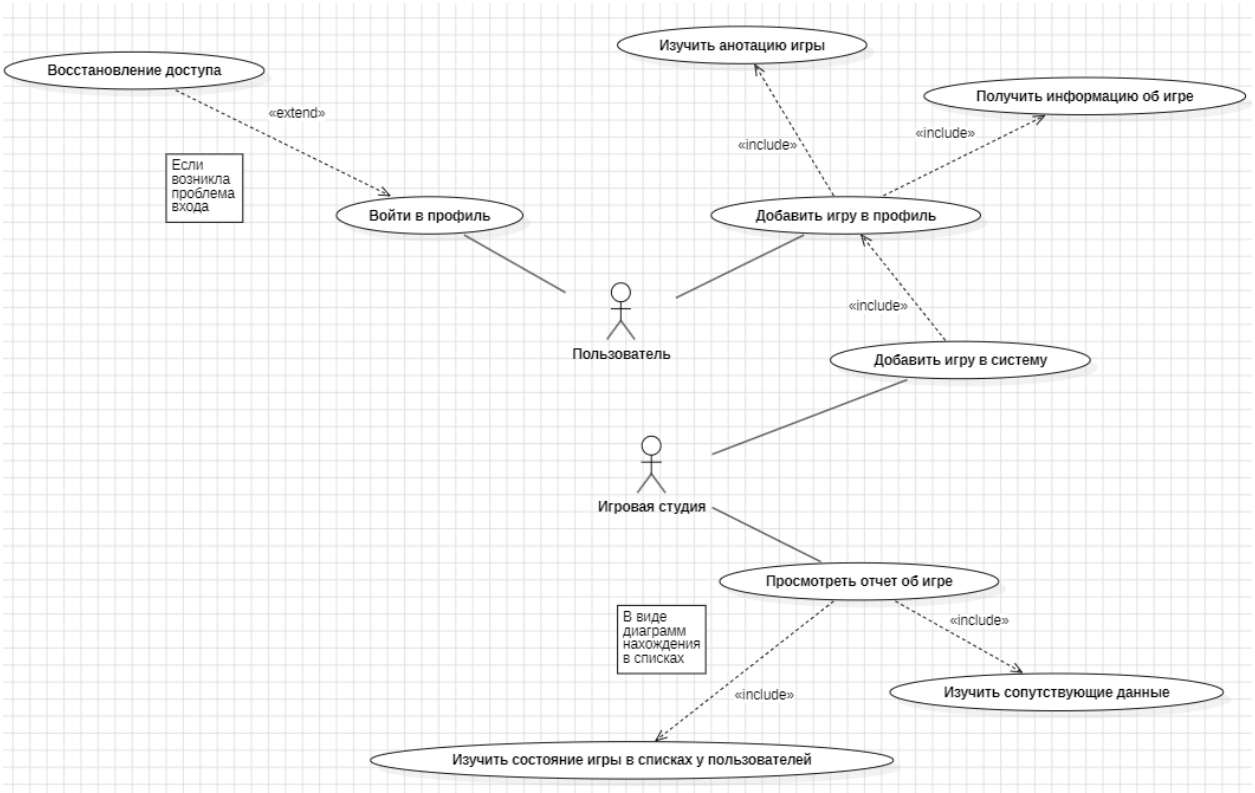

Рисунок 10 - Диаграмма вариантов использования для «Календарь игровых новинок»

ДИАГРАММА КЛАССОВ

Диаграмма классов — это ключевой инструмент объектно-ориентированного моделирования, позволяющий наглядно отобразить структуру системы и взаимосвязи между ее элементами. Она содержит классы, представляющие основные понятия или объекты в системе, а также показывает связи между ними.

На диаграмме классов классы отображаются в виде прямоугольников, которые разделены на три части: верхняя часть содержит название класса, в середине перечислены атрибуты класса, а в нижней части перечислены методы класса.

Атрибуты класса — это особенности или свойства класса, которые определяют его состояние и могут быть отображены в виде текстовых меток в центре прямоугольника класса. Методы класса — это действия или функции, которые может выполнять класс и которые могут быть представлены в виде текстовых меток в нижней части прямоугольника класса.

Отношения между классами — это связи, которые могут быть изображены в виде стрелок или линий между прямоугольниками классов. Существует несколько видов связей между классами, таких как ассоциация, агрегация, композиция, наследование и реализация.

Ассоциация — это связь между двумя классами, определяющая отношение между их экземплярами. Эта связь может быть изображена линией между прямоугольниками классов с описанием типа ассоциации.

Агрегация и композиция являются более узкими формами ассоциации, определяющими отношение "часть-целое" между классами. При агрегации один класс является частью другого класса, но может существовать независимо от него. В случае композиции один класс не может существовать без другого.

Наследование — это отношение между классами, при котором один класс наследует свойства и методы другого класса. Это отношение может быть изображено стрелкой между прямоугольниками классов с указанием типа

наследования.

Реализация — это отношение между классами, при котором один класс реализует интерфейс, заданный другим классом. Эта связь может быть изображена стрелкой между прямоугольниками классов с описанием типа реализации.

Диаграмма классов является важным инструментом при проектировании объектно-ориентированных систем, так как она позволяет визуально представлять структуру системы и ее взаимосвязи. Она используется для определения классов, их атрибутов, методов и связей между ними, а также для определения требований к системе и ее архитектуры.

Диаграмма представлена на Рисунке 11.

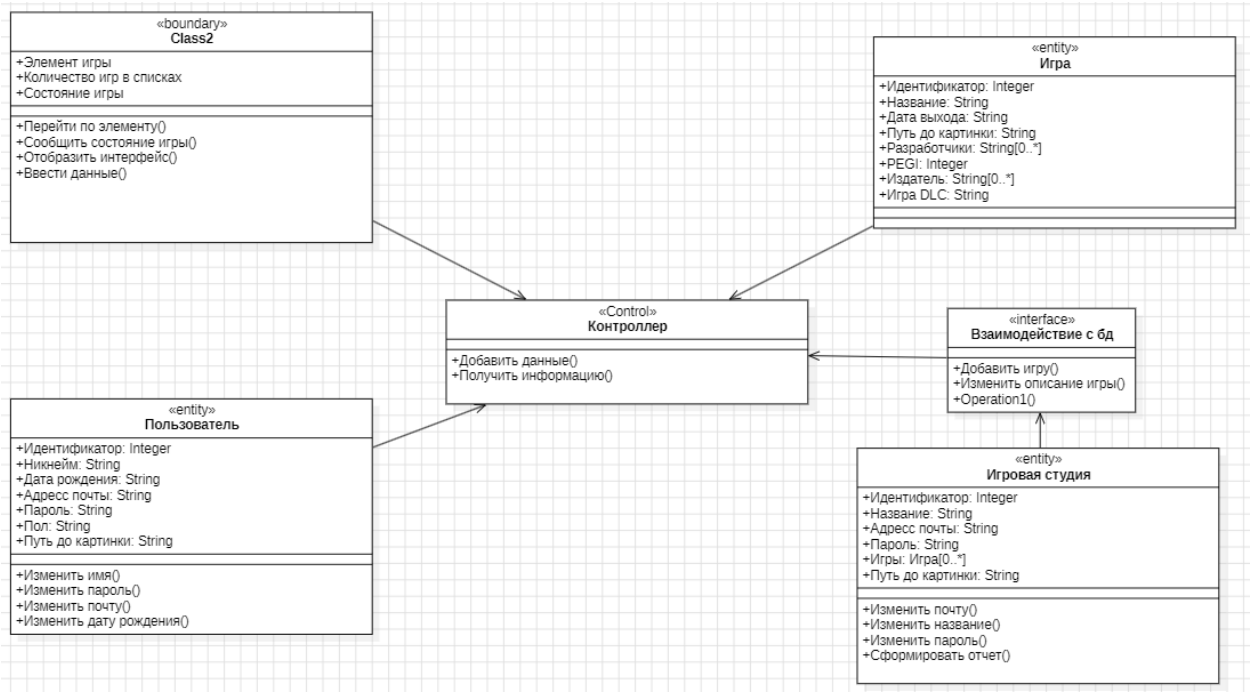

Рисунок 11 - Диаграмма классов для варианта использования «Календарь игровых новинок»

ДИАГРАММА ПОСЛЕДОВАТЕЛЬНОСТИ

Диаграмма последовательности - один из видов диаграмм взаимодействия, который используется для визуализации взаимодействия между объектами или акторами в системе через последовательность сообщений. Она отражает, какие сообщения передаются между объектами и в какой последовательности они передаются.

Структура диаграммы последовательности включает набор горизонтальных линий, представляющих объекты или акторов, и набор вертикальных линий, представляющих передаваемые сообщения между объектами. Каждое сообщение изображается стрелкой, направленной от одного объекта к другому, с текстовой меткой, определяющей тип сообщения.

На диаграмме последовательности различаются два типа стрелок: синхронные и асинхронные. Синхронные стрелки представляют мгновенное передачу сообщений без задержек, в то время как асинхронные стрелки отображают сообщения, передаваемые с задержкой, например, в случае использования очередей сообщений.

Диаграмма последовательности может быть использована для моделирования различных сценариев взаимодействия между объектами, регистрация игры (Рисунок 12).

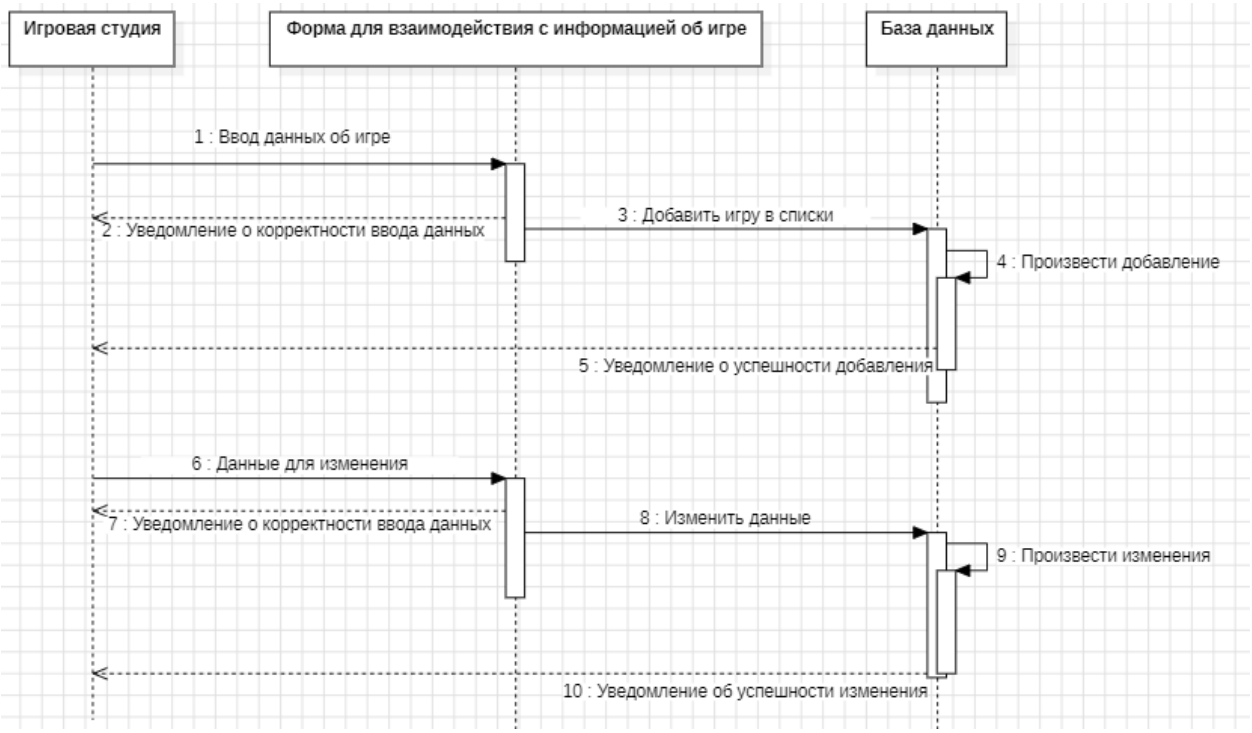

Рисунок 12 - Диаграмма последовательности для варианта использования «Регистрация игры»

Диаграмма последовательности является важным инструментом при проектировании систем, так как она позволяет визуально представлять взаимодействие между объектами и идентифицировать потенциальные проблемы и узкие места в системе. Она используется для определения потока данных и контроля между объектами, а также для определения требований к системе и ее архитектуры.

ДИАГРАММА КООПЕРАЦИЙ

Диаграмма объектных взаимодействий, также известная как диаграмма кооперации, используется для визуализации взаимодействия между объектами в системе. Она показывает, как объекты передают сообщения и выполняют операции, чтобы достичь определенной цели.

На диаграмме объектных взаимодействий объекты изображены в виде прямоугольников, а сообщения - в виде стрелок, соединяющих объекты. Каждая стрелка представляет вызов метода или операции одного объекта другим.

Объекты на диаграмме могут быть изображены как прямоугольники с указанием класса и идентификатора. Сообщения представлены стрелками, направленными от одного объекта к другому, с текстовой меткой, описывающей тип сообщения.

Реализация данной диаграммы для варианта использования «Регистрация игры» представлена на Рисунке 13.

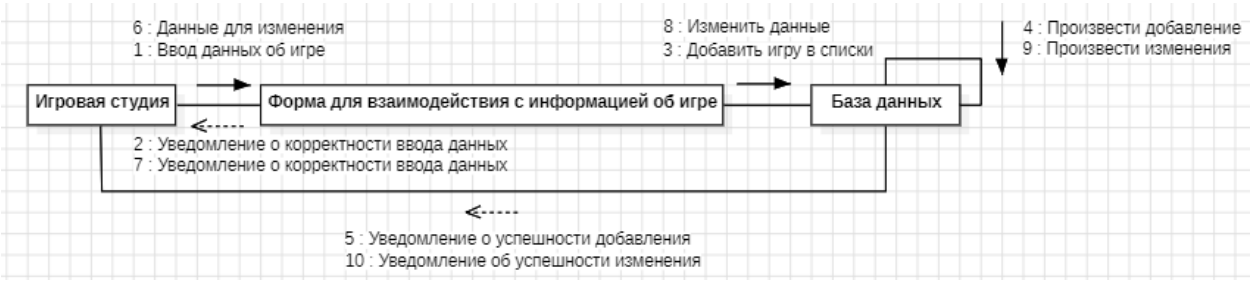

Рисунок 13 - Диаграмма кооперации для варианта использования «Регистрация игры»

ДИАГРАММА СОСТОЯНИЙ

Диаграмма состояний представляет собой графическое изображение всех возможных состояний объекта или системы, а также переходов между этими состояниями в ответ на определенные события. Состояния обозначаются окружностями, а переходы - стрелками, соединяющими состояния. Каждое состояние на диаграмме представляет определенное состояние объекта или системы, а переходы показывают события, приводящие к изменению состояния.

На диаграмме состояний каждое состояние обычно имеет название, указанное внутри окружности, а переходы представлены стрелками, направленными от одного состояния к другому. На стрелках указывается текст, описывающий событие, приводящее к переходу.

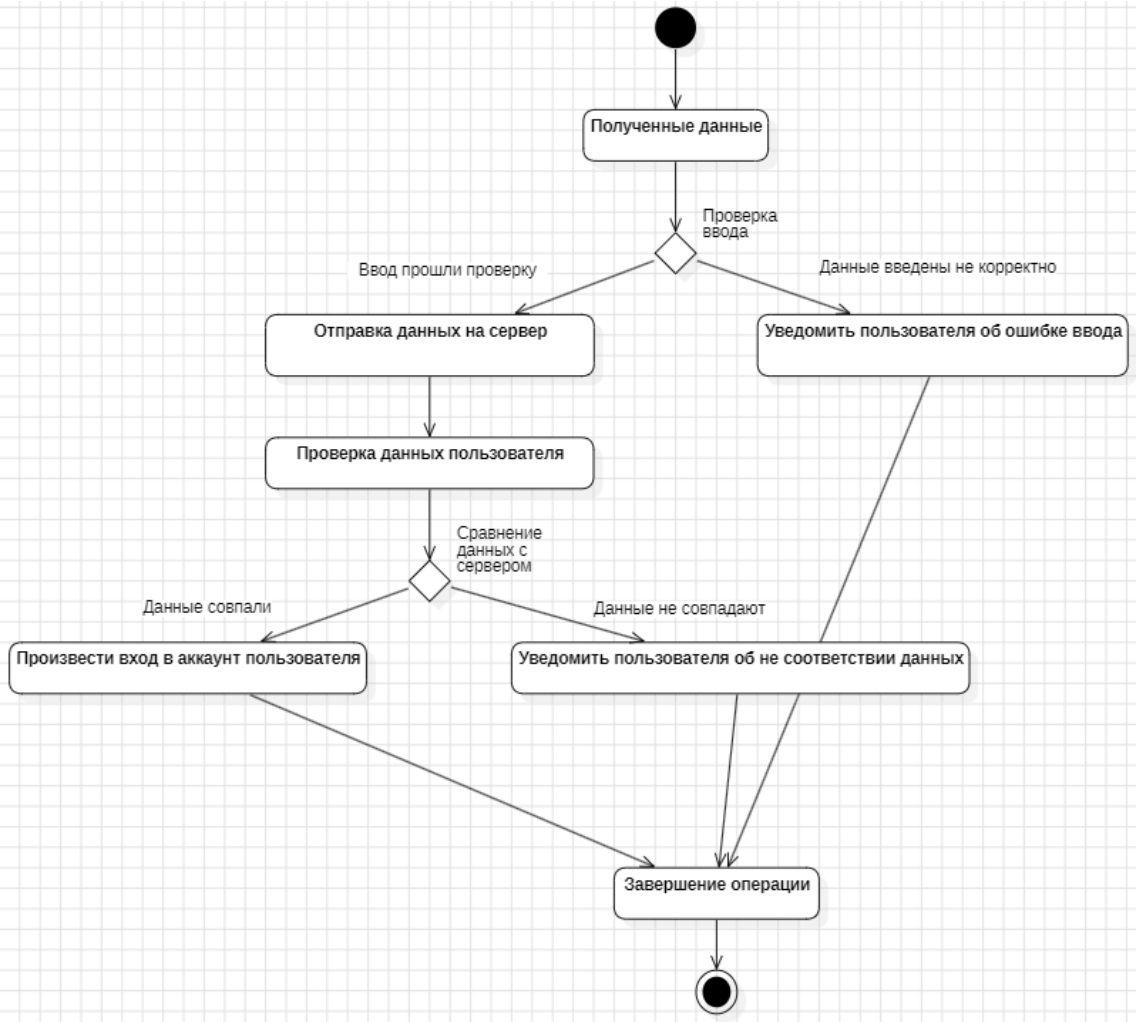

Рисунок 14 – Диаграмма состояний для варианта использования «Вход пользователя в сервис»

РЕЛЯЦИОННАЯ АЛГЕБРА

Таблица 1 — Игра

ID игры	ID студии	Название игры	Дата выхода	ID PEGI
0001	0001	Genshin Impact	28/09/2020	12
0002	0001	Honkai Impact 3	28/03/2018	12
0003	0001	Honkai: Star Rail	26/04/2023	12
0004	0002	Stray	19/07/2022	12
0005	0003	Cyberpunk 2077	10/12/2020	18

Таблица 2 — Списки пользователя

ID пользователя	ID списка	Наименование	Количество добавлений
0001	01	Играю	4
0001	02	В планах	601
0001	03	Пройдено	261
0001	04	Отложено	30
0001	05	Брошено	98

Таблица 3 — Списки пользователя

ID пользователя	ID списка	Наименование	Количество добавлений
0001	01	Играю	112
0001	02	В планах	601
0001	03	Пройдено	61
0001	04	Отложено	30
0001	05	Брошено	8

1. Операция пересечение

T4 = T2 ∩ T3

*Таблица 4 — Результат выполнения операции **Пересечение***

ID пользователя	ID списка	Наименование	Количество добавлений
0001	02	В планах	601
0001	03	Пройдено	61
0001	04	Отложено	30

2. Операция выборка

T5 = σ(**ID PEGI** > 16)T1

*Таблица 5 – Результат выполнения операции **Выборка***

ID игры	ID студии	Название игры	Дата выхода	ID PEGI
0005	0003	Cyberpunk 2077	10/12/2020	18

3. Операция естественное соединение

T6 = T2 **JOIN** T3

Таблица 6 — Список игр

ID пользователя	ID списка	Наименование	Количество добавлений
0001	01	Играю	4
0001	02	В планах	601
0001	03	Пройдено	261
0001	04	Отложено	30
0001	05	Брошено	98
0001	01	Играю	112
0001	02	В планах	601
0001	03	Пройдено	61
0001	04	Отложено	30
0001	05	Брошено	8

4. Операция проекция

T7 = **π** (***Наименование***, ***Количество добавлений***) T2

*Таблица 7 — Результат выполнения операции **Проекция***

Наименование	Количество добавлений
Играю	4
В планах	601
Пройдено	261
Отложено	30
Брошено	98

5. Операция разность

T8= T2 - T3

*Таблица 8 – Результат выполнения операции **Разность***

ID пользователя	ID списка	Наименование	Количество добавлений
0001	01	Играю	4
0001	03	Пройдено	261
0001	05	Брошено	98
0001	01	Играю	112
0001	03	Пройдено	61
0001	05	Брошено	8

6. Операция деление

T10 = T2/T9

Таблица 9

Наименование
Играю

*Таблица 10 – Результат выполнения операции **Деление***

ID пользователя	ID списка	Количество добавлений
0001	01	4

МОДЕЛЬ НОТАЦИИ ПИТЕРА-ЧЕНА

Модель нотации Питера Чена (Peter Chen notation) является расширением модели сущность-связь (entity-relationship, ER-модели) и используется для описания концептуальных схем баз данных. Модель нотации Питера Чена была разработана Питером Ченом в 1976 году и является одной из самых популярных нотаций для ER-моделирования.

Модель нотации Питера Чена включает следующие элементы:

1. Сущности (entities) - объекты или концепции, которые имеют независимое существование и могут быть идентифицированы. Сущности изображаются в виде прямоугольников с названием сущности внутри.

2. Связи (relationships) - ассоциации между сущностями. Связи изображаются в виде линий, соединяющих сущности. На концах линий могут быть указаны роли сущностей в связи.

3. Атрибуты (attributes) - характеристики сущностей или связей. Атрибуты изображаются в виде овалов, соединенных линией с сущностью или связью, которой они принадлежат.

4. Слабые сущности (weak entities) - сущности, которые не могут быть идентифицированы без ссылки на другую сущность. Слабые сущности изображаются в виде двойных прямоугольников с названием сущности внутри.

5. Идентификаторы (identifiers) - атрибуты, которые однозначно идентифицируют сущность. Идентификаторы изображаются в виде подчеркнутых атрибутов.

6. Кардинальность (cardinality) - число экземпляров сущности, участвующих в связи. Кардинальность изображается в виде цифр или диапазонов, указанных рядом с линией связи.

7. Модальность (modality) - ограничения на участие сущности в связи. Модальность изображается в виде символов, указанных рядом с линией связи.

Модель нотации Питера Чена используется для создания

концептуальных схем баз данных, которые описывают структуру данных и отношения между ними. Концептуальная схема базы данных является основой для создания логической и физической схем базы данных.

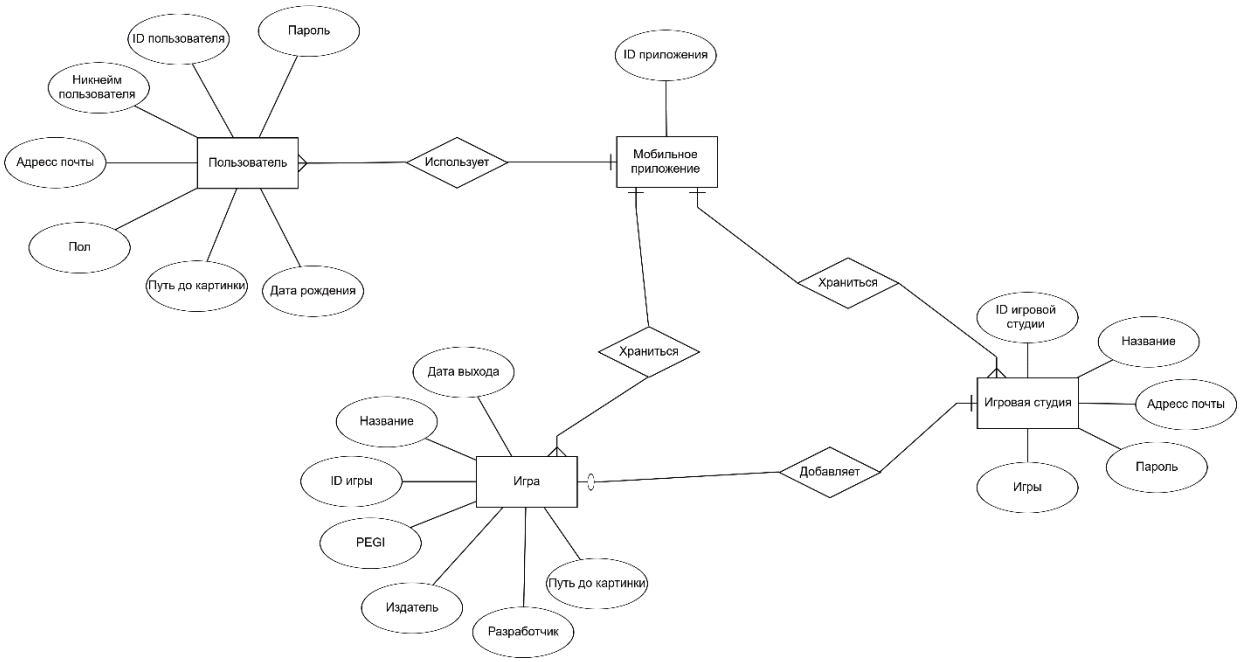

Рисунок 15 – Диаграмма нотации Питера Чена для варианта использования «Календарь игровых новинок»

ЛОГИЧЕСКАЯ МОДЕЛЬ БД

Логическая модель базы данных (ЛМ БД) представляет собой набор таблиц, которые описывают структуру и отношения между данными в базе данных. ЛМ БД создается на основе концептуальной модели, которая определяет сущности, атрибуты и отношения в области применения.

Логическая модель базы данных состоит из следующих элементов:

- Таблицы (реляции) - представляют собой набор строк (записей), каждая из которых содержит данные о конкретном экземпляре сущности.
- Столбцы (атрибуты) - представляют собой характеристики сущности, которые описывают ее свойства. Каждый столбец имеет имя и тип данных.
- Первичные ключи - представляют собой один или несколько столбцов, которые однозначно идентифицируют каждую строку в таблице.
- Внешние ключи - представляют собой столбцы, которые ссылаются на первичный ключ другой таблицы, устанавливая связь между таблицами.
- Ограничения - представляют собой правила, которые определяют допустимые значения для столбцов и таблиц.

Для варианта использования " Календарь игровых новинок " можно предложить следующую структуру таблиц (Рисунок 16):

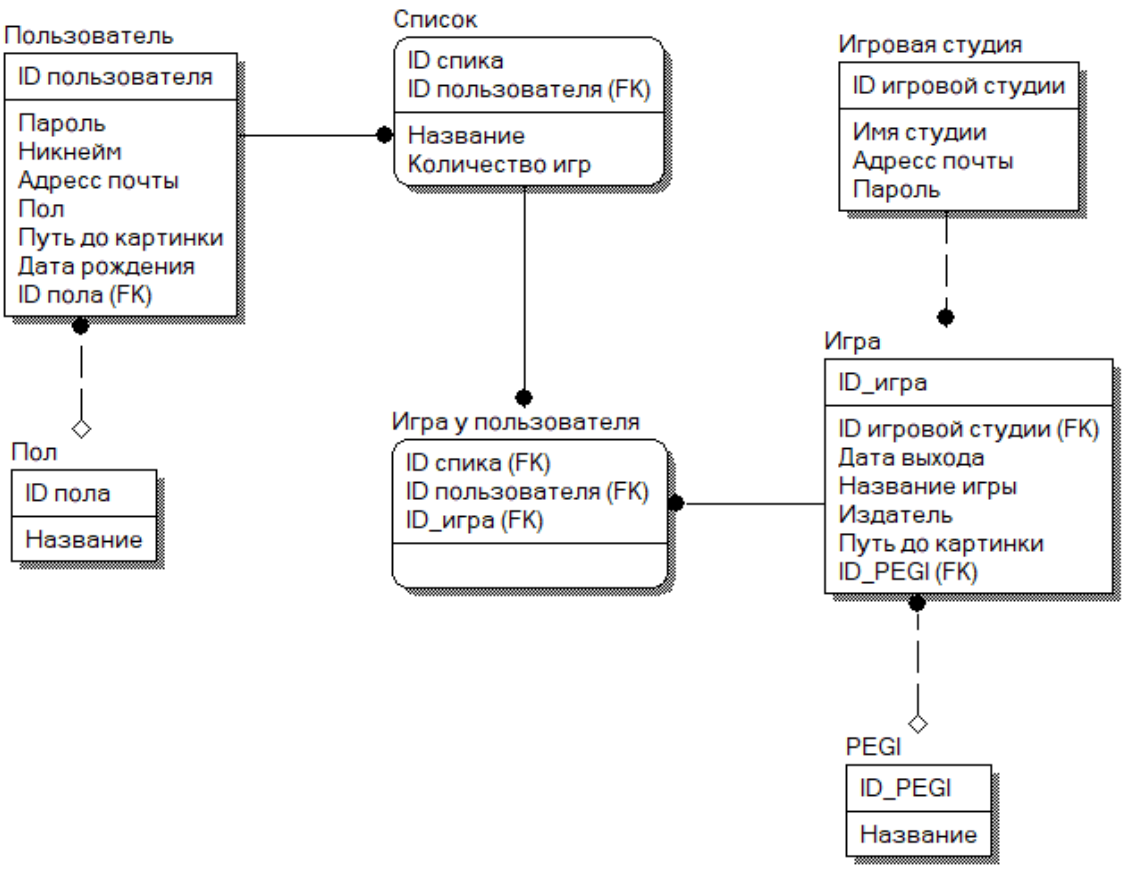

Рисунок 16 – Логическая модель базы данных для варианта использования «Календарь игровых новинок»

ФИЗИЧЕСКАЯ МОДЕЛЬ БД

Физическая модель базы данных представляет собой применение логической модели базы данных в конкретной системе управления базами данных. Она определяет структуру и организацию данных в СУБД, включая способы хранения данных в файлах, таблицах и других структурах, а также их взаимосвязь при помощи индексов, ключей и прочих средств.

Физическая модель базы данных должна учитывать технические характеристики конкретной СУБД, такие как поддерживаемые типы данных, размеры страниц, методы доступа к данным и другие. Кроме того, физическая модель должна способствовать оптимальному использованию ресурсов СУБД, таких как память, процессорное время и дисковое пространство.

Физическая модель БД представлена на Рисунке 17.

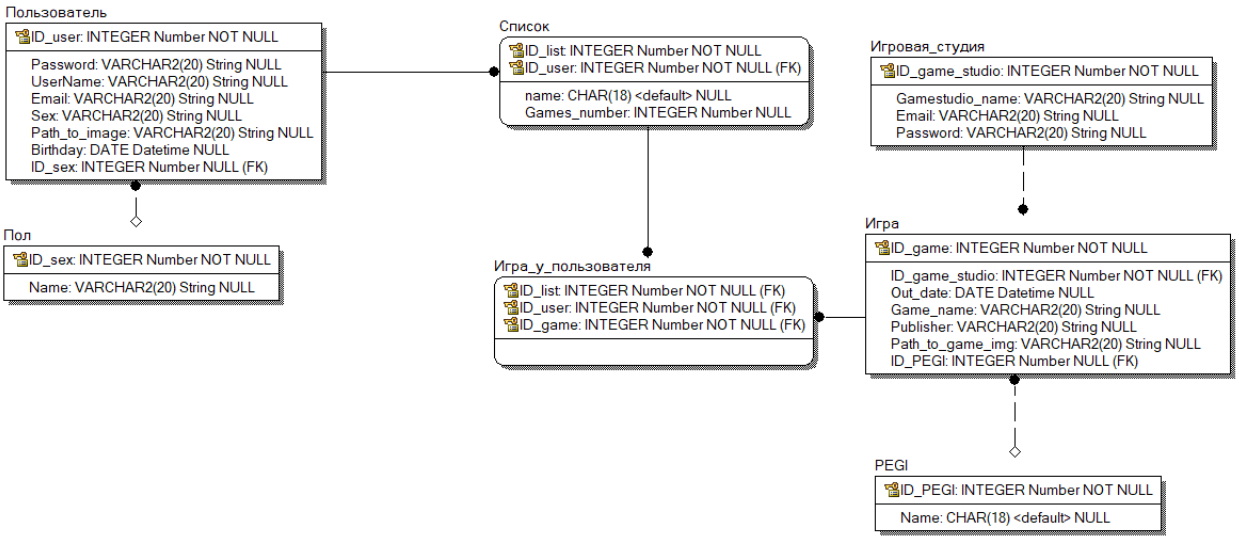

Рисунок 17 – Физическая модель базы данных для варианта использования «Календарь игровых новинок»

ВЫВОД

Было разработано программное обеспечение для создания календаря игровых новинок путем выполнения следующих этапов:

1. Изучение предметной области и выявление требований к системе.
2. Создание модели IDEF0 для объяснения процессов и взаимодействий.
3. Разработка модели DFD для описания потоков данных и их преобразований.
4. Проектирование системы с применением UML, включая разработку различных диаграмм.
5. Использование реляционной алгебры для создания запросов к базе данных.
6. Создание логической модели базы данных.
7. Создание физической модели базы данных, описывающей структуру таблиц и их взаимосвязи.

В результате выполненной работы была разработана система, которая позволяет организовать работу календаря игровых новинок. Учитывая все необходимые параметры и требования.

Выполненная работа позволила создать полноценную систему для организации работы календаря, которая может быть использована для организации такого мобильного приложения.