Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИУ «ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ»

КАФЕДРА ИУ7 «ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ЭВМ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ»

КУРСОВАЯ РАБОТА

НА ТЕМУ:

Разработка базы данных соревновательной игры

Студент	**ИУ7-61Б**		
	(группа)	(подпись, дата)	(И.О. Фамилия)
Руководитель курсового проекта			**Волкова Л.Л.**
		(подпись, дата)	(И.О. Фамилия)
Консультант			
		(подпись, дата)	(И.О. Фамилия)

2025 г.

РЕФЕРАТ

Расчетно-пояснительная записка 25 с., 8 рис., 8 источников, 1 прил.

KEYWORD1 KEYWORD2 KEYWORD3.

Объектами исследования стали TODO

Цель работы — TODO

В процессе работы составлены TODO

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Целью курсовой работы является разработка базы данных соревновательной игры.

Для достижения поставленной цели необходимо решить следующие задачи:

— формализовать соревновательную игру, описать предметную область;

— провести анализ существующих решений;

— формализовать сущности базы данных;

— спроектировать архитектуру базы данных и ограничения целостности;

— спроектировать процедуру выдачи наград;

— выбрать средства реализации базы данных и приложения;

— разработать сущности базы данных соревновательной игры и описать интерфейс доступа к базе данных;

— описать методы тестирования разработанного функционала и разработать тесты для проверки корректности работы приложения;

— исследовать зависимость времени выдачи наград от количества вознаграждаемых игроков в двух случаях: при помощи хранимой процедуры и при помощи обычных запросов.

1 Аналитическая часть

Для сравнения были выбраны соревновательные и обучающие игры. Составлены следующие критерии:

— наличие временных соревнований (а);

— внутриигровые вознаграждения за соревнования (б);

— образовательная направленность (в).

Сравнение представлено в таблице 1.1.

Таблица 1.1 — Сравнение существующих решений

	а	б	в
Клавогонки	+	-	+
Математический тренажёр	-	-	+
Clash Royale	+	+	-

1.1 Формализация соревновательной игры

Соревновательная игра состоит из основной части, оцениваемая количеством очков (целым числом), и заключительной, где очки игроков сравниваются между собой.

Соревнование характеризуется сроками проведения (начало и конец), информацией о выдаваемых наградах, а также параметрами игры. Выдаваемые награды характеризуются собственно наградой, а также критерием выдачи исходя из таблицы лидеров. Возможны следующие критерии:

— место в таблице лидеров — игрок занял место в таблице лидеров в пределе указанного диапазона;

— ранг в таблице лидеров — доля игроков (число от 0 до 1), которых игрок опередил в таблице лидеров, лежит в пределе указанного диапазона.

Параметры игры представляют собой текстовый файл.

В основной части игрок зарабатывает очки, решая математические примеры на скорость, указанные в параметрах соревнования. Результат, как и время подачи результата сохраняется в таблицу лидеров соответствующего соревнования.

Заключительная часть наступает по истечению срока соревнования. Составляется таблица лидеров, сортируя игроков сначала по убыванию игрового счёта, потом по возрастанию времени подачи последнего результата. Награды выдаются игрокам, исходя из таблицы лидеров.

В рамках поставленной цели требуется разработать базу данных, содержащую информацию о соревнованиях, об игроках, о их участии в соревнованиях, а также информацию о выданных наградах и выдаваемых наградах соревнования.

1.2 Формализация данных

Разрабатываемая база данных должна содержать информацию об игроках, профилей игроков, соревнований, результатах игры, наград. Информация об игровом уровне хранится как Сущности базы данных представлены в таблице 1.2

Таблица 1.2 — Описание сущностей базы данных

Сущность	Данные
Аккаунт	Логин, почта, права доступа, пароль
Профиль	Имя, описание, изображение
Соревнование	Имя, описания, даты начала и конца, описание уровня, выданы ли награды
Тип награды	Имя, описание, редкость, изображение, внутриигровое представление
Награда	Дата выдачи. Ссылается на тип награды, соревнование и игрока
Награда соревнования	Критерий выдачи. Ссылается на тип награды и соревнование
Критерий выдачи по месту	Минимальное и максимальное место
Критерий выдачи по рангу	Минимальный и максимальный ранг

Диаграмма сущностей базы данных представлена на рисунке 1.1

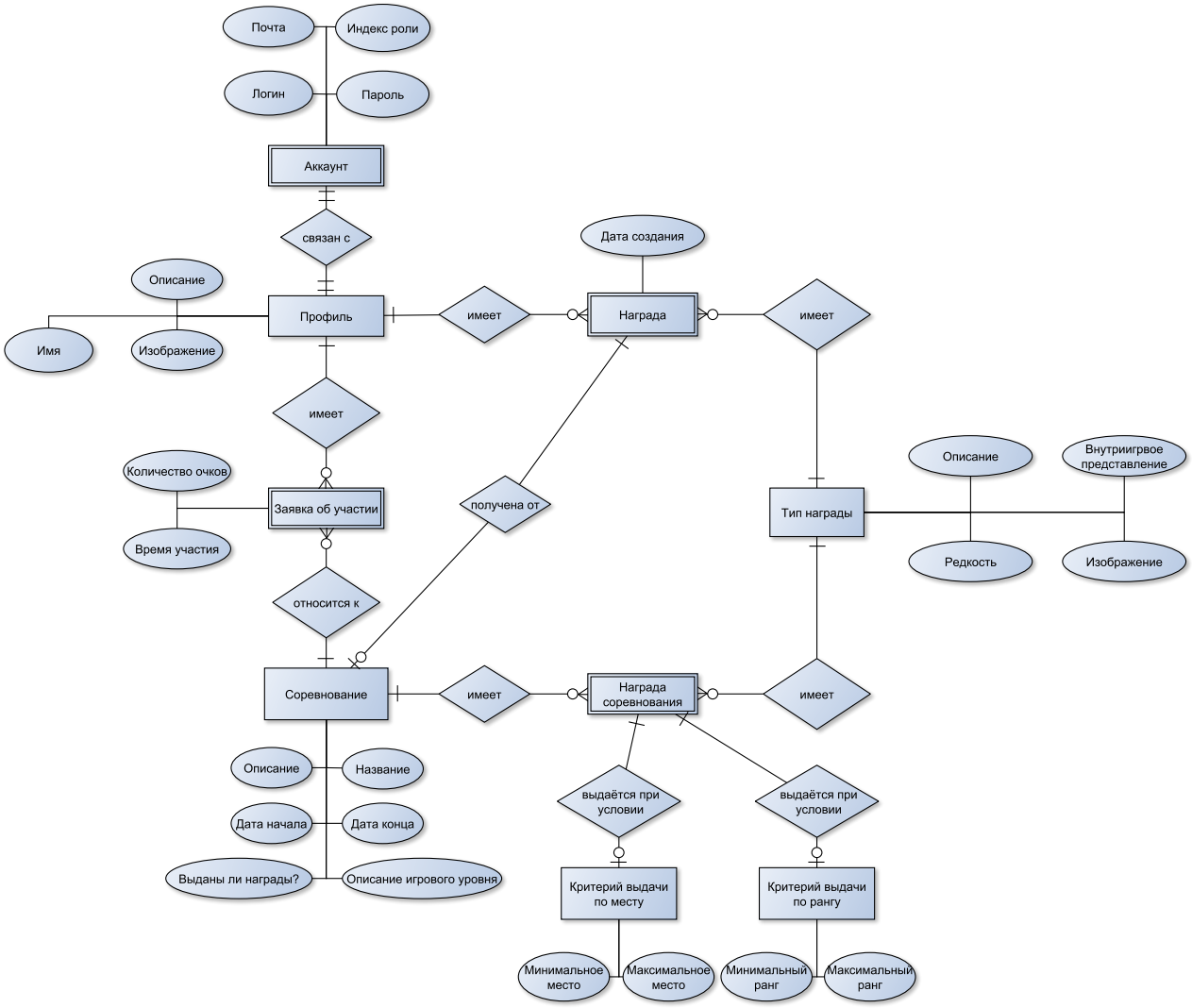

Рисунок 1.1 — Диаграмма сущностей базой данных в нотации Чена

1.3 Категории пользователя

В рамках задачи было выделено три категории пользователя:

— гость,

— игрок,

— администратор.

Все категории пользователя имеют возможность просматривать профили игроков, таблицы лидеров соревнования, а также собственно соревнований (данные о сроках проведения и наградах)

Гость имеет возможность авторизации и создания аккаунта. При авторизации гость может стать игроком или администратором.

Игрок может просматривать свои награды, редактировать свой профиль, а также участвовать в соревнованиях.

Администратор может:

— назначать и отзывать награды игрока;

— создавать и редактировать соревнования;

— отзывать нежелательные результаты, удаляя их;

— создавать и редактировать типы наград.

Игрок и администратор имеют возможность выйти из аккаунта, в последствии чего устанавливается категория пользователя "гость".

Диаграмма пользования базой данных представлена на рисунке 1.2.

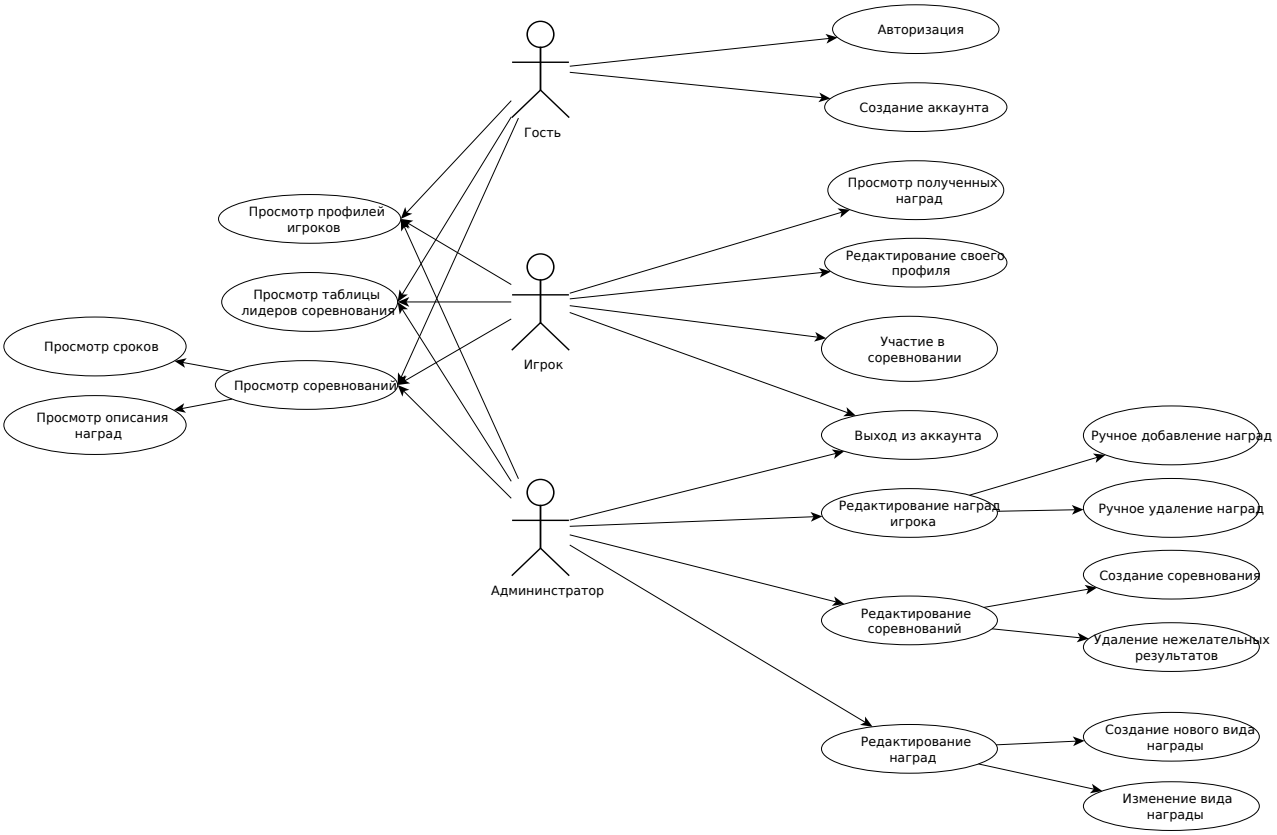

Рисунок 1.2 — Диаграмма пользования базой данных

1.4 Выбор модели данных

Вывод

TODO

2 Конструкторская часть

2.1 Описание сущностей базы данных

Используются следующие сокращения для обозначения ограничений на поля сущности:

— PK — первичный ключ;

— FK — внешний ключ;

— U — значение уникально в рамках таблицы;

— NN — пустое значение поля недопустимо.

Таблица 2.1 — Описание таблицы account

Поле	Тип	Описание	Ограничения
ID	int	Идентификатор	PK
login	varchar(32)	Логин игрока	U
password_hash	varchar	Хэш пароля	
email	varchar	Почта	
privilegy_level	int	Категория пользователя	
profile_image	bytea	Изображение профиля	
description	varchar	Описание профиля	

Дополнительные ограничения к таблице 2.1: login не содержит пробелов.

Таблица 2.2 — Описание таблицы competition

Поле	Тип	Описание	Ограничения
ID	int	Идентификатор	PK
competition_name	varchar(64)	Название соревнования	NN
description	varchar(128)	Описание соревнования	
start_time	timestamp	Время начала соревнования	NN
end_time	timestamp	Время конца соревнования	NN
level_data	bytea	Данные об уровне соревнования	
has_ended	bool	Закончилось ли соревнование? (выданы ли награды)	NN

Дополнительные ограничения к таблице 2.2: время конца всегда больше времени начала.

Таблица 2.3 — Описание таблицы player_participation

Поле	Тип	Описание	Ограничения
competition_ID	int	Идентификатор соревнования	FK, NN
account_ID	int	Идентификатор игрока	FK, NN
score	int	Очки	NN
last_update_time	timestamp	Последнее время обновления результата	

Дополнительные ограничения к таблице 2.3: пара значений competition_ID и account_ID уникально в рамках таблицы.

Таблица 2.4 — Описание таблицы reward_description

Поле	Тип	Описание	Ограничения
ID	int	Идентификатор	PK
reward_name	varchar(64)	Название награды	NN
description	varchar(128)	Описание награды	
icon_image	bytea	Изображение награды	
ingame_data	bytea	Представление награды в игре	

Тип condition_type_enum является перечисляемым типом, возможными значениями которого являются 'rank' и 'place'.

Таблица 2.5 — Описание таблицы competition_reward

Поле	Тип	Описание	Ограничения
ID	int	Идентификатор	PK
reward_description_id	int	Идентификатор описания награды	FK, NN
competition_id	int	Идентификатор соревнования	FK, NN
condition_type	condition_type_enum	Тип награды	NN
min_place	int	Минимальное место	
max_place	int	Максимальное место	
min_rank	decimal(4,3)	Минимальный ранг	
max_rank	decimal(4,3)	Максимальный ранг	

Дополнительные ограничения к таблице 2.5:

— если condition_type = rank, то соблюдается соотношение $0 \leq min\_rank \leq max\_rank \leq 1$ и соответствующим полям присвоено значение.

— если condition_type = place, то соблюдается соотношение $1 \leq min\_place \leq max\_place$ и соответствующим полям присвоено значение.

Таблица 2.6 — Описание таблицы player_reward

Поле	Тип	Описание	Ограничения
ID	int	Идентификатор	PK
reward_description_id	int	Идентификатор описания награды	FK, NN
player_id	int	Идентификатор игрока	FK, NN
competition_id	int	Идентификатор соревнования	FK
creation_date	timestamp	Дата создания награды	NN

На рисунке 2.1 представлена схема базы данных.

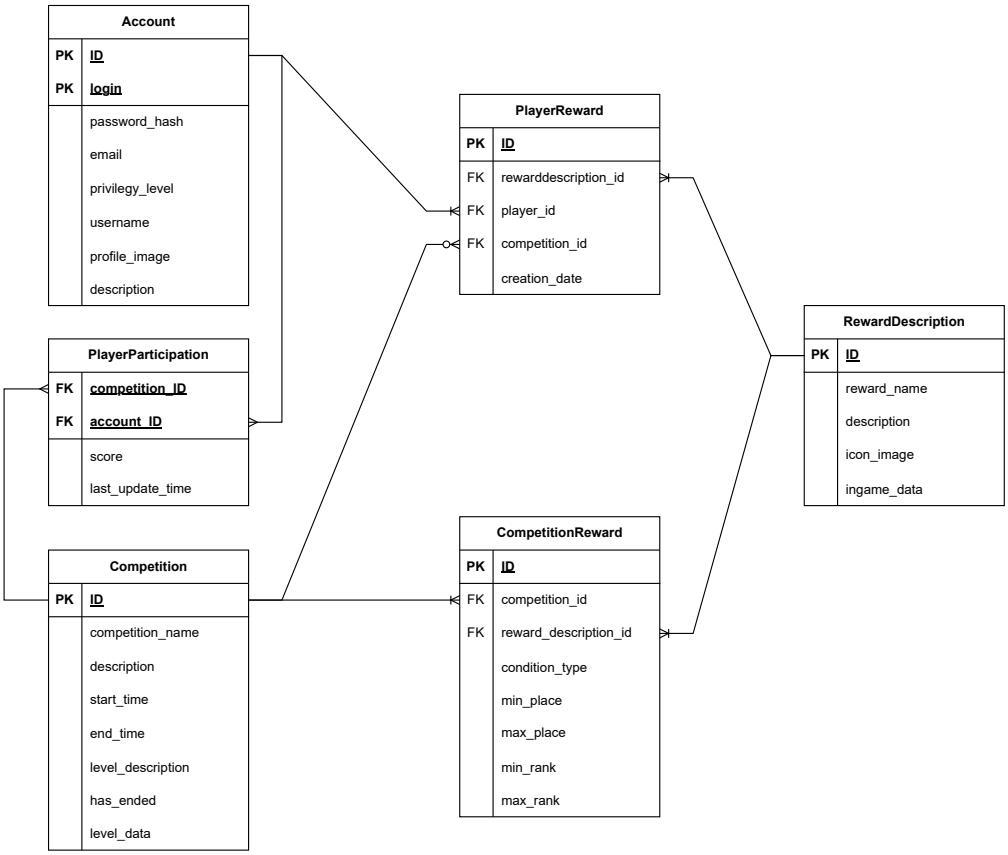

Рисунок 2.1 — Схема проектируемой базы данных

2.2 Функциональная модель

При истечении срока соревнования, игрокам должны быть выданы награды, исходя из таблицы лидеров. Награды соревнования выдаются всем участникам в таблице лидеров, которые подошли под критерий выдачи награды. После этого, для соревнования устанавливается, что награды за него выданы, чтобы избежать повторных вознаграждений.

Схема хранимой процедуры выдачи наград представлена на рисунках 2.2, 2.3.

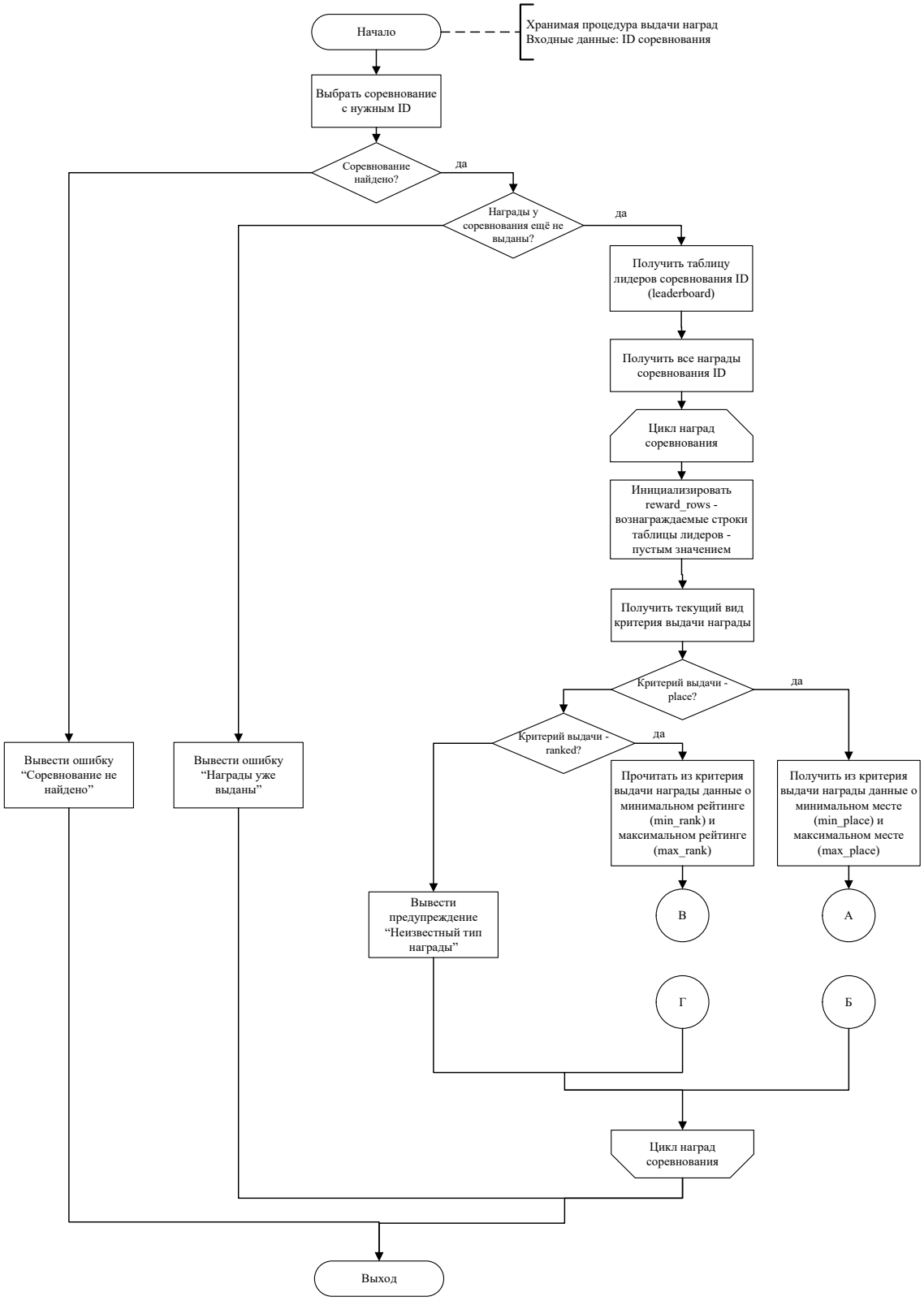

Рисунок 2.2 — Хранимая процедура выдачи наград

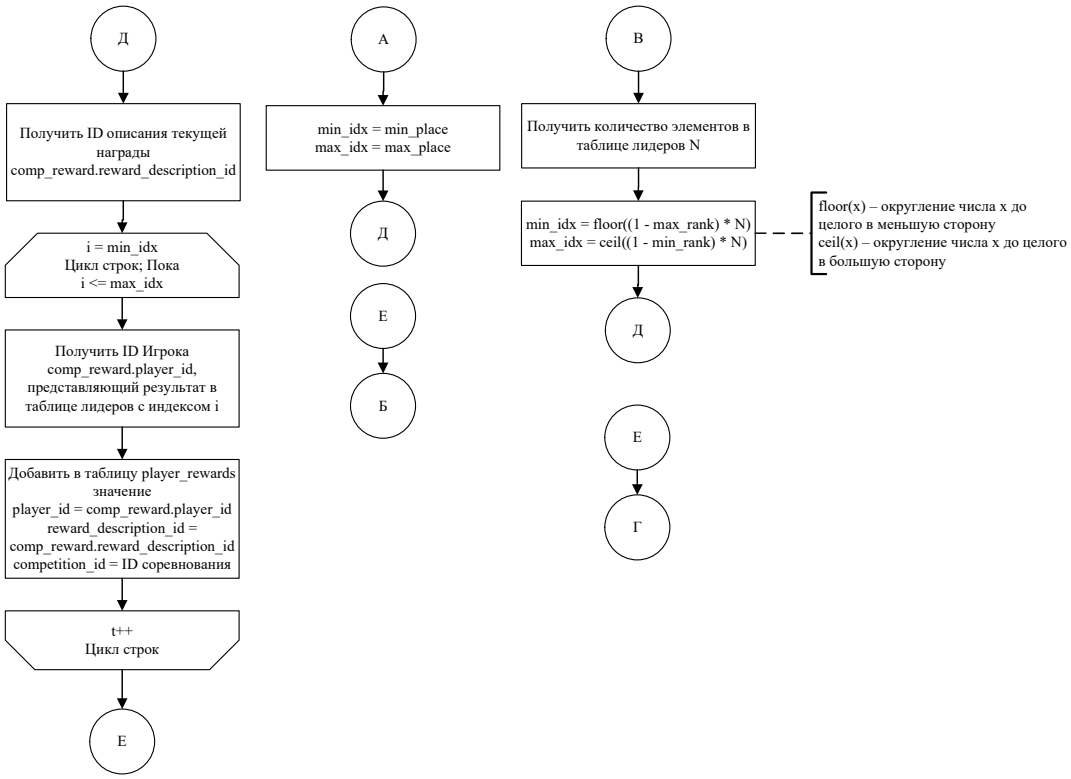

Рисунок 2.3 — Хранимая процедура выдачи наград

Для обеспечения безопасности, значение полей "хэш пароля"у таблицы аккаунтов должно быть недоступно. Для этого нужно определить образ таблицы, где данное поле заменено пустыми значениями. Помимо этого, следует определить функцию на уровне базы данных, которая сравнивает хэши паролей, чтобы предоставить возможность определять правильность пароля. Схема представлена на рисунке 2.4

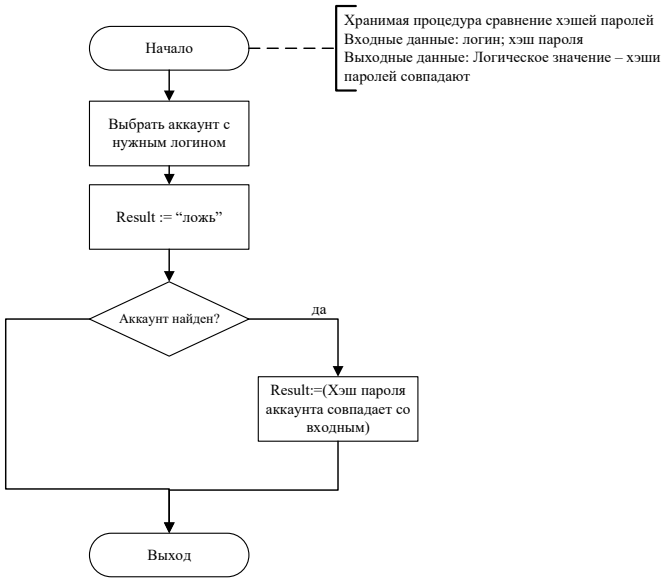

Рисунок 2.4 — Хранимая процедура сравнения хэшей паролей

2.3 Ролевая модель

В рамках задачи было выделено четыре роли на уровне базы данных:

— гость,

— игрок,

— администратор,

— демон выдачи наград.

Гость имеет следующие возможности:

— читать данные из таблиц account_readonly, competition, competition_reward, player_participation, reward_description;

— добавлять новые данные в таблицу account (создавать новый аккаунт);

— вызвать хранимую процедуру check_password_hash.

Игрок имеет все возможности, что имеет гость. Дополнительно, он имеет следующие возможности:

— обновлять таблицу account по полям username, description, profile_image;

— добавлять данные и обновлять таблицу player_participation;

— читать таблицу player_reward

Админ имеет все возможности, что имеет гость. Дополнительно, он имеет следующие возможности:

— удалять данные из player_participation;

— добавлять, изменять, удалять данные из player_reward;

— добавлять, изменять данные из таблиц competition, competition_reward, reward_description.

Демон выдачи наград имеет следующие возможности:

— вызвать хранимую процедуру grant_rewards();

— читать и изменять данные из таблицы competition;

— читать данные из таблиц player_participation, competition_reward;

— добавлять данные в таблицу player_reward.

Вывод

Спроектирована архитектура базы данных и ограничения целостности. Спроектированы хранимые процедуры выдачи наград и проверки хэша пароля.

3 Технологическая часть

В качестве СУБД была выбрана PostgreSQL [1], поскольку в ней представлены все необходимые возможности, в частности создание хранимых процедур.

Для реализации приложения был выбран язык C# версии 11.0 вместе с фреймворком ASP.NET Core, поскольку в нём представлены все необходимые возможности для создания программного интерфейса взаимодействия с базой данных.

Для интерфейса доступа к СУБД был выбран Entity Framework Core, поскольку он предоставляет возможность взаимодействия с объектами базы данных как с классами C#.

Для обработки изображений была выбрана библиотека Magick.NET [2].

Для хранения и выполнения отложенных задач использовалась библиотека Quartz.NET [3].

3.1 Компоненты программного обеспечения

На рисунке 3.1 представлена диаграмма компонентов приложения.

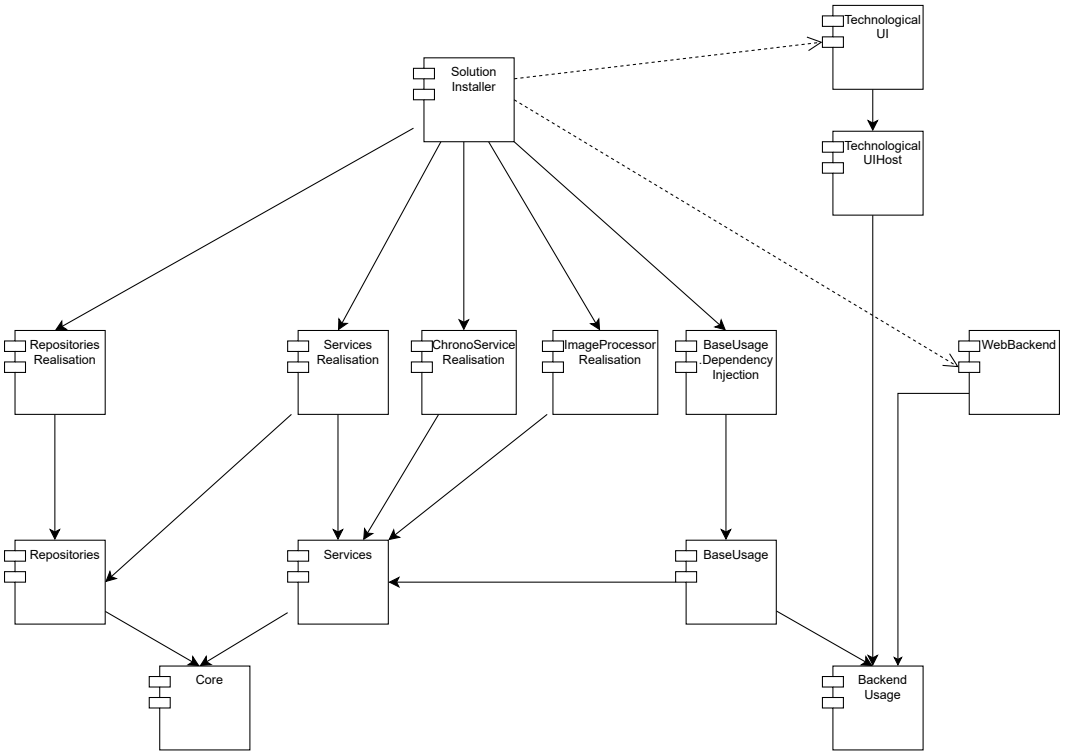

Рисунок 3.1 — Диаграмма компонентов приложения

Модуль Core содержит описание базовых объектов, с которыми работает приложение.

Модуль Repositories содержит описание уровня взаимодействия с базой данных.

Модуль RepositoriesRealisation содержит реализацию уровня взаимодействия с базой данных.

Модуль Services содержит описание уровня бизнес-логики. В нём также содержатся описание сервиса постановки задач с отложенным ожиданием и сервиса обработки изображений.

Модуль ChronoServiceRealisation содержит реализацию сервиса постановки задач с отложенным ожиданием.

Модуль ImageProcessorRealisation содержит реализацию сервиса обработки изображений.

Модуль ServciesRealisation содержит реализацию уровня бизнес-логики.

Модуль BackendUsage содержит описание взаимодействия с базой данных.

Модуль BaseUsage содержит реализацию взаимодействия с базой данных.

3.2 Тестирование

Для тестирования приложения были предложены следующие виды тестов:

— интеграционные тесты для компонентов реализации взаимодействия с базой данных (74 теста, покрытие кода — 78.4%);

— интеграционные тесты для реализации компонента отложенных задач (6 тестов, покрытие кода — 84.4%);

— интеграционные тесты для реализации компонента обработки изображений (7 тестов, покрытие кода — 87.3%);

— модульные тесты для реализации компонента бизнес – логики (76 тестов, покрытие кода — 77.8%).

Общее покрытие кода всеми тестами — 56.53% Все тесты пройдены успешно.

Вывод

TODO

4 Исследовательская часть

Цель исследования - сравнить зависимость времени выдачи наград от количества вознаграждаемых игроков в двух случаях: при помощи хранимой процедуры и при помощи обычных запросов.

Исследование выполнялись на вычислительной машине со следующими характеристиками:

— процессор: Intel(R) Core(TM) i7-7700K 4.20 ГГц 4.20 ГГц. [4];

— количество ядер процессора: 4, количество потоков: 8;

— объём оперативной памяти: 32 ГБ.

Описание способов выдачи наград:

— на стороне базы данных — вызов хранимой процедуры;

— на стороне приложения — реализация логики добавления наград с помощью Entity Framework Core.

Для создания различных входных данных, создаются независимые варианты базы данных, инициализируемые различными параметрами. Для создания независимых образов PostgreSQL использовалась библиотека Testcontainers.Postgresql [5]

Исследование проводилось в два этапа: подготовка данных и замеры времени.

На каждую подготовку данных выделялся отдельный образ PostgreSQL, которым был инициализирован сценарием (см. приложение А). Программа заполняла СУБД нужными данными, после чего делала снимок базы данных при помощи pg_dump [6] в файл. Для заполнения базы данных случайными данными использовалась библиотека Bogus [7].

Если снимок данных подготавливался для N выдаваемых наград, генерировалось:

— 1 соревнование;

— $N/2$ участников и записей в таблицы лидеров;

— 5 критериев наград, каждый из которых вознаграждает $\frac{N}{5}$ участников.

Критерии выдачи награды генерировались с равной вероятностью двух возможных видов: по диапазону доли в таблице лидеров, по диапазону мест. Параметр $N \in \{250, 500, 1000, 2000, 3000, 4000, 5000\}$

На каждый замер времени выделялся отдельный образ PostgreSQL, который был инициализирован снимком базы данных, полученным на этапе подготовке данных. После этого, выполнялся один замер времени выдачи наград на стороне БД/приложения. Для измерения реального времени работы использовался класс Stopwatch [8].

Для каждого параметра N подготавливалось 10 вариантов входных данных. Для каждого варианта входных данных было произведено 30 замеров. Итоговое время работы для одного параметра N является усреднённым временем по 300 значениям.

В таблице 4.1 приведены результаты исследования.

Таблица 4.1 — Время выполнения от количества дополнительных потоков

Кол-во наград	Время, мс	
	На стороне базы данных	На стороне приложения
250	26.2485	43.8758
500	35.4606	64.9545
1000	45.4333	107.0000
2000	67.2545	171.0727
3000	90.6091	242.9091
4000	116.9364	342.6629
5000	128.8848	365.5333

При помощи метода наименьших квадратов были найдены коэффициенты полинома 3 степени, аппроксимирующие данные точки. Время на стороне базы данных аппроксимирует функция формулы (4.1); время на стороне приложения — формула (4.2)

$$t_1(N) = -5.774 \cdot 10^{-10} \cdot N^3 + 3.743 \cdot 10^{-6} \cdot N^2 + 1.684 \cdot 10^{-2} \cdot N + 24.066 \tag{4.1}$$

$$t_2(N) = -2.76 \cdot 10^{-9} \cdot N^3 + 1.888 \cdot 10^{-5} \cdot N^2 + 4.087 \cdot 10^{-2} \cdot N + 38.777 \tag{4.2}$$

Исходя из того, что коэффициенты при N^3 и N^2 достаточно малы ($< 10^{-4}$), зависимость времени выдачи наград от количества выдаваемых наград близка к линейной зависимости.

По результатам исследования, время работы на стороне сервера меньше, чем время работы на стороне приложения. При этом, линейный коэффициент роста времени на стороне приложения в 2.427 раза больше, чем на стороне сервера.

По данным таблицы 4.1 построен график, представленный на рисунке 4.1.

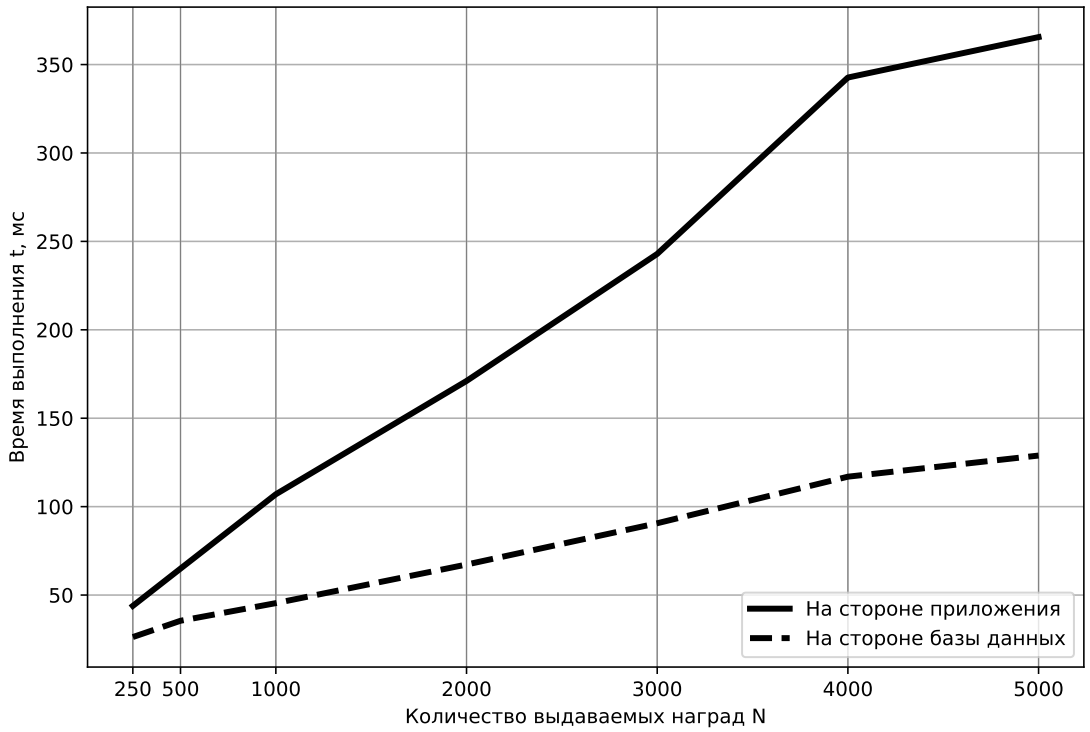

Рисунок 4.1 — Зависимость времени выдачи наград от количества выдаваемых наград

Вывод

Проведено исследование времени выдачи наград от количества вознаграждаемых игроков в двух случаях: при помощи хранимой процедуры и при помощи обычных запросов. Время работы на стороне сервера меньше, чем время работы на стороне приложения. Зависимость времени от количества наград в обоих случаях близка к линейной, при этом линейный коэффициент роста времени на стороне приложения в 2.427 раза больше, чем на стороне сервера. Следовательно, использование хранимой процедуры для вознаграждения пользователей целесообразно для увеличения скорости работы приложения.

ЗАКЛЮЧЕНИЕ

Были выполнены следующие задачи:

— DONE1

— DONE2

Исследование DONE

Цель DONE

Поставленные задачи выполнены. Цель работы достигнута.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Документация PostgreSQL [Электронный ресурс]. — Режим доступа: https://www.postgresql.org/docs/ (дата обращения: 05.05.2025).
2. Документация библиотеки Magick.NET [Электронный ресурс]. — Режим доступа: https://github.com/dlemstra/Magick.NET/tree/main/docs (дата обращения: 06.05.2025).
3. Документация библиотеки Quartz.NET [Электронный ресурс]. — Режим доступа: https://www.quartz-scheduler.net/documentation/ (дата обращения: 05.05.2025).
4. Спецификация процессора Intel Core i7 [Электронный ресурс]. — Режим доступа: https://www.intel.com/content/www/us/en/products/sku/97129/intel-core-i77700k-processor-8m-cache-up-to-4-50-ghz/specifications.html (дата обращения: 15.10.2024).
5. Документация Testcontainers.Postgresql [Электронный ресурс]. — Режим доступа: https://dotnet.testcontainers.org/modules/postgres/ (дата обращения: 05.05.2025).
6. Документация команды pgdump в PostgreSQL [Электронный ресурс]. — Режим доступа: https://www.postgresql.org/docs/current/app-pgdump.html (дата обращения: 05.05.2025).
7. Описание библиотеки Bogus [Электронный ресурс]. — Режим доступа: https://www.nuget.org/packages/bogus (дата обращения: 05.05.2025).
8. Документация класса System.Diagnostics.Stopwatch [Электронный ресурс]. — Режим доступа: https://learn.microsoft.com/ru-ru/dotnet/api/system.diagnostics.stopwatch?view=net-8.0 (дата обращения: 05.05.2025).

ПРИЛОЖЕНИЕ А

Презентация состоит из 16 слайдов