

Министерство образования Республики Беларусь
Учреждение образования “Белорусский государственный университет
информатики и радиоэлектроники”

Факультет компьютерных систем и сетей
Кафедра информатики
Дисциплина “Основы алгоритмизации и программирования”

К ЗАЩИТЕ ДОПУСТИТЬ
Руководитель курсового проекта
ассистент кафедры информатики
_____ В.Д. Иванович
____.____.2025

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к курсовому проекту
на тему:
**“Разработка десктопного приложения для интеллектуальной игры на
платформе Windows с использованием технологии .Net”**

БГУИР КП 6-05 0612 02 113 ПЗ

Выполнил студент группы 453503
ХАЛАМОВ Николай Андреевич

(подпись студента)

Курсовой проект представлен на
проверку _____.____.2025

(подпись студента)

Минск 2025

СОДЕРЖАНИЕ

Введение	5
1 Анализ предметной области	8
1.1 Исследование понятия и история интеллектуальных игр	8
1.2 Инструменты и приложения для создания игр	11
1.3 Анализ существующих разработок компьютерных интеллектуальных игр	13
1.4 Постановка задачи	14
2 Архитектура игрового приложения	15
2.1 Структура игры и её основные компоненты	15
2.2 Игровой процесс: основные механики и логика	17
2.3 Поддержка различных режимов игры	19
3 Программная реализация игры	20
3.1 Анализ используемых программных средств, инструментов для разработки	20
3.2 Описание реализации основного функционала игры	22
4 Тестирование игрового приложения	23
4.1 Основные этапы	23
4.1.1 Отображение меню LoginWindow	24
4.1.2 Тестирование MainMenuWindow	24
4.1.3 Тестирование SinglePlayerPackSelectionWindow	25
4.1.4 Тестирование RankingWindow	26
4.1.5 Тестирование AchievementsWindow	26
4.1.6 Тестирование BotDifficultySelectionWindow	27
4.1.7 Тестирование QuestionEditorWindow	27
4.1.8 Тестирование QuestionEditWindowRound1	28
4.1.9 Тестирование QuestionEditWindowRound234	28
4.1.10 Тестирование QuestionSelectionWindow	29
4.1.11 Тестирование ThemeCreationWindow	29
4.1.12 Тестирование QuestionCreationWindow	30
4.1.13 Тестирование MainWindow	30
4.1.14 Тестирование FriendGameWindow	33
4.1.15 Тестирование BotGameWindow	33
4.1.16 Тестирование общей работы проекта и связи всех режимов	33
4.2 Выявленные проблемы и способы их решения	33
Заключение	35
Список использованных источников	37
Приложение А (обязательное) Листинг программного кода	38
Приложение Б (обязательное) Функциональная схема программы	41
Приложение В (обязательное) Блок-схема алгоритма нечеткого поиска	42

ВВЕДЕНИЕ

В современном мире интеллектуальные игры приобретают всё большую популярность, становясь неотъемлемой частью досуга для миллионов людей по всему миру. Они предоставляют уникальную возможность не только увлекательно провести время, но и активно развивать когнитивные способности, такие как память, внимание, логическое и стратегическое мышление, а также эрудицию. Это особенно актуально в эпоху цифровых технологий, когда доступ к интеллектуальным развлечениям стал проще, чем когда-либо, а их интеграция в повседневную жизнь позволяет гармонично сочетать умственное развитие с комфортом и интерактивностью. Такие игры способствуют не только личностному росту, но и укреплению социальных связей, поскольку они часто объединяют людей в соревновательной или кооперативной среде.

Одними из наиболее популярных форм интеллектуальных игр являются викторины, а также форматы, заимствованные из телевизионных шоу и прочно вошедшие в культурный контекст, такие как “Своя игра”(Jeopardy), “Что? Где? Когда?”и их многочисленные адаптации. Эти форматы отличаются уникальным сочетанием элементов, требующих от участников не только глубоких знаний, но и быстрой реакции, умения анализировать информацию, применять дедуктивное мышление и принимать стратегические решения под давлением времени. Участие в подобных играх позволяет игрокам не только продемонстрировать свою эрудицию, но и развивать навыки, которые находят применение в профессиональной и личной жизни, такие как критическое мышление, умение работать в команде и способность быстро адаптироваться к новым условиям. Кроме того, интеллектуальные соревнования часто сопровождаются рейтинговыми системами, турнирами с денежными или призовыми наградами, а также возможностью зарабатывать очки достижений, что значительно повышает мотивацию участников и делает игровой процесс более захватывающим и динамичным.

В этой связи разработка современного, функционального и удобного десктопного приложения для Windows, предназначенного для проведения интеллектуальных игр, представляет собой актуальную и востребованную задачу. Такое приложение должно быть универсальным и поддерживать разнообразные игровые режимы, включая одиночную игру для индивидуальной тренировки, парный режим для совместного досуга с друзьями, соревнование с искусственным интеллектом для отработки стратегий. Это делает приложение подходящим для широкого круга пользователей: от любителей, желающих провести время с пользой, до профессиональных игроков, стремящихся к совершенствованию своих навыков и участию в рейтинговых матчах. Кроме того, приложение может стать платформой для образовательных инициатив, таких как школьные или университетские викторины, а также корпоративные мероприятия, направленные на развитие командного духа и интеллектуальных способностей.

На сегодняшний день рынок программного обеспечения для интеллектуальных игр остаётся относительно ограниченным, особенно в сегменте десктопных приложений с акцентом на гибкость, высокую функциональность и интуитивно понятный интерфейс. Существующие решения часто страдают от недостаточной проработки пользовательского опыта, ограниченного набора функций или устаревшего дизайна, что создаёт барьеры для широкой аудитории. Предлагаемое приложение стремится восполнить этот пробел, предоставляя пользователям продуманную платформу с возможностью самостоятельного создания, редактирования и обмена пакетами вопросов, а также гибкими настройками игрового процесса. Это позволяет игрокам адаптировать игру под свои предпочтения, будь то тренировка на специфические темы, подготовка к турнирам или создание уникального контента для друзей и сообществ.

Приложение ориентировано на создание современного, эстетически привлекательного и интуитивно понятного интерфейса, который минимизирует отвлекающие факторы и сосредотачивает внимание пользователя на игровом процессе. Визуальный стиль приложения сочетает минимализм и функциональность, обеспечивая комфортное взаимодействие как для новичков, так и для опытных игроков. Пользователи получают доступ к системе достижений, которая мотивирует к дальнейшему прогрессу, глобальной таблице рейтингов для сравнения результатов с другими игроками, настраиваемому музыкальному сопровождению для создания подходящей атмосферы, а также персонализированным настройкам, позволяющим адаптировать игру под индивидуальные предпочтения. Эти элементы делают игровой процесс более вовлекающим, эмоционально насыщенным и разнообразным, способствуя долгосрочному удержанию пользователей.

Целью данного курсового проекта является разработка десктопного приложения для интеллектуальной игры, которое сочетает в себе удобные и интуитивно понятные средства взаимодействия, мощный редактор пользовательских пакетов вопросов, разнообразные игровые режимы, а также интегрированные системы рейтинга, достижений и современный визуальный стиль, отвечающий ожиданиям широкой аудитории.

Для достижения цели проекта поставлены следующие задачи:

- 1 Проведение анализа существующих интеллектуальных игр и приложений, включая их функциональные возможности, пользовательский интерфейс и игровые механики, с целью выявления сильных и слабых сторон, а также формирования ключевых требований к собственной разработке, учитывающих современные стандарты и ожидания пользователей.

- 2 Проектирование архитектуры приложения, включающей модульную структуру для обеспечения масштабируемости и удобства поддержки, разработку моделей данных для эффективного хранения и обработки информации, проектирование интуитивного взаимодействия с пользователем

- 3 Разработка логики игровых механик, включая алгоритмы выбора и отображения вопросов, обработку пользовательских ответов с учётом

различных форматов (множественный выбор, текстовый ввод), точный подсчёт очков с учётом правил каждого режима, а также определение победителя на основе прозрачных и справедливых критериев.

4 Создание удобного и адаптивного пользовательского интерфейса, обеспечивающего комфортное взаимодействие на устройствах с различными разрешениями экрана, а также визуального редактора вопросов с интуитивно понятными инструментами для создания, редактирования.

5 Реализация дополнительных элементов вовлечённости, включая систему достижений с разнообразными целями для мотивации игроков, глобальную рейтинговую таблицу с возможностью фильтрации и поиска, настраиваемое музыкальное сопровождение с поддержкой пользовательских треков, а также гибкие настройки игрового процесса для персонализации опыта.

6 Проведение комплексного тестирования, отладки и оптимизации приложения, включая модульное и интеграционное тестирование для выявления ошибок, сбор отзывов от тестовых пользователей для улучшения интерфейса и механик, устранение узких мест в производительности, а также доработку пользовательского опыта на основе полученных данных для достижения максимального удобства и стабильности.

Пояснительная записка оформлена в соответствии с СТП 01-2024 [1].

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 Исследование понятия и история интеллектуальных игр

Интеллектуальные и творческие игры представляют собой одну из наиболее востребованных форм организации досуга, особенно в образовательной и культурной среде. Широкую популярность они приобрели благодаря телевизионным форматам, завоевавшим внимание аудитории всех возрастов. На сегодняшний день подобные игры активно используются в школах, библиотеках, культурных учреждениях, а также в клубах по работе с молодежью. Можно с уверенностью утверждать, что практически каждое общественное объединение в той или иной форме использует интеллектуальные и творческие игры как средство развития мышления и организации досуга своих участников.

Интеллектуальная игра представляет собой индивидуальное или коллективное выполнение заданий, требующих применения логического и продуктивного мышления в условиях ограниченного времени и соревновательной среды. Эти игры органично сочетают в себе элементы как учебной, так и игровой деятельности, развивая абстрактное и теоретическое мышление, формулирование понятий, а также выполнение мыслительных операций — классификации, анализа, синтеза и других.

Следует отметить, что в контексте интеллектуальных игр результат (победа в соревновании) часто отходит на второй план, уступая место самой деятельности — процессу поиска, формулировки и обоснования решения. Тем самым повышается образовательный и развивающий потенциал таких игр.

По типологии интеллектуальные игры делятся на элементарные и составные. Элементарные игры классифицируются, в том числе, по числу вариантов ответов, из которых участникам предлагается выбрать верный. Такие игры могут проводиться как в индивидуальном, так и в командном формате. Наиболее простой формой интеллектуальной игры являются тестовые задания, содержащие утверждения и ограниченное количество вариантов ответа — от двух (например, формат “Верить — не верить”) до пяти (“Эрудит-лото”). Подобные игры используются как вводный этап, средство активизации аудитории или интерактивное сопровождение основного игрового мероприятия. Их отличительная черта — высокая роль элемента случайности, что позволяет участвовать и добиваться успеха даже игрокам с невысоким уровнем подготовки. В то же время они могут служить эффективным инструментом развития логического мышления при наличии скрытого алгоритма или необходимости парадоксального мышления.

Организация интеллектуально-познавательных игр включает в себя несколько ключевых компонентов:

- 1 Название игры, которое должно отражать как соревновательный, так и познавательный характер мероприятия (например: “Турнир эрудитов”, “Интеллектуальный бой”).

2 Корректность вопросов, используемых в игре. Вопросы должны быть чётко сформулированы, логически обоснованы и лишены двусмысленности. От качества вопросов напрямую зависит степень вовлечённости участников и общее восприятие мероприятия.

3 Правила игры, обеспечивающие единообразие и прозрачность условий для всех участников. Несмотря на вариативность игровых форматов, большинство правил сводятся к ответам на типовые организационные вопросы.

4 Игровой сюжет, способствующий удержанию внимания участников и рациональному распределению игрового времени.

Одним из наиболее известных и устойчивых форматов интеллектуальных игр является телевизионная передача “Своя игра”— российская адаптация американского шоу Jeopardy!, созданного в 1964 году Мервом Гриффином. В России передача впервые вышла в эфир в 1994 году. Участники выбирают вопросы различной сложности, разделённые по темам и стоимости, что вносит элемент стратегии в игру: важно не только обладать знаниями, но и правильно оценивать риски при выборе вопросов. Процесс телевизионной игры приведен на рисунке 1.1.



Рисунок 1.1 – телепередача “Своя игра”

Другим знаковым примером является передача “Что? Где? Когда?”— советская и российская телевизионная интеллектуальная игра, созданная в 1975 году Владимиром Ворошиловым и Наталией Стеценко. В данной игре команда из шести “знатоков” за одну минуту должна найти верный ответ на вопрос телезрителя, используя элементы мозгового штурма. За каждый правильный ответ команда получает очко, за каждый неправильный — очко получают телезрители. Побеждает сторона, первой набравшая шесть очков. Популярность игры привела к появлению её спортивной версии и развитию клубного движения по всей стране. Программа до сих пор транслируется на российском телевидении. Момент игры представлен на рисунке 1.2.



Рисунок 1.2 – Ход игры "Что? Где? Когда?"

Исторически развитие интеллектуальных игр активизировалось в XX веке с распространением телевидения. С переходом в цифровую эпоху стало возможным создание компьютерных версий игр, обладающих широкими возможностями: автоматизация расчёта результатов, добавление сетевого мультиплеера, возможность редактирования вопросов и рейтинговой системы. Примером удачной реализации такого проекта является программа SiGame от Владимира Хиля, поддерживающая многопользовательский режим, редактор вопросов и интеграцию с игровым сообществом.

Таким образом, интеллектуальные игры эволюционировали из элементов телешоу в самостоятельное направление цифровых развивающих продуктов. Их адаптация к компьютерной среде открывает новые горизонты для образования, досуга и развития когнитивных способностей пользователей.

1.2 Инструменты и приложения для создания игр

Разработка десктопных приложений, таких как интеллектуальная игра "Своя игра", требует использования современных и мощных инструментов, обеспечивающих высокую производительность, удобство разработки и гибкость адаптации под различные задачи. В применении наиболее часто используются:

1 C# (C-Sharp) — современный объектно-ориентированный язык программирования от Microsoft, активно используемый при разработке десктопных приложений под Windows. Язык сочетает в себе простоту синтаксиса, типизацию, высокоуровневые абстракции и широкую поддержку библиотек. При разработке “Своей игры” на C# использовались механизмы событийной модели, асинхронного программирования (async/await) и взаимодействия с базой данных, что позволило обеспечить отзывчивый интерфейс и корректную обработку сетевых операций. Благодаря тесной интеграции с Visual Studio и .NET, C# остаётся одним из наиболее удобных и надёжных языков для разработки настольных приложений.

2 Microsoft Visual Studio — одна из ведущих интегрированных сред разработки (IDE) для платформы Windows. В этот момент наиболее часто используется версия 2022 Community Edition, предоставляемая бесплатно и обладающая мощным функционалом: подсветка синтаксиса, автодополнение кода, встроенные средства отладки, поддержка множества языков программирования, включая C#. Среда обеспечивает тесную интеграцию с платформами .NET Framework и .NET Core, что даёт возможность создавать как классические десктопные, так и кроссплатформенные приложения. Особенно важна встроенная поддержка отладочных инструментов — трассировка, профилирование, точечные остановки — что критично при разработке сетевых и логически насыщенных программ. Логотип Visual Studio представлен на рисунке 1.3.



Рисунок 1.3 – Логотип среды разработки Visual Studio

3 WPF (Windows Presentation Foundation) — это графическая

подсистема, встроенная в .NET, предназначенная для создания современных и насыщенных пользовательских интерфейсов. Одним из её главных преимуществ является использование XAML — декларативного языка для описания визуальной структуры интерфейса. Это разделяет представление и бизнес-логику, упрощая разработку. WPF поддерживает векторную графику, анимации, масштабируемые элементы, привязку данных (data binding) и использует аппаратное ускорение через DirectX, что делает его мощным инструментом для построения интерактивных интерфейсов с высокой производительностью.

4 MySQL — популярная система управления реляционными базами данных с открытым исходным кодом. Благодаря архитектуре клиент-сервер, она отлично подходит для многопользовательских приложений, включая сетевые игры, в которых данные должны быть централизованно доступны. Для взаимодействия с C# используется библиотека MySql.Data, устанавливаемая через NuGet. Она позволяет выполнять основные SQL-запросы (SELECT, INSERT, UPDATE и др.) и работать с транзакциями. В процессе разработки часто применяется MySQL Community Server, предоставляющий весь необходимый функционал бесплатно. Логотип представлен на рисунке 1.4.



Рисунок 1.4 – Логотип MySQL

5 XAML (Extensible Application Markup Language) — декларативный язык, используемый для создания интерфейсов в WPF. С его помощью задаются визуальные элементы, их свойства и логика взаимодействия с данными. Он поддерживает повторное использование шаблонов (styles, control templates), а также легко интегрируется с инструментами визуального проектирования. Это значительно ускоряет и упрощает создание гибких пользовательских интерфейсов.

6 Qt — это кроссплатформенный фреймворк для разработки графических пользовательских интерфейсов и приложений с использованием языка C++. Он активно используется в случаях, когда требуется высокая производительность, гибкость и возможность компиляции под различные операционные системы, включая Windows, macOS и Linux. Qt предоставляет мощный инструментарий для построения интерфейсов, управления событиями, работы с файлами, сетью и мультимедиа. Также он включает в

себя встроенные средства визуального проектирования (Qt Designer), что значительно упрощает создание GUI. Несмотря на использование C++, Qt обеспечивает уровень абстракции, сопоставимый с современными высокоуровневыми средами разработки.

Совокупное применение перечисленных технологий позволяет разрабатывать мощные, масштабируемые и удобные в использовании приложения. Microsoft Visual Studio и C# формируют надёжную основу разработки, WPF и XAML предоставляют современные средства построения интерфейсов, MySQL — централизованное хранилище данных, а Qt представляет альтернативный путь реализации сложных пользовательских приложений с кроссплатформенной поддержкой. Такой подход обеспечивает реализацию всех запланированных функций, включая игровые механики, многопользовательский режим и динамический интерфейс, с акцентом на стабильность, производительность и расширяемость.

1.3 Анализ существующих разработок компьютерных интеллектуальных игр

Интеллектуальные игры в цифровом формате существуют давно, и одной из самых известных реализаций является SiGame от Владимира Хиля, созданная в 2010-х годах. SiGame поддерживает сетевой и локальный режимы, позволяет пользователям создавать свои пакеты вопросов и проводить игры с ведущим, который оценивает ответы. Однако у SiGame есть недостатки: интерфейс устарел, а поддержка рейтинговой системы отсутствует. Логотип программы представлен на рисунке 1.5.



Рисунок 1.5 – Логотип приложения "SiGame"

Другим примером является мобильное приложение "Jeopardy! World Tour", выпущенное в 2017 году. Оно ориентировано на одиночную игру с возможностью соревнования с другими игроками онлайн, но не поддерживает локальный режим и редактор вопросов. Интерфейс приложения представлен на рисунке 1.6.



Рисунок 1.6 – Логотип приложения " Jeopardy! World Tour "

На основе анализа существующих решений можно сделать вывод, что современные пользователи ценят гибкость (поддержка разных режимов), возможность создавать собственный контент и соревновательный элемент (например, глобальный рейтинг). Данный проект стремится объединить эти аспекты, добавив современный интерфейс и систему рейтинга.

1.4 Постановка задачи

В рамках курсовой работы планируется разработка десктопного приложения интеллектуальной игры на платформе Windows. Приложение должно поддерживать три режима: одиночная игра, игра против бота (искусственного интеллекта) и игра с другом на одном устройстве. Пользователь начинает игру с нулевым количеством очков. Перед началом он выбирает один из доступных пакетов вопросов. Повторное прохождение уже сыгранного пакета не допускается.

Игра состоит из четырёх раундов. Первый раунд — это классическая викторина. В нём от одной до пяти тем, каждая содержит от одного до пяти вопросов, объединённых общей тематикой. Каждый вопрос имеет четыре варианта ответа, из которых только один правильный. За правильный ответ игрок получает очки, равные стоимости вопроса, за неправильный — очки не изменяются.

Второй раунд — "Своя игра". Его структура аналогична первому, но вопросы требуют ввода текстового ответа. Игроку даётся одна попытка. За правильный ответ начисляется номинал вопроса, за неправильный — вычитается. При этом игрок может не отвечать, тогда счёт останется прежним. Для проверки используется алгоритм нечёткого сравнения (расстояние Левенштейна). Ответ считается верным, если он отличается от одного из 1–3 авторских вариантов не более чем на два символа.

Третий раунд — "Что? Где? Когда?". В нём от одной до пяти тем, в каждой от одного до двух вопросов. Правила ответа и оценки аналогичны второму раунду, но за неправильный ответ очки не вычитаются. Ответ также сравнивается с авторскими с учётом допустимой погрешности.

Четвёртый раунд — финальный вопрос, в котором игрок делает ставку. Если у него положительное количество очков, он может поставить от одного до всей суммы. Если очков нет или значение отрицательное, ставка автоматически равна одному. За правильный ответ игрок получает очки, равные ставке, за неправильный — теряет их.

В конце игры пересчитывается рейтинг и присуждаются достижения. В режимах игры с другом и против бота используется таймер для обеспечения честной игры. В приложении также предусмотрен редактор вопросов, с помощью которого пользователь может создавать и редактировать собственные пакеты, сохранять их в базе данных и делиться ими с другими.

Цель проекта — разработка функционального и удобного приложения с продуманной системой очков, рейтингов и достижений. Предусматривается яркий и интуитивно понятный интерфейс, музыкальное сопровождение, гибкий редактор вопросов и поддержка разных сценариев игры. Важно обеспечить устойчивую работу всех компонентов и корректную обработку пользовательских данных.

Основные задачи включают разработку игровых и интерфейсных модулей, проектирование структуры данных для хранения пакетов вопросов, очков и рейтингов, реализацию механик выбора, ответа и подсчёта баллов, настройку финального раунда и логики ставок, а также подключение алгоритма нечёткого сравнения для текстовых ответов. Кроме того, потребуется реализовать систему рейтингов, достижений и глобального лидерборда, а также провести тестирование и оптимизацию приложения.

Для реализации проекта используются язык программирования C#, фреймворк WPF, среда разработки Visual Studio и база данных MySQL.

2 АРХИТЕКТУРА ИГРОВОГО ПРИЛОЖЕНИЯ

2.1 Структура игры и её основные компоненты

Приложение "MEGAGame" представляет собой десктопное решение для проведения интеллектуальных викторин, ориентированное на платформу Windows. Его основная цель — предоставить пользователям удобную среду для соревнования в знаниях, а также реализовать функциональность управления игровым процессом с возможностью редактирования контента. Архитектура проекта построена с опорой на паттерн MVVM (Model–View–ViewModel), обеспечивающий чёткое разделение логики, данных и визуального представления, что упрощает сопровождение, расширение и тестирование приложения.

2.1.1 Архитектура проекта

Приложение состоит из двух ключевых компонентов (подпроектов):

1. MEGAGame.Client

Это основной пользовательский интерфейс приложения. Он реализует представления (View) и модели представления (ViewModel), обеспечивая взаимодействие с пользователем. Включает все визуальные окна, такие как:

- Главное меню
- Игровое поле (в зависимости от выбранного режима)
- Окна с вопросами и ответами
- Окна финального раунда и результатов
- Редактор вопросов и пакетов
- Окна настроек, рейтинга, достижений и лидерборда.

Каждое окно связано с соответствующим ViewModel-классом, в котором реализуется логика отображения, привязки данных и взаимодействия с моделью (Model). Вся навигация и управление окнами реализуются с помощью команд и привязок данных (Binding, ICommand), что соответствует принципам MVVM.

2. MEGAGame.Core

Этот модуль инкапсулирует бизнес-логику и работу с базой данных. Он содержит:

- Модели данных (классы, описывающие структуру пакетов вопросов, пользователей, рейтингов, достижений и т.д.)
- Интерфейсы и реализации сервисов доступа к данным (через Repository или DbContext на базе Entity Framework Core)
- Классы конфигурации и миграции базы данных
- Логiku игровых режимов (настройки, правила, расчёт очков, проверка ответов)
- Алгоритмы, включая реализацию нечеткого поиска (на базе расстояния Левенштейна)
- Поддержку сериализации и импорта/экспорта пакетов вопросов
- Модуль Core не зависит от UI и может использоваться повторно или быть протестирован отдельно.

2.1.2 MVVM-подход

Структура приложения строго следует принципам MVVM:

Model — предоставляет структуру данных, взаимодействует с базой, хранит бизнес-логику. Все модели сериализуемы и подлежат валидации.

- ViewModel — связывает модель с представлением. Здесь реализуются:
 - команды для UI (RelayCommand, AsyncCommand)
 - обработка пользовательского ввода
 - логика смены состояний окон (например, переключение между раундами, загрузка вопросов, подсчёт очков)
 - взаимодействие с Core (через сервисы и репозитории)
- View — XAML-файлы, реализующие интерфейс пользователя. Все элементы управления (кнопки, текстовые поля, списки и пр.) привязаны к свойствам ViewModel через механизмы Binding, DataContext, INotifyPropertyChanged.

Благодаря MVVM обеспечивается полная отделённость интерфейса от логики, что делает возможной простую подмену представлений, масштабируемость и удобство тестирования логики без необходимости запуска интерфейса.

2.2 Игровой процесс: основные механики и логика

Приложение MEGAGame — это десктопное интеллектуальное викторинное решение, разработанное с использованием архитектуры MVVM. Оно предназначено как для индивидуальной, так и для коллективной игры, и включает полноценную механику игрового процесса, редактор контента, а также рейтинговую и достиженческую систему. Главной целью является создание удобного, динамичного и захватывающего пространства для соревнований в знаниях.

Приложение поддерживает три игровых режима: одиночная игра, игра против бота (искусственного интеллекта) и игра с другом на одном устройстве. Все режимы предполагают участие одного или двух игроков, без использования сетевых функций. В начале каждой игры у игрока ноль очков. Он выбирает пакет вопросов, доступный в библиотеке. Повторное прохождение одного и того же пакета запрещено, чтобы сохранить честность соревнования и актуальность рейтинговой системы.

Игровой процесс делится на четыре раунда. Каждый раунд строится на своей логике и оценке ответов, что делает игру разнообразной и комплексной.

Первый раунд представляет собой классическую викторину. Игроку доступны от одной до пяти тематик, каждая из которых включает от одного до пяти вопросов. Все вопросы в рамках одной тематики объединены общим смыслом. Для каждого вопроса предоставляется четыре варианта ответа, только один из которых является правильным. За верный ответ начисляются очки, равные номиналу вопроса. При ошибочном выборе очки не вычитаются.

Второй раунд реализован по принципу "Своя игра". Структура аналогична первому раунду — от одной до пяти тематик, в каждой от одного до пяти вопросов. Однако игроку необходимо самостоятельно ввести текст ответа, и даётся только одна попытка. За правильный ответ начисляются очки, равные номиналу вопроса. За неверный — вычитается такая же сумма. Если игрок пропускает вопрос (оставляет поле пустым), количество его очков остаётся неизменным. Оценка ответов проводится с использованием алгоритма нечеткого поиска, основанного на расстоянии Левенштейна. В базе предусмотрено до трёх авторских вариантов ответов. Ответ считается верным, если он отличается от любого из этих вариантов не более чем на два символа, что позволяет избежать спорных ситуаций и признавать допустимые опечатки.

Третий раунд выполнен в стиле "Что? Где? Когда?". Он также предполагает от одной до пяти тематик, но в каждой теме содержится от одного до двух сложных вопросов. Игроку даётся одна попытка ввести свой ответ. За правильный ответ он получает очки, равные номиналу вопроса. За неправильный — очки не изменяются. Проверка осуществляется по тому же принципу, что и во втором раунде — с помощью сравнения по расстоянию Левенштейна с авторскими ответами.

Четвёртый, финальный раунд представляет собой решающий вопрос со ставкой. Если у игрока положительное количество очков, он может сделать ставку от одного до всей имеющейся суммы. Если очков ноль или меньше — ставка по умолчанию равна одному. В случае правильного ответа игрок получает дополнительно сумму ставки, а при ошибке теряет её. Это позволяет игроку как отыграться, так и потерять лидерство на последних минутах игры.

По завершении всех раундов производится перерасчёт рейтинга игрока и проверка на выполнение условий для получения достижений. Это стимулирует участие в игре и рост интереса к повторным попыткам в других режимах или с новыми пакетами.

Во всех режимах игры с соперником предусмотрены таймеры на каждый вопрос, чтобы избежать затягивания времени, обеспечить честность и поддерживать динамику геймплея. Искусственный интеллект реализован с возможностью варьировать уровень сложности и адаптируется к теме и текущему прогрессу игрока.

Приложение также содержит полнофункциональный редактор вопросов. Пользователь может создавать собственные пакеты, редактировать уже созданные, а также публиковать их в общий доступ, где другие игроки смогут использовать их в своей игре. Вся информация о пакетах хранится в базе данных, реализованной через Entity Framework Core, с поддержкой миграций и сериализации.

Для повышения вовлечённости реализованы система достижений и рейтинг игроков, основанный на результатах игр в разных режимах. Пользовательский интерфейс выполнен на базе WPF и поддерживает современный визуальный стиль: яркие цветовые схемы, анимации, звуковое сопровождение, адаптивность и интуитивную навигацию.

Таким образом, "MEGAGame" сочетает в себе интеллектуальную нагрузку, соревновательный азарт, возможность творчества и развитую техническую основу, что делает его универсальным и привлекательным инструментом для игр, обучения и досуга.

2.3 Поддержка различных режимов игры

Приложение поддерживает три режима игры: одиночную игру, игру с другом и игру против бота.

В одиночном режиме пользователь проходит четыре раунда вопросов подряд. Таймер при этом не используется, и переход к следующему раунду возможен только после того, как игрок ответит на все вопросы текущего раунда. Игровое поле состоит из квадратных ячеек, каждая из которых содержит вопрос определённой тематики и стоимости. На ячейке указана тема и номинал вопроса. При наведении курсора ячейка подсвечивается. Цвета помогают отличать статус вопроса: зелёным обозначаются несыгранные вопросы, красным — уже сыгранные.

После нажатия на ячейку открывается окно с вопросом. В первом раунде игроку предлагаются четыре варианта ответа, из которых он выбирает один и подтверждает свой выбор кнопкой. Начиная со второго раунда, игрок вводит ответ вручную в текстовое поле и подтверждает его нажатием кнопки. Система использует интеллектуальную проверку на основе расстояния Левенштейна, что позволяет засчитывать ответы с незначительными опечатками. По завершении игры игрок получает прибавку к рейтингу, равную количеству набранных очков, делённому на десять. Игровой пакет считается отыгранным, и его невозможно сыграть повторно ни в одном из режимов. Также игроку начисляются достижения. Среди достижений — три, пять и десять правильных ответов подряд в одной игре, правильные ответы на все вопросы одного раунда, достижение рейтинга свыше 2000, 3000 и 5000, а также победы над ботом на всех уровнях сложности.

В режиме игры с другом два пользователя вводят свои ники в отдельном окне перед началом. Интерфейс повторяет оформление одиночной игры, но добавлены элементы для отображения очков каждого игрока. На каждый вопрос устанавливается таймер: десять секунд в первом раунде, двадцать — во втором, и тридцать — в третьем. Если за основной промежуток времени никто не дал ответ, запускается дополнительный таймер: пять, десять или пятнадцать секунд в зависимости от сложности раунда. В течение дополнительного времени ответить может только тот, кто первым нажмёт на клавишу: клавиша A предназначена для первого игрока, а клавиша L — для второго. Возможность ответить в дополнительное время предоставляется только одному игроку — второму шанса не будет. Раунд со ставками в этом режиме отсутствует. В конце игры очки подсчитываются, определяется победитель, пакет блокируется, а выполненные достижения фиксируются. Изменения рейтинга при этом не происходят.

В режиме игры с ботом игрок сначала выбирает уровень сложности: простой, средний или сложный. Структура игры и таймеров такая же, как и в режиме с другом. Однако вместо второго игрока выступает бот, который симулирует ответ в течение заранее заданного диапазона времени, выбирая конкретное значение случайным образом. Вероятность правильного ответа зависит от выбранной сложности: на простом уровне бот часто ошибается, на среднем отвечает правильно примерно в половине случаев, а на сложном — в большинстве. На легком уровне бот верно отвечает на 1 из 5 вопросов, на среднем уровне бот верно отвечает на 3 из 5 вопросов, на сложном уровне бот верно отвечает на 4 из 5 вопросов. Бот не просто статистическая модель, он динамически реагирует на игровую ситуацию. В случае победы над ботом игроку начисляется соответствующее достижение. По окончании игры пакет блокируется, а рейтинг увеличивается согласно набранным очкам.

3 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ИГРЫ

3.1 Анализ используемых программных средств, инструментов для разработки

Разработка настольного приложения интеллектуальной игры "Своя игра" требует применения современных и функциональных средств программирования, обеспечивающих высокую производительность и удобство при создании сложной логики и интерфейса. В рамках проекта были выбраны следующие инструменты и технологии.

Средой разработки для создания проекта выступила Visual Studio 2022 Community Edition — популярная и мощная IDE от Microsoft, предоставляющая широкий набор возможностей для написания, отладки и сопровождения программ на языке C#. Данная версия бесплатна и отлично подходит для разработки Windows-приложений благодаря поддержке современных технологий .NET, интеграции с менеджером пакетов NuGet и множеству встроенных инструментов для анализа кода, управления проектом и профилирования производительности. Эти функции помогают ускорить разработку и повысить стабильность приложения.

Для создания интерфейса была использована платформа Windows Presentation Foundation (WPF), которая входит в экосистему .NET и позволяет строить визуально привлекательные и отзывчивые пользовательские интерфейсы. С помощью декларативного языка разметки XAML достигается чёткое разделение логики и визуального оформления, что облегчает работу как программистам, так и дизайнерам. Фреймворк предоставляет мощные возможности для создания анимаций, масштабируемой векторной графики и организации привязки данных, позволяя динамически отображать текущие результаты игры и создавать плавные переходы между экранами. Кроме того, WPF оптимизирован для использования ресурсов многоядерных процессоров и аппаратного ускорения, что улучшает общую производительность и качество визуальных эффектов.

XAML, как язык описания интерфейсов, применяется для построения элементов управления — кнопок, таблиц, текстовых полей и панелей, а также для настройки их внешнего вида и поведения. Это позволяет формировать гибкие шаблоны оформления и связывать визуальные компоненты с данными, обеспечивая автоматическое обновление интерфейса при изменении состояния приложения. Применение XAML способствует созданию удобного и интуитивного пользовательского опыта.

Для хранения и управления данными в проекте используется реляционная система управления базами данных MySQL, которая обеспечивает централизованное хранение информации о вопросах, пользователях и результатах игр. В отличие от встроенных легковесных решений, MySQL рассчитана на работу в клиент-серверном режиме, что особенно актуально при необходимости совместного использования данных. Взаимодействие с базой происходит через специализированную библиотеку MySql.Data, которая интегрируется с C# и позволяет выполнять эффективные SQL-запросы для выборки и обновления информации. MySQL поддерживает создание структурированных таблиц с индексами, что ускоряет доступ к

данным и обеспечивает надежность хранения. Для локальной разработки и тестирования применяется бесплатный дистрибутив MySQL Community Server.

В целях безопасности паролей пользователей реализована система хэширования с использованием алгоритма BCrypt. Этот метод гарантирует, что пароли никогда не сохраняются в открытом виде, а вместо них хранится только их хэш, созданный с помощью уникальной соли и заданного уровня сложности вычислений. Такой подход защищает от атак с применением радужных таблиц и перебора, поскольку процесс восстановления исходного пароля практически невозможен. При входе в систему введенный пользователем пароль повторно обрабатывается с использованием той же соли, и результаты сравниваются с сохранённым хэшем, обеспечивая надёжную проверку подлинности без компрометации данных.

3.2 Описание реализации основного функционала игры

Основной функционал интеллектуальной игры "Своя игра" реализован с использованием выбранных технологий и инструментов, что обеспечило создание стабильного и удобного приложения с богатыми возможностями взаимодействия и высокой производительностью.

В основе логики игры лежит обработка пользовательских действий и управление состояниями игры, что реализовано на языке C# в среде Visual Studio. Приложение поддерживает разные игровые режимы, включая одиночную игру, игру с другом и игру с ботом, каждый из которых имеет свои особенности и интерфейсные решения. Взаимодействие с пользователем происходит через графический интерфейс, построенный с помощью WPF и описанный декларативно на языке XAML. Это позволило создать интуитивно понятный и динамичный UI, который обновляется в режиме реального времени благодаря механизму привязки данных (Data Binding). Игровое поле, меню, окна вопросов и редактор контента — все эти компоненты разработаны с применением визуальных стилей и анимаций, обеспечивающих плавность и визуальную привлекательность.

Для хранения данных и обеспечения многопользовательской работы, в том числе поддержки рейтинга и истории игр, используется серверная база данных MySQL. Она служит центральным хранилищем пакетов вопросов, пользовательских профилей и результатов, обеспечивая быстрый и надежный доступ к информации. Через библиотеку MySql.Data осуществляется выполнение необходимых запросов для загрузки, обновления и сохранения данных, что позволяет синхронизировать состояние игры и поддерживать актуальность рейтинга.

Особое внимание уделено безопасности пользовательских данных: для защиты паролей применяется алгоритм BCrypt, который предотвращает хранение паролей в открытом виде и обеспечивает устойчивость к атакам методом перебора. Это повышает уровень доверия к приложению и защищает

конфиденциальную информацию пользователей.

В целом, интеграция перечисленных технологий позволила создать комплексное приложение с удобным интерфейсом, надежной системой хранения данных и безопасным управлением учетными записями, что обеспечивает комфортный и увлекательный игровой процесс.

4 ТЕСТИРОВАНИЕ ИГРОВОГО ПРИЛОЖЕНИЯ

В процессе тестирования игры, было проведено тщательное и последовательное исследование всех ключевых сценариев взаимодействия пользователя с интерфейсом и основными механиками игрового процесса.

4.1 Основные этапы

Тестирование проходило в несколько этапов, каждый из которых включал проверку конкретных функций и систем игры. Особое внимание уделялось выявлению ошибок и проблем, их классификации и разработке методов устранения.

4.1.1 Отображение меню LoginWindow

В этом меню демонстрируются три поля ввода: никнейм, электронная почта и пароль. Пользователь может выбрать два варианта: войти либо зарегистрироваться. Все поля ввода являются обязательными для заполнения, и система проверяет, чтобы ни одно из них не оставалось пустым. Если пользователь хочет войти, но пароль не совпадает или пользователя с указанным никнеймом и электронной почтой не существует, то выводится сообщение с предупреждением. Если пользователь выбирает регистрацию, но никнейм или электронная почта уже заняты другими пользователями, ему потребуется изменить данные. После успешного входа или регистрации пользователь переходит в главное меню MainMenuWindow. Также доступна возможность завершить работу приложения. Интерфейс окна приведен на рисунке 4.1.

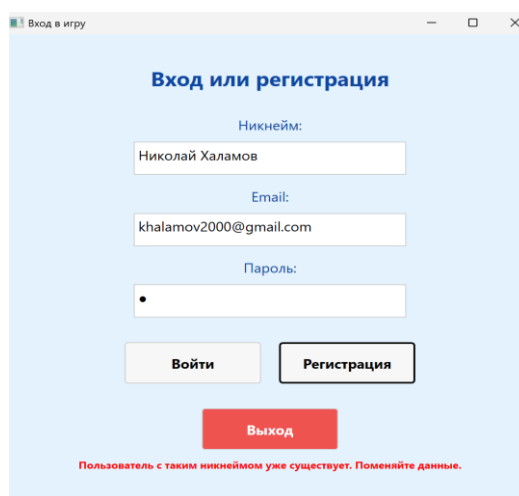


Рисунок 4.1 – Меню регистрации и входа

4.1.2 Тестирование MainMenuWindow

В главном меню приложения MEGA Game пользователь в любой момент может завершить работу приложения, воспользовавшись кнопкой "Выход", что приводит к остановке фоновой музыки и завершению работы программы. На верхней части интерфейса отображаются имя пользователя (ник) и его текущий рейтинг, предоставляя игроку персонализированную информацию о

его статусе в игре. Игрок имеет возможность выбрать один из пяти фоновых музыкальных треков или полностью отключить музыку. Выбранные треки автоматически продолжают по порядку после завершения текущего трека и остаются активными при переходах в другие окна приложения, обеспечивая непрерывное музыкальное сопровождение во время работы.

Далее пользователь может выбрать, чем заняться, используя интуитивно понятный интерфейс с кнопками, расположенными в два столбца. Левый столбец включает игровые режимы: "Одиночная игра", "Игра с другом", "Игра против бота", каждый из которых открывает соответствующее окно для выбора пакета вопросов. Правый столбец предлагает дополнительные функции: "Рейтинг" для просмотра таблицы лидеров, "Достижения" для ознакомления с личными успехами, "Редактор вопросов" для создания и редактирования пакетов вопросов, а также "Выход" для завершения работы. Эти опции обеспечивают широкий спектр возможностей для взаимодействия с игрой, адаптируясь под предпочтения и навыки пользователя. Интерфейс окна приведен на рисунке 4.2.

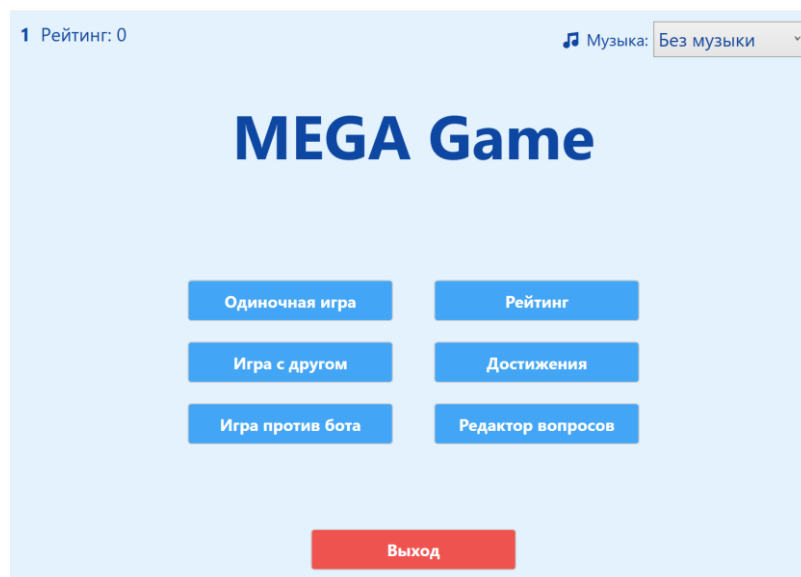


Рисунок 4.2 – Главное меню

4.1.3 Тестирование SinglePlayerPackSelectionWindow

Перед началом игры в каждом режиме запускается окно выбора пакета вопросов, где отображаются все выложенные пакеты, но при этом некоторые пакеты уже заблокированы. Сыгранные пакеты выделяются красным цветом, доступные выделяются зеленым цветом. Интерфейс окна приведен на рисунке 4.3.

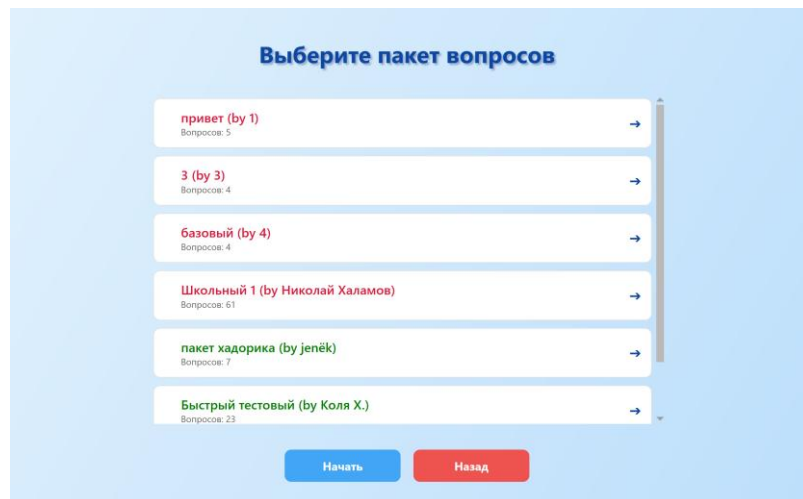


Рисунок 4.3 – Окно выбора пакета для игры

4.1.4 Тестирование RankingWindow

В данном окне отображается полный мировой рейтинг всех игроков по убыванию рейтинга, где указывается позиция, никнейм и рейтинг. Лучшие 3 игрока анимированы с медалями. Интерфейс окна приведен на рисунке 4.4.

Место	Никнейм	Рейтинг
1 🏆	jenëk	3900.70
2 🥈	Николай Халамов	3191.00
3 🥉	Тест достижений	2028.40
4	тест	1830.00
5	12212	1790.10
6	4	1720.30
7	тест достижения	1675.05
8	вргц	1667.50
9	тест 6	1640.90
10	vrgseqdqcqwefc	1612.10
11	тест 9	1600.20
12	Коля	1577.70
13	тест левенштейна	1567.50
14	вццу	1557.90
15	цв	1545.00

Рисунок 4.4 – Рейтинг игроков

4.1.5 Тестирование AchievementsWindow

В данном окне показываются все возможные достижения и те достижения, которые игрок выполнил, окрашены в другой цвет. Интерфейс

окна приведен на рисунке 4.5.

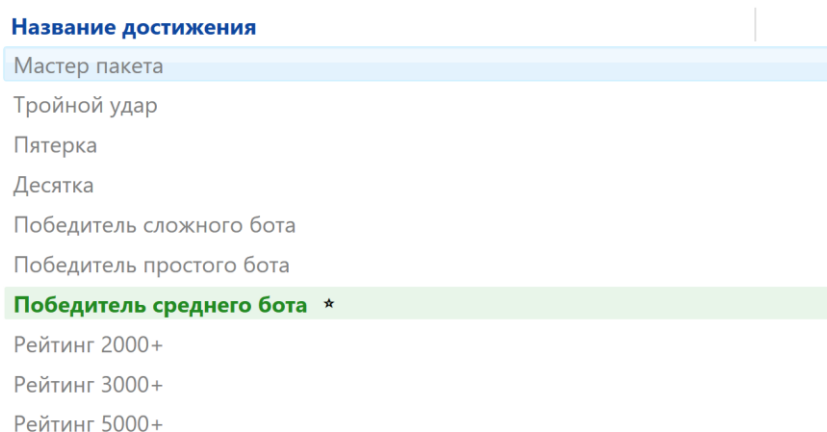


Рисунок 4.5 – Достижения игрока

4.1.6 Тестирование BotDifficultySelectionWindow

Перед запуском игры с ботом пользователь в этом окне выбирает сложность бота(ИИ). Интерфейс окна приведен на рисунке 4.6.

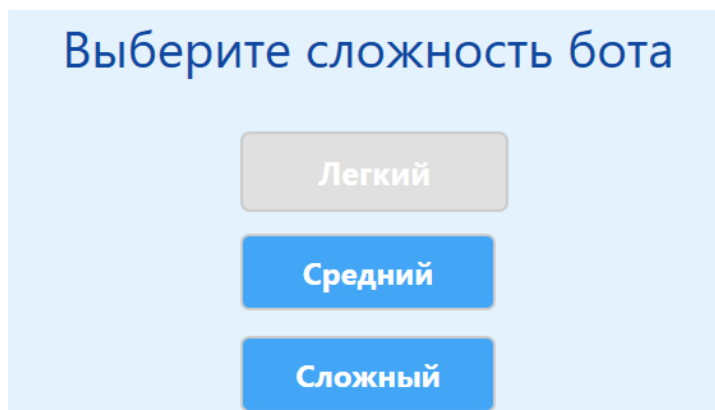


Рисунок 4.6 – Выбор сложности бота, выбран легкий

4.1.7 Тестирование QuestionEditorWindow

Это главное окно редактора вопросов, на которое можно перейти из главного меню. Можно создать или выбрать уже существующий свой пакет для создания или редактирования. Ниже отображается полный список вопросов и тем пакета. Их можно редактировать либо полностью удалять из базы данных. Пакет можно опубликовать после создания, если в каждом раунде присутствует минимум 1 тема, и в каждой созданной теме есть минимум 1 вопрос. После этого пакет смогут играть все пользователи. Интерфейс окна приведен на рисунке 4.7.

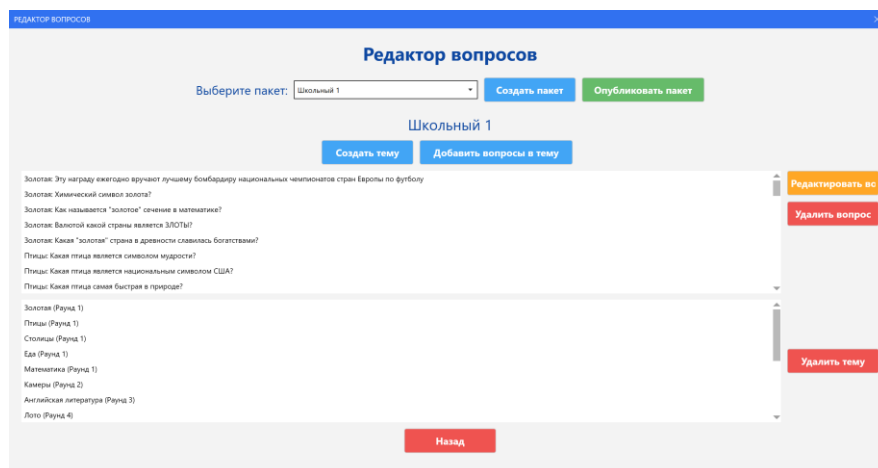


Рисунок 4.7 – Редактор вопросов

4.1.8 Тестирование QuestionEditWindowRound1

В этом окне можно редактировать вопрос 1 раунда. В полях уже виден существующий текст с вариантами ответа и номиналом. В соответствующих полях все эти значения можно поменять и затем сохранить в базе данных. Интерфейс окна приведен на рисунке 4.8.

Рисунок 4.8 – Редактирование вопроса 1 раунда

4.1.9 Тестирование QuestionEditWindowRound234

В этом окне можно редактировать вопрос 2, 3, 4 раундов. В полях уже виден существующий текст с вариантами ответа и номиналом. В

соответствующих полях все эти значения можно поменять и затем сохранить в базе данных. Интерфейс окна приведен на рисунке 4.9.

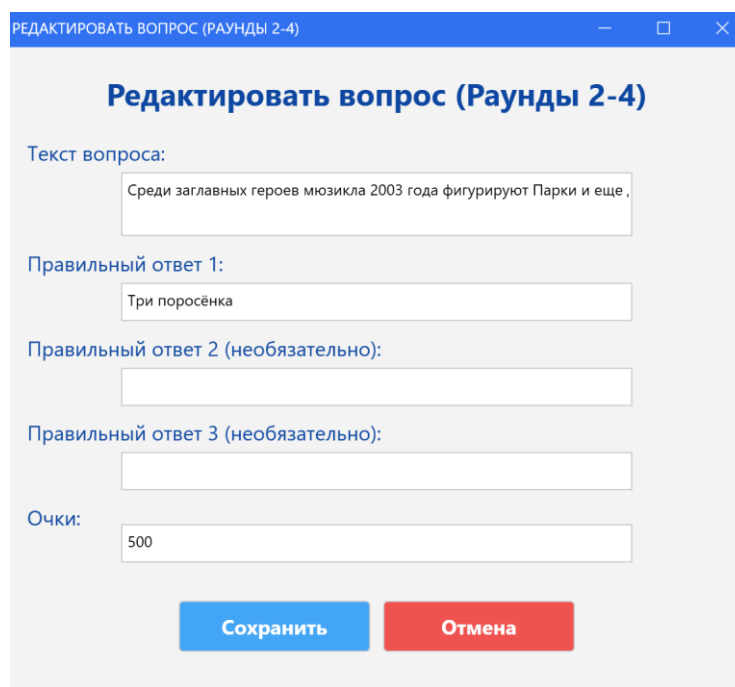


Рисунок 4.9 – Редактирование вопроса 2, 3, 4 раундов

4.1.10 Тестирование QuestionSelectionWindow

Данное окно создается для выбора той темы, в которую будет добавлен вопрос. Интерфейс окна приведен на рисунке 4.10.

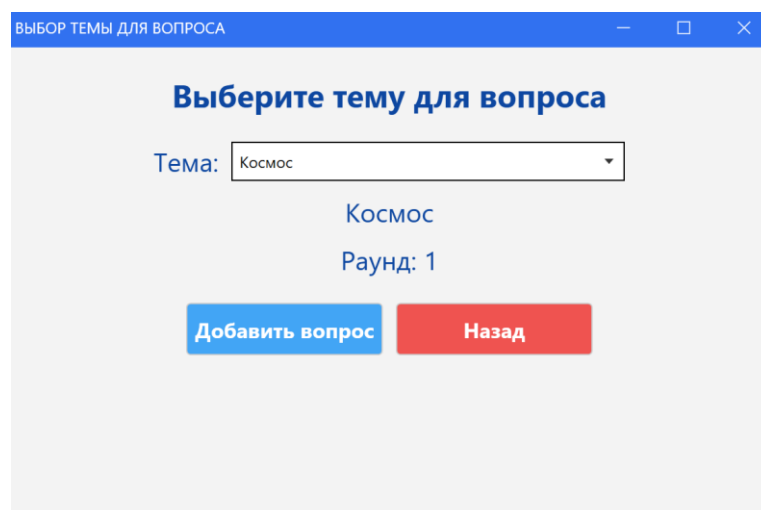


Рисунок 4.10 – Выбор темы для вопроса

4.1.11 Тестирование ThemeCreationWindow

В данном окне пользователь вводит раунд и название темы для ее создания. Интерфейс окна приведен на рисунке 4.11.

Рисунок 4.11 – Создание темы

4.1.12 Тестирование QuestionCreationWindow

Это окно для создания вопроса для любого раунда. Пользователь должен ввести текст, номинал. Для первого раунда нужны варианты ответа, для остальных нужны правильные ответы (1 обязательный и 2 необязательных). Соответственно здесь поля не являются обязательными. После создания одного вопроса пользователь остается в этом окне для удобства создания последующих вопросов темы.

Рисунок 4.12 – Создание вопроса

4.1.13 Тестирование MainWindow

Это основное окно для реализации одиночной игры. Оно открывается после выбора пакета. Здесь есть все ячейки для выбора вопросов, кнопка с

информацией о соответствующем раунде, кнопка ,завершающая игру и выводящая конечный счет, после выбора вопроса выводится текст вопроса и варианты ответа для 1 раунда и поле для ввода ответа для других раундов. После отыгрыша всех вопросов раунда появляется кнопка для перехода к следующему раунду. Интерфейс первого раунда приведен на рисунке 4.13.



Рисунок 4.13 – Первый раунд

Интерфейс второго раунда приведен на рисунке 4.14.



Рисунок 4.14 – Второй раунд

Интерфейс третьего раунда приведен на рисунке 4.15.

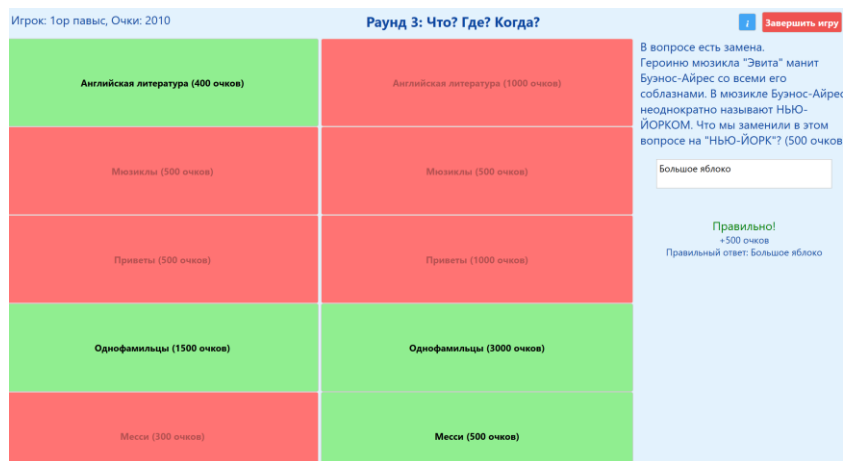


Рисунок 4.15 – Третий раунд

Интерфейс четвертого раунда приведен на рисунке 4.16.



Рисунок 4.16 – Четвертый раунд

Интерфейс окночания игры приведен на рисунке 4.17.

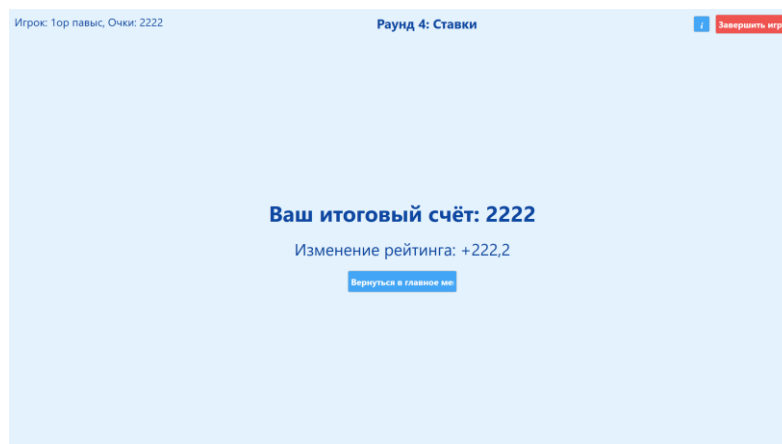


Рисунок 4.17 – Конец игры

4.1.14 Тестирование FriendGameWindow

Это основное окно игры с другом. Оно открывается после выбора пакета и ввода ников. Здесь есть все ячейки для выбора вопросов, кнопка с информацией о соответствующем раунде, кнопка, завершающая игру и выводящая конечный счет, после выбора вопроса выводится текст вопроса и варианты ответа для 1 раунда и поле для ввода ответа для других раундов. После отыгрыша всех вопросов раунда появляется кнопка для перехода к следующему раунду. При этом отображается счет каждого игрока. На каждом вопросе выдаются нужные подсказки для понимания игроками интерфейса и таймер. После каждого вопроса идет информация о набранных игроками очках. В конце идет окно с выводом счета игроков и победителем.

4.1.15 Тестирование BotGameWindow

Это основное окно игры с ботом. Оно открывается после выбора пакета и сложности бота. Здесь есть все ячейки для выбора вопросов, кнопка с информацией о соответствующем раунде, кнопка, завершающая игру и выводящая конечный счет, после выбора вопроса выводится текст вопроса и варианты ответа для 1 раунда и поле для ввода ответа для других раундов. После отыгрыша всех вопросов раунда появляется кнопка для перехода к следующему раунду. При этом отображается счет каждого игрока. На каждом вопросе выдаются нужные подсказки для понимания игроком интерфейса и таймер. После каждого вопроса идет информация о набранных игроками очках. В конце идет окно с выводом счета игроков и победителем.

4.1.16 Тестирование общей работы проекта и связи всех режимов

Таким образом, приложение представляет функционал для связанной работы разных режимов и понятного интерфейса интересной интеллектуальной игры.

4.2 Выявленные проблемы и способы их решения

В ходе тестирования игры были выявлены несколько ключевых проблем, которые негативно сказывались на визуальном восприятии и игровом процессе. Для каждой из них были разработаны и внедрены эффективные решения, значительно повысившие стабильность и удобство управления. Рассмотрим некоторые из них и способы их решения:

1 Неудобное расположение ячеек выбора и тяжелый интерфейс. Было решено расставить ячейки на большую часть экрана, добавить разный цвет, удобное поле с вопросом и вводом ответа для удобного взаимодействия. Также была добавлена кнопка с информацией для

пользователя.

2 Отсутствие вариативности ответов. Было решено добавить в базу данных 3 варианта ответа для вопросов 2,3,4 раунда. Это полезно в тех случаях, когда ответ состоит из нескольких слов, но не все они важны, и ответ будет считаться верным без них. Также был внедрен алгоритм нечеткого поиска, который позволил засчитывать те ответы, в которых пользователь случайно либо по незнанию совершил одну или две грамматических либо пунктуационных ошибки. Теперь ответ всегда правильно засчитывается и это не вызывает недовольства пользователя.

3 Невозможность менять вопрос в редакторе вопросов и удалять. Автор пакета может узнавать от игроков об ошибках в пакете и теперь может их исправлять и улучшать качество пакета.

4 Неудобство игры с другом. Когда два игрока играют на одном персональном компьютере, им придется бороться за право первым ответить на одной мышке либо тачпаде. Теперь внедрены кнопки персонально для игроков, чтобы было удобно отвечать первому захотевшему игроку.

Благодаря этим доработкам и оптимизации основных механик игры достигнута высокая стабильность работы, плавность анимаций и комфорт управления, что существенно улучшило общее впечатление от игрового процесса.

ЗАКЛЮЧЕНИЕ

Разработка десктопного приложения для интеллектуальной игры MEGAGame стала значительным шагом в создании современной и функциональной платформы, способной удовлетворить запросы широкой аудитории, увлечённой викторинами и интеллектуальными соревнованиями. Проект, выполненный с использованием языка программирования C#, фреймворка WPF, среды разработки Visual Studio и базы данных MySQL, представляет собой комплексное решение, сочетающее в себе разнообразные игровые режимы, интуитивно понятный интерфейс и мощные инструменты для создания пользовательского контента. Реализация приложения позволила не только воплотить задуманные идеи, но и заложить основу для дальнейшего развития и масштабирования, что делает его востребованным инструментом как для развлечения, так и для образовательных целей.

Основной целью проекта было создание приложения, которое объединяет классические форматы интеллектуальных игр, такие как “Своя игра”, “Что? Где? Когда?” и викторины, в единую цифровую платформу с современным дизайном и гибкими настройками. Эта цель была достигнута благодаря продуманной архитектуре, основанной на паттерне MVVM, который обеспечил чёткое разделение логики, данных и визуального представления. Такое разделение упростило разработку, тестирование и сопровождение приложения, а также открыло перспективы для будущих доработок, таких как добавление сетевого мультиплеера или интеграция с мобильными платформами. Использование WPF и XAML позволило создать визуально привлекательный и адаптивный интерфейс, который одинаково удобен как для новичков, так и для опытных игроков. Динамичные анимации, яркие цветовые схемы и настраиваемое музыкальное сопровождение усилили вовлечённость пользователей, делая игровой процесс эмоционально насыщенным и запоминающимся.

Ключевым достижением проекта стало внедрение разнообразных игровых механик, адаптированных под разные режимы: одиночную игру, соревнование с другом и противостояние с искусственным интеллектом. Каждый режим был тщательно проработан с учётом специфики взаимодействия игроков. Например, в режиме игры с другом добавлены таймеры и персонализированные клавиши для ответа, что решило проблему конкуренции за управление на одном устройстве. В режиме с ботом реализована динамическая модель поведения, зависящая от выбранного уровня сложности, что делает игру с искусственным интеллектом увлекательной и сбалансированной. Алгоритм нечёткого поиска на основе расстояния Левенштейна, применённый для проверки текстовых ответов, стал важным элементом, повысившим гибкость оценки и устранившим раздражение пользователей из-за мелких ошибок в ответах. Эти решения обеспечили справедливость и комфорт в игровом процессе, что особенно важно для поддержания интереса к приложению.

Значительное внимание было уделено системе хранения данных и безопасности. Использование MySQL в качестве серверной базы данных позволило организовать централизованное управление пакетами вопросов, профилями игроков и результатами игр. Это обеспечило надёжность и масштабируемость приложения, а также возможность обмена пользовательскими пакетами вопросов, что стимулирует творческую активность сообщества. Реализация хэширования паролей с помощью алгоритма BCrypt повысила уровень защиты пользовательских данных, что является важным аспектом в условиях растущих требований к кибербезопасности. Редактор вопросов, интегрированный в приложение, предоставил пользователям мощный инструмент для создания и редактирования контента, что делает MEGAGame не только игрой, но и платформой для творчества и обучения.

Тестирование приложения выявило ряд проблем, таких как неудобное расположение элементов интерфейса, ограниченная вариативность ответов и сложности в режиме игры с другом. Все они были успешно устранены путём редизайна интерфейса, внедрения дополнительных вариантов ответов и улучшения механик взаимодействия. Эти доработки повысили стабильность, удобство и общее качество пользовательского опыта. Результаты тестирования подтвердили, что приложение способно работать без сбоев, обеспечивая плавный игровой процесс и корректную обработку данных. Система достижений и глобальный рейтинг дополнительно мотивируют игроков, создавая соревновательную атмосферу и поощряя долгосрочное взаимодействие с приложением.

В заключение, разработанное приложение MEGAGame представляет собой успешную реализацию интеллектуальной игры, которая сочетает в себе развлекательный, образовательный и творческий потенциал. Оно отвечает современным требованиям к функциональности, удобству и безопасности, предлагая пользователям уникальный опыт, объединяющий азарт соревнования, радость открытия новых знаний и возможность самовыражения через создание собственного контента. Проект не только достиг поставленных целей, но и создал прочную основу для дальнейших инноваций, что подтверждает его актуальность и перспективность в контексте развития цифровых интеллектуальных развлечений.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- [1] Стандарт предприятия – Минск : БГУИР, 2024. – 178 с.
- [2] Документация по Visual Studio / Microsoft Learn [Электронный ресурс] – <https://learn.microsoft.com/ru-ru/visualstudio/windows/?view=vs-2022>.
- [3] WPF Documentation / Microsoft Learn [Электронный ресурс] – <https://learn.microsoft.com/ru-ru/dotnet/desktop/wpf/>.
- [4] Паттерны проектирования MVVM / Microsoft Learn [Электронный ресурс] – <https://learn.microsoft.com/ru-ru/dotnet/architecture/maui/mvvm>.
- [5] Лиманова, Н.И. Алгоритм нечеткого поиска в базах данных и его практическая реализация : учебное пособие / Н.И. Лиманова. – 1-е изд. – Самара : ПГУТиФ, 2017. – 1893 - 1897 с.
- [6] Селлс, К. Programming WPF: Building Windows UI with Windows Presentation Foundation / К. Селлс. – 1-е изд. – 2007. – 387 с.
- [7] Нэтан, А. XAML unleashed / А. Нэтан. – 1-е изд. – 2014. – 252 с.
- [8] Тарасов, С.В. СУБД для программиста. Базы данных изнутри/ С.В. Тарасов. – 1-е изд. – Москва : СОЛОН, 2015. – 320 с.

ПРИЛОЖЕНИЕ А

(обязательное)

Листинг программного кода

Листинг А.1 — Содержимое файла Question.cs

```
#nullable enable
using System;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace MEGAGame.Core.Models
{
    public class Question
    {
        [Key] public int QuestionId { get; set; }

        [Required] public string Text { get; set; } = null!;

        [Required] public int Points { get; set; }
        [Required] public int Round { get; set; }

        [Required] public string Option1 { get; set; } = null!;
        [Required] public string Option2 { get; set; } = null!;
        [Required] public string Option3 { get; set; } = null!;
        [Required] public string Option4 { get; set; } = null!;

        public int? CorrectOption { get; set; }

        [Required] public string Answer { get; set; } = null!;
        public string? Answer2 { get; set; }
        public string? Answer3 { get; set; }

        [Required] public int ThemeId { get; set; }
        [ForeignKey(nameof(ThemeId))] public Theme Theme { get; set; } = null!;

        [Required] public int PackId { get; set; }
        [ForeignKey(nameof(PackId))] public QuestionPack Pack { get; set; } = null!;

        [Required] public int CreatedBy { get; set; }
        [ForeignKey(nameof(CreatedBy))] public Player CreatedByPlayer { get; set; } = null!;

        [Required] public DateTime CreatedDate { get; set; }
        [Required] public DateTime LastUpdated { get; set; }
        [Required] public bool IsActive { get; set; }
        [Required] public bool IsPlayed { get; set; }
    }
}
```

Листинг А.2 — Содержимое файла Player.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;

namespace MEGAGame.Core.Models
{
    public class Player
    {
        [Key]
        public int PlayerId { get; set; }
        public string? Username { get; set; }
        public string? Password { get; set; }
        public string? Email { get; set; }
        public DateTime RegistrationDate { get; set; }
        public DateTime LastLogin { get; set; }
        public int Score { get; set; }
        public double Rating { get; set; }
    }
}

```

Листинг А.3 — Содержимое файла ==GameDbContext.cs

```

using Microsoft.EntityFrameworkCore;
using Microsoft.Extensions.Logging;
using MEGAGame.Core.Models;

namespace MEGAGame.Core.Data
{
    public class GameDbContext : DbContext
    {
        public DbSet<Player> Players { get; set; }
        public DbSet<QuestionPack> QuestionPacks { get; set; }
        public DbSet<Theme> Themes { get; set; }
        public DbSet<Question> Questions { get; set; }
        public DbSet<PlayedPack> PlayedPacks { get; set; }
        public DbSet<Achievement> Achievements { get; set; }
        public DbSet<PlayerAchievement> PlayerAchievements { get; set; }

        public GameDbContext() { }

        public GameDbContext(DbContextOptions<GameDbContext> options) :
base(options) { }

        protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
        {
            if (!optionsBuilder.IsConfigured)
            {
                optionsBuilder.UseMySQL("Server=localhost;Database=megagame_db;User=Nikolay;Password=
7b2bru43.2d3we;");
            }
        }
    }
}

```

```

        new MySqlServerVersion(new Version(8, 4, 0)))
        .LogTo(Console.WriteLine, LogLevel.Information)
        .EnableSensitiveDataLogging();
    }
}

protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    modelBuilder.Entity<Question>()
        .HasOne(q => q.Theme)
        .WithMany(t => t.Questions)
        .HasForeignKey(q => q.ThemeId);

    modelBuilder.Entity<Question>()
        .HasOne(q => q.Pack)
        .WithMany(p => p.Questions)
        .HasForeignKey(q => q.PackId);

    modelBuilder.Entity<Question>()
        .HasOne(q => q.CreatedByPlayer)
        .WithMany()
        .HasForeignKey(q => q.CreatedBy);

    modelBuilder.Entity<PlayedPack>()
        .HasOne(pp => pp.Player)
        .WithMany()
        .HasForeignKey(pp => pp.PlayerId);

    modelBuilder.Entity<PlayedPack>()
        .HasOne(pp => pp.Pack)
        .WithMany()
        .HasForeignKey(pp => pp.PackId);

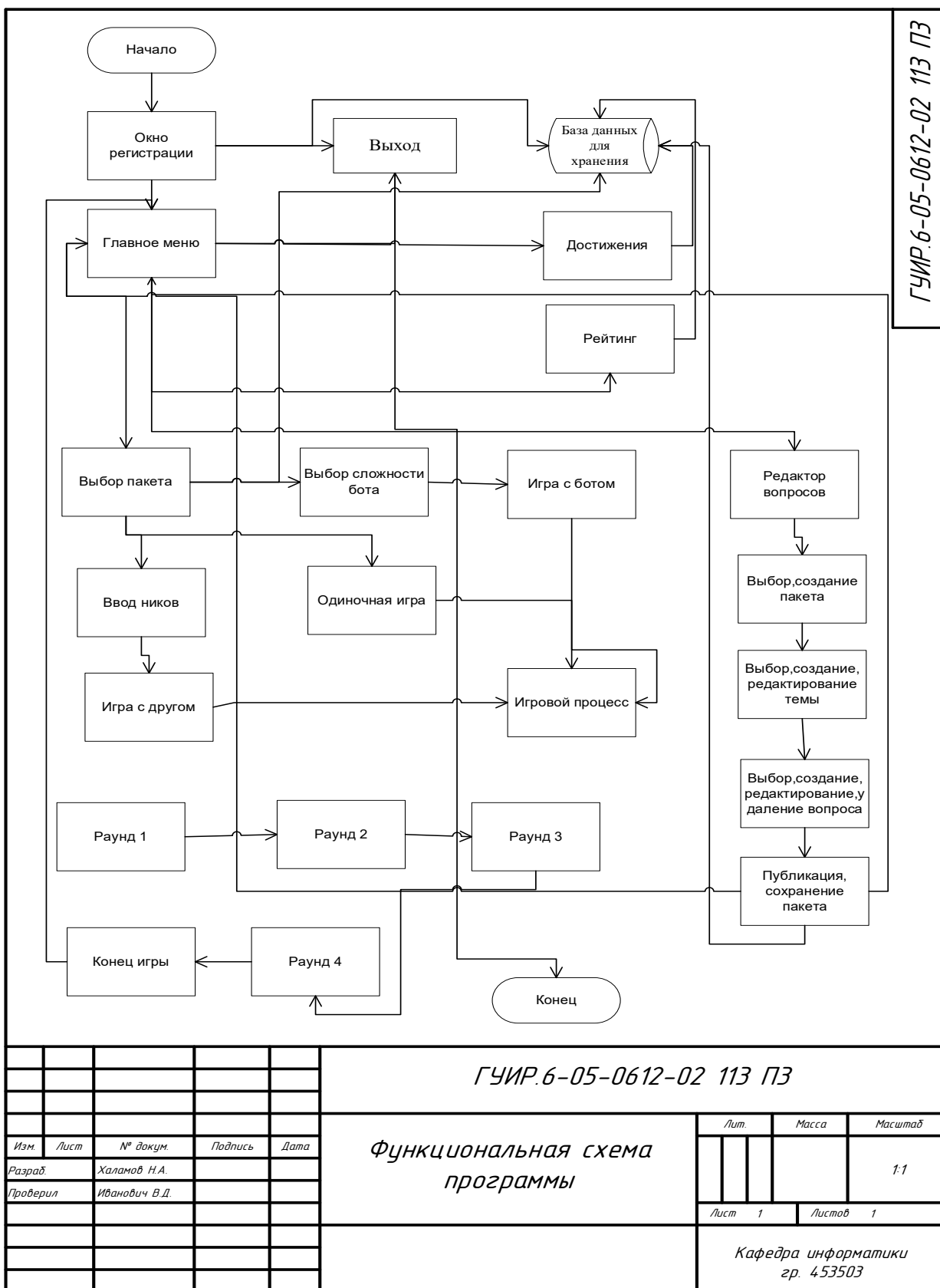
    modelBuilder.Entity<Question>(e =>
    {
        e.Property(q => q.Answer2).IsRequired(false);
        e.Property(q => q.Answer3).IsRequired(false);
    });

    modelBuilder.Entity<PlayerAchievement>()
        .HasOne(pa => pa.Player)
        .WithMany()
        .HasForeignKey(pa => pa.PlayerId);

    modelBuilder.Entity<PlayerAchievement>()
        .HasOne(pa => pa.Achievement)
        .WithMany()
        .HasForeignKey(pa => pa.AchievementId);
    }
}

```

Приложение Б **(обязательное)** **Функциональная схема программы**



ГУИР.6-05-0612-02 113 ПЗ

ГУИР.6-05-0612-02 113 ПЗ

**Функциональная схема
программы**

Изм.	Лист	№ докум.	Подпись	Дата
Разраб.		Халамов Н.А.		
Проверил		Иванович В.Д.		

Лит.	Масса	Масштаб
		1:1
Лист 1	Листов 1	
Кафедра информатики гр. 453503		

Приложение В Блок-схема алгоритма нечеткого поиска

