Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт ИнЭО
Направление подготовки Информатика и вычислительная техника
Кафедра Автоматики и компьютерных систем

БАКАЛАВРСКАЯ РАБОТА

Тема работы
Разработка кроссплатформенного клиент-серверного игрового приложения «Найди пару»

УДК 004.4'22

Студент

Группа	ФИО	Подпись	Дата
3-8В12	Григорьев Станислав Сергеевич		

Руководитель

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Суходоев Михаил Сергеевич	К.т.н.		

КОНСУЛЬТАНТЫ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ассистент	Николаенко Валентин Сергеевич	Аспирант		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Профессор	Назаренко Ольга Брониславовна	Д.т.н.		

ДОПУСТИТЬ К ЗАЩИТЕ:

Зав. кафедрой	ФИО	Ученая степень, звание	Подпись	Дата
АиКС	Фадеев Александр Сергеевич	К.т.н.		

Томск 2016 г.

ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЕ ПО ООП

Код результата	Результат обучения (выпускник должен быть готов)
	Профессиональные компетенции
ПК1	Разрабатывать бизнес-планы и технические задания на оснащение отделов, лабораторий, офисов компьютерным и сетевым оборудованием.
ПК2	Осваивать методики использования программных средств для решения практических задач
ПК3	Разрабатывать интерфейсы "человек - электронно-вычислительная машина".
ПК4	Разрабатывать модели компонентов информационных систем, включая модели баз данных.
ПК5	Разрабатывать компоненты программных комплексов и баз данных, использовать современные инструментальные средства и технологии программирования.
ПК6	Обосновывать принимаемые проектные решения, осуществлять постановку и выполнять эксперименты по проверке их корректности и эффективности.
ПК7	Готовить презентации, научно-технические отчеты по результатам выполненной работы, оформлять результаты исследований в виде статей и докладов на научно-технических конференциях.
ПК8	Готовить конспекты и проводить занятия по обучению сотрудников применению программно-методических комплексов, используемых на предприятии.
ПК9	Участвовать в настройке и наладке программно-аппаратных комплексов.
ПК10	Сопрягать аппаратные и программные средства в составе информационных и автоматизированных систем.
ПК11	Инсталлировать программное и аппаратное обеспечение для информационных и автоматизированных систем.

Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт ИнЭО
Направление подготовки Информатика и вычислительная техника
Кафедра Автоматики и компьютерных систем

УТВЕРЖДАЮ:
Зав. кафедрой

_____ _____ _____
(Подпись) (Дата) (Ф.И.О.)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

Бакалаврской работы

Студенту:

Группа	**ФИО**
З-8В12	Григорьев Станислав Сергеевич

Тема работы:

Разработка кроссплатформенного клиент-серверного игрового приложения «Найди пару»

Утверждена приказом директора (дата, номер)	От 15.04.2016, №2917/с

Срок сдачи студентом выполненной работы:	1.06.2016

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе	*1) Язык разработки C#* *2) Среда разработки MS Visual Studio 2015 Enterprise with XAMARIN* *3) .NET Framework 4.5.2* *4) Android SDK API level 20-23*
Перечень подлежащих исследованию, проектированию и разработке вопросов	*1) Обзор технологии кроссплатформенной разработки XAMARIN* *2) Проектирование кроссплатформенной переносной библиотеки классов бизнес логики* *3) Проектирование Android версии приложения* *4) Проектирование Windows версии приложения* *5) Проектирование серверной части* *6) Финансовый менеджмент проекта* *7) Социальная ответственность проекта*
Перечень графического материала	---

Консультанты по разделам выпускной квалификационной работы	
Раздел	**Консультант**
Финансовый менеджмент	Николаенко Валентин Сергеевич, ассистент
Социальная ответственность	Назаренко Ольга Брониславовна, профессор
Названия разделов, которые должны быть написаны на русском и иностранном языках:	

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	30.01.2016

Задание выдал руководитель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Суходоев Михаил Сергеевич	К.т.н.		30.01.2016

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
3-8В12	Григорьев Станислав Сергеевич		30.01.2016

Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт ИнЭО
Направление подготовки (специальность) Информатика и вычислительная техника
Уровень образования Бакалавр
Кафедра АиКС
Период выполнения весенний семестр 2015/2016 учебного года

Форма представления работы:

Бакалаврская работа

(бакалаврская работа, дипломный проект/работа, магистерская диссертация)

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Срок сдачи студентом выполненной работы:	1.06.2016

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
	Введение	
	Обзор литературы	
	Объект и методы исследования	
	Описание предметной области	
	Техническое задание	
	Проектирование	
	Руководство пользователя	
	Финансовый менеджмент	
	Социальная ответственность	

Составил преподаватель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Суходоев Михаил Сергеевич	К.т.н.		

СОГЛАСОВАНО:

Зав. кафедрой	ФИО	Ученая степень, звание	Подпись	Дата
АиКС	Фадеев Александр Сергеевич	К.т.н.		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ»

Студенту:

Группа	ФИО
3-8В12	Григорьев Станислав Сергеевич

Институт	ИнЭО	**Кафедра**	АиКС
Уровень образования	Бакалавр	**Направление/специальность**	Информатика и вычислительная техника

Исходные данные к разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:	
1. *Стоимость ресурсов научного исследования (НИ): материально-технических, энергетических, финансовых, информационных и человеческих*	*Человеческие ресурсы: 1 чел.*
2. *Нормы и нормативы расходования ресурсов*	
3. *Используемая система налогообложения, ставки налогов, отчислений, дисконтирования и кредитования*	
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. *Оценка коммерческого потенциала, перспективности и альтернатив проведения НИ с позиции ресурсоэффективности и ресурсосбережения*	*1) Анализ конкурентов* *2) SWOT - анализ*
2. *Планирование и формирование бюджета научных исследований*	
3. *Определение ресурсной (ресурсосберегающей), финансовой, бюджетной, социальной и экономической эффективности исследования*	
Перечень графического материала:	

Дата выдачи задания для раздела по линейному графику	1.03.2016

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ассистент	Николаенко Валентин Сергеевич	Аспирант		1.03.2016

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
3-8В12	Григорьев Станислав Сергеевич		1.03.2016

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»

Студенту:

Группа	ФИО
3-8В12	Григорьев Станислав Сергеевич

Институт	ИнЭО	**Кафедра**	АиКС
Уровень образования	Бакалавр	**Направление/специальность**	Информатика и вычислительная техника

Исходные данные к разделу «Социальная ответственность»:	
Описание рабочего места (рабочей зоны, технологического процесса, механического оборудования)	*Рабочее место программиста*
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Анализ выявленных вредных факторов проектируемой производственной среды в следующей последовательности: – *физико-химическая природа вредности, её связь с разрабатываемой темой;* – *действие фактора на организм человека;* – *приведение допустимых норм с необходимой размерностью (со ссылкой на соответствующий нормативно-технический документ);* – *предлагаемые средства защиты (сначала коллективной защиты, затем – индивидуальные защитные средства)*	*1) Требования к помещениям при работе за компьютером* *2) Требования к микроклимату, ионному составу и концентрации вредных химических веществ в воздухе помещений* *3) Требования к освещению помещений и рабочих мест* *4) Требования к шуму и вибрации в помещениях* *5) Требования к организации и оборудованию рабочих мест* *6) Режим труда и отдыха при работе с компьютером*
2. Анализ выявленных опасных факторов проектируемой произведённой среды в следующей последовательности – *механические опасности (источники, средства защиты;* – *электробезопасность;* – *пожаровзрывобезопасность*	*Обеспечение электробезопасности и пожарной безопасности на рабочем месте*
Перечень графического материала:	

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Профессор	Назаренко Ольга Брониславовна	Д.т.н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
3-8В12	Григорьев Станислав Сергеевич		

РЕФЕРАТ

Выпускная квалификационная работа 68 с., 19 рис., 3 табл., 16 источников, 1 прил.

Ключевые слова: кроссплатформенность, клиент-серверная архитектура, игровая индустрия

Объектом исследования является: технология кроссплатформенной разработки на базе XAMARIN.

Цель работы: разработка кроссплатформенного клиент-серверного игрового приложения «Найди пару».

В процессе исследования проводились: анализ предметной области, проектирование и разработка переносной библиотеки классов, клиентских версий приложения для Android и Winsows, серверной части.

В результате исследования: разработано клиентское приложение для Android и Windows, серверная часть в виде REST сервиса.

Основные конструктивные, технологические и технико-эксплуатационные характеристики: поддержка платформ Android, Windows, клиент-серверная архитектура

Степень внедрения: планируется вывести на рынок в 4 квартале текущего года

Область применения: игровая сфера деятельности

Экономическая эффективность/значимость работы: одна из второстепенных сфер развития компании, после выхода приложения на рынок планируется повышение доходов компании в целом.

В будущем планируется: расширить функционал, перенести на платформы WinPhone, Windows 10 Mobile, iOS, глобализировать, расширить серверную часть (перенести на несколько серверов).

Оглавление

ВВЕДЕНИЕ 10
1 СРЕДЫ РАЗРАБОТКИ И ТЕХНОЛОГИЯ XAMARIN 11
1.1 Обзор существующих средств для кроссплатформенной разработки 11
1.2 Обоснование выбора среды разработки 14
1.3 Технология кроссплатформенной разработки XAMARIN 14
1.4 Описание предметной области 18
1.5 Техническое задание 18
2 ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЯ «НАЙДИ ПАРУ» 22
2.1 Переносимая библиотека классов PortableLogicLibrary 24
2.2 Серверная часть 28
2.3 Клиентская часть на платформе Android 34
2.4 Клиентская часть на платформе Windows 39
2.5 Результат разработки 40
3 ФИНАНСОВЫЙ МЕНЕДЖМЕНТ 41
3.1 Введение 41
3.2 Анализ конкурентных технических решений 41
3.3 SWOT анализ 44
3.4 Заключение 46
4 СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ 47
4.1 Требования к помещениям при работе за компьютером 47
4.2 Требования к микроклимату, ионному составу и концентрации вредных химических веществ в воздухе помещений 48
4.3 Требования к освещению помещений и рабочих мест 48
4.4 Требования к организации и оборудованию рабочих мест 50
4.5 Режим труда и отдыха при работе с компьютером 52
4.6 Обеспечение электро-пожаробезопасности на рабочем месте 55
ЗАКЛЮЧЕНИЕ 58
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 59
Приложение А 61

ВВЕДЕНИЕ

Ни для кого не секрет, что видео игры прочно заняли свою позицию в современной индустрии развлечений. Существуют попытки выделить компьютерные игры как отдельную область искусства, наряду с театром, кино и т.п. Разработка игр может оказаться не только увлекательным, но и прибыльным делом. А в связи с экспоненциальным ростом количества мобильных устройств и унификации способов разработки приложений для каждого из них, появляется широкая перспектива коммерческой разработки. Рынок мобильных устройств, а соответственно и их пользователей стремительно растет и требуется все больше и больше контента для них: приложения, игры, утилиты, направленные на полное раскрытие потенциала мобильных устройств. Этим и определяется актуальность выбранной темы.

Целью работы является разработка кроссплатформенного клиент-серверного игрового приложения «Найди пару».

Объектом исследования является технология кроссплатформенной разработки на базе XAMARIN, позволяющая разрабатывать кроссплатформенные приложения в среде MS Visual Studio на языке C# и компилировать их в нативный исполняемый код для каждой платформы.

Предметом исследования является проектирование переносной кроссплатформенной библиотеки классов, разработка серверной части, интерфейса на платформах Windows и Android.

Практическая новизна: исследуемая технология совершенно новая и стремительно развивающаяся, с каждым месяцем привлекающая все большее количество разработчиков. Тем самым приближая и объединяя множество существующих платформ воедино для удобной, быстрой и качественной разработки приложений.

Практическая значимость результата ВКР: разработанное приложение можно использовать в коммерческих целях, как продажи, так и заработка на рекламе (в перспективе роста количества пользователей).

1 СРЕДЫ РАЗРАБОТКИ И ТЕХНОЛОГИЯ XAMARIN

1.1 Обзор существующих средств для кроссплатформенной разработки

1.1.1 Интегрированная среда разработки Eclipse

Eclipse – это расширяемая, open-source интегрированная среда разработки. Совместно с Java Development Tools, платформа Eclipse предоставляют множество различных возможностей, которые можно наблюдать в коммерческих средах разработки: подсветка синтаксиса в редакторе, компиляция кода, отладчик уровня исходного кода с поддержкой "нитей" (threads), навигатор по классам, файловый менеджер и менеджер проектов, интерфейсы для стандартных контролирующих систем исходного кода, таких как, например, CVS и ClearCase. Помимо этого Eclipse содержит ряд уникальных возможностей, например, рефакторинг кода, автоматическое обновление и сборка кода, список задач, поддержка возможности тестирования модулей с помощью JUnit, а также интеграция с инструментом сборки приложений Jakarta Ant.[1,2]

Несмотря на большое количество стандартного набора возможностей, Eclipse отличается от традиционных сред разработки по нескольким фундаментальным особенностям. Самая интересная возможность Eclipse – это абсолютная нейтральность относительно платформы и языка программирования. Вдобавок к эклектичному набору языков программирования, которые поддерживаются Eclipse Consortium (Java, C/C++, Cobol), существует множество сторонних проектов, с помощью которых можно обеспечить поддержку интересующего языка программирования в Eclipse. На сегодняшний день существуют реализации следующих популярных языков программирования: Python, Eiffel, PHP, Ruby, и C#.[1,2]

Платформа Eclipse предоставляется, благодаря Eclipse Consortium, в виде заранее скомпилированных исполняемых файлов для Windows, Linux, Solaris, HP-UX, AIX, QNX, и Mac OS X. Очень много внимания концентрируется вокруг архитектурной системы plug-in'ов этой платформы, а также

"богатых" API (Application Programming Interface), поставляемых с Plug-in Development Environment для расширения Eclipse. Добавить поддержку нового типа редактора или языка программирования очень просто, благодаря хорошо спроектированным API и строительным блокам, которые предоставляет Eclipse.[1,2]

1.1.2 Интегрированная среда разработки NetBeans

NetBeans является официальной средой разработки для Java 8, предоставляет все средства, необходимые для создания профессиональных десктоп приложений, корпоративных, мобильных и веб-приложений. С помощью его редакторов, анализаторов и преобразователей кода, разработчик может легко и быстро обновлять существующие приложения, чтобы использовать новые языковые конструкции Java 8, таких как лямбда-выражений, функциональных операций, и т.д..[3,4]

Пакетные анализаторы и преобразователи кода позволяют выполнять поиск по нескольким приложениям одновременно, сравнивая существующие структуры кода для преобразования их в новые языковые конструкции Java 8.[3,4]

Рабочая область среды является полностью настраиваемой - существует возможность пользовательской настройки действий, выполняемых с помощью панели, назначения "горячих" клавиш и т.д. Так же среда имеет в своем составе расширенный многоязыковой редактор для различных языков программирования - Java, C/C++, Ruby, Groovy, PHP, JavaScript, CSS, XML, HTML, RHTML, JSP, документацию Javadoc. Существует возможность расширения функций редактора с целью поддержки любого другого языка.[3,4]

Редактор NetBeans делает отступы строк, проверяет соответствие скобок и слов, подсвечивает синтаксис исходного кода. Производится проверка ошибок во время ввода, отображение вариантов для автозавершения кода и фрагментов документации по требуемому языку программирования. Для ускорения разработки среда может генерировать и вставлять в исходный код стандартные фрагменты кода на Java или других языках.[3,4]

1.1.3 Интегрированная среда разработки Visual Studio 2015 Enterprise

Visual Studio 2015 Enterprise – это интегрированная среда разработки, которая обладает всем необходимым для создания отличных приложений для мобильных устройств, настольных приложений, веб-приложений и облачных решений. Позволяет писать код для iOS, Android и Windows в одной интегрированной среде разработки. Предоставляет доступ к удобной и функциональной среде IntelliSense, преимуществам простой навигации по коду, быстрой сборки и развертывания в кратчайшие сроки. [5,9]

С помощью инструментов диагностики Visual Studio 2015 Enterprise и IntelliTrace можно просматривать журнал выполнения кода и переходить к проверяемому состоянию без задания точек останова вручную. Это сэкономит немало времени, которое целесообразно потратить более продуктивно.[5]

Visual Studio 2015 Enterprise упрощает интеграцию нагрузочного тестирования в процесс разработки и помогает избежать опасных сюрпризов в рабочей среде. Причиной этих сюрпризов может быть распространение по всему миру, масштаб базы клиентов или проблемы, проявляющиеся только при многодневных циклах выполнения, — нагрузочное тестирование Visual Studio 2015 предоставит аналитические данные для устранения этих проблем еще до развертывания.[5]

Visual Studio 2015 Enterprise включает функции Xamarin для разработки приложений для Android, iOS и Windows. Позволяет создавать приложения с производительностью машинного кода и машинным пользовательским интерфейсом с помощью инструментов корпоративного уровня. Облегчает оптимизацию производительности приложения с помощью данных профилирования, а так же исследование их в среде выполнения для быстрого поиска ошибок. Позволяет проверить качество разработки на настоящих устройствах в тестовом облаке Xamarin.[5,7]

1.2 Обоснование выбора среды разработки

Поскольку основным изучаемым языком программирования в процессе обучения является C#, а его «родная» среда разработки - Visual Studio и платформа .NET, средой разработки следует выбрать VisualStudio 2015 Enterprise с расширением XAMARIN. Это позволит использовать всю мощь платформы .NET в сочетании с новыми встроенными инструментами и библиотеками для кроссплатформенной разработки для всех существующих платформ (в том числе универсальных приложений для Windows 10 и Windows Phone), обеспечит удобство разработки при использовании привычных инструментов и вспомогательных утилит. Помимо этого выбор обусловлен наличием ученической лицензией по подписке DreamSpark для студентов высших учебных заведений.

Интегрированная среда разработки Eclipse является первоначально направленной на разработку на Java, C/C++, Cobol, хоть и позволяет разрабатывать приложения на языке C#, подключив специальное расширение, она все равно уступает по возможностям VisualStudio 2015 Enterprise с расширением XAMARIN.

Интегрированная среда разработки NetBeans является родной средой для разработки на Java и основной акцент в ней сделан на разработку именно на этом языке, так же по умолчанию среда не поддерживает язык C#.

1.3 Технология кроссплатформенной разработки XAMARIN

1.3.1 Объект исследования. Объектом в данной работе является технология кроссплатформенной разработки на базе XAMARIN.

Xamarin — это фреймворк для кроссплатформенной разработки мобильных приложений (iOS, Android, Windows Phone) с использованием языка C#. Идея технологии такова: дать возможность .NET разработчику писать код на привычном языке, с применением всех привычных языковых особен-

ностей LINQ, лямбда-выражений, Generic`ов и async`ов. При этом обеспечить ему полный доступ ко всем возможностям SDK платформы и родному механизму создания пользовательского интерфейса, получая на выходе приложение, которое, строго говоря, ничем не отличается от нативных и не уступает им в производительности.[6,7]

Xamarin основан на open-source реализации платформы .NET — Mono. Эта реализация включает в себя собственный компилятор C#, среду выполнения, а так же основные .NET библиотеки. Цель проекта — позволить запускать программы, написанные на C#, на операционных системах, отличных от Windows — Unix-системах, Mac OS и других. [6,7]

С точки зрения исполнения приложений между iOS и Android есть одно ключевое различие — способ их предварительной компиляции. Как известно, для выполнения приложений в Android используется виртуальная Java-машина Dalvik. Нативные приложения, которые пишутся на Java, компилируются в некий промежуточный байт-код, который интерпретируется Dalvik`ом в команды процессора в момент исполнения программы (т.е. аналогично тому, как работает CLR в .NET). Это так называемая Just-in-time компиляция (компиляция на лету). В iOS используется другая модель компиляции — Ahead-of-Time (компиляция перед исполнением). Xamarin учитывает это различие, предоставляя отдельные компиляторы для каждой из этих платформ, которые позволяют на выходе получать настоящие, нативные приложения, которые выполняются вне контекста браузера и могут использовать все аппаратные и программные ресурсы платформы.[6,7]

Для iOS ситуация простая — никакой виртуальной машины нет и программный код должен быть просто заранее скомпилирован в машинный. Для этой цели используется AOT компилятор Mono.[6,7]

Для Android ситуация выглядит сложнее. При компиляции приложения происходит перевод кода на C# в промежуточный байт-код, понятный виртуальной машине Mono и сама эта виртуальная машина также добавляется в упакованное приложение. И Mono и Dalvik написаны на языке C и работают

поверх ядра Linux (Android основана на Linux). При запуске приложения на Android обе виртуальные машины начинают работать бок о бок и обмениваются данными через специальный механизм.[6,7,8]

Для каждой платформы Xamarin предоставляет возможность использовать нативные средства разработки пользовательского интерфейса и нативные элементы пользовательского интерфейса. Для Android создание пользовательского интерфейса может происходить непосредственно в коде или же при помощи декларативного подхода с описанием интерфейса в XML. Для iOS это также либо код, либо использование нативных средств проектирования интерфейса — отдельные xib-файлы или же один большой Storyboard. Редактирование этих файлов происходит в привычной для iOS-разработчика среде X-CODE. И это означает, что для этого необходимо обладать Mac. Для разработки iOS-приложений в любом случае потребуется Mac по двум причинам:

Во-первых, для редактирования пользовательского интерфейса в среде X-CODE. Во-вторых, для отладки приложений требуется симулятор iPhone/iPad, который также доступен только на Mac.

Разработчики Xamarin в качестве среды разработки предлагают использовать либо свою собственную — Xamarin Studio, либо Visual Studio.[6,7,8]

1.3.2 Методы исследования. Для исследования объекта используются следующие методы:

1) обзор существующих целевых платформ
2) обзор методов портирования приложения на разные платформы

Эти методы являются основными, поскольку исследуемая технология направленна именно на кроссплатформенную разработку на языке C#. Необходимо выбрать целевые платформы, под какие будет производиться разработка, а так же возможные методы переноса разработанного приложения на другие платформы в будущей перспективе. Поскольку разрабатываемое приложение должно иметь архитектуру клиент-сервер, это позволит более глу-

боко погрузится в технологию при разработке, в частности изучить ее возможности для работы с сетью.

В качестве целевых платформ следует обратить внимание на Windows и Android. Поскольку они наиболее распространены и для написания приложения для них, не требуется приобретать дополнительное оборудование и лицензию. Как например, для того чтобы скомпилировать приложения для iOS, необходимо обладать MAC совместимым компьютером и предустановленной на нем среде разработки X-CODE, что в свою очередь требует ощутимых материальных затрат. А для разработки для Windows и Android, достаточно иметь компьютер под управлением ОС Windows и предустановленной средой разработки VisualStudio, ученической лицензии для которых достаточно, чтобы решить все поставленные задачи выпускной квалификационной работы.

Методы портирвоания в основном определяет архитектура приложения. Самые распространённые методы проектирования приложения с общим кодом: решение с общим проектом и решение с переносной библиотекой классов.

Решение с общим проектом подразумевает реализацию в одном решении проектов интерфейса, поведения, моделей данных и т.д. Для портирования такого приложения нужно добавить в решение проект интерфейса требуемой целевой платформы и используя реализованные модели данных и поведения реализовать новую версию приложения.

Решение с переносной библиотекой классов подразумевает отделение моделей данных и поведения в отдельную динамическую библиотеку и в дальнейшем подключать ее к новым реализациям под конкретные платформы.

Самым оптимальным вариантом для данной разработки будет использовать решение с переносной библиотекой классов, поскольку она будет содержать модели данных и поведения, используемые на сервере и клиентах, а так же обеспечит простоту портирования под другие платформы.

1.4 Описание предметной области

Разрабатываемая игра направлена на развитие памяти и мелкой моторики пальцев. Для этого игроку предлагается сопоставить ряд картинок по парам.

Игровое поле представляет собой 16 и 36 клеток для нормального и тяжелого уровня сложности соответственно. В каждой клетке случайным образом расположена одна из 8 (нормальный уровень сложности) и 18 (тяжелый уровень сложности) пар картинок. Каждая картинка скрыта от игрока. Игроку предстоит открыть все клетки с картинками как можно быстрее.

Игровой процесс представляется следующим образом: игрок выбирает клетку, ему показывается скрытое изображение, затем выбирает вторую клетку и если изображения на обеих клетках совпадают, то они остаются открытыми, и игрок выбирает дальше. Если изображения не совпадают, то через некоторое время они скрываются и игроку предстоит вновь искать пары.

Игра заканчивается, как только игрок откроет все клетки с картинками. По результату время прохождения заносится в таблицу рекордов.

1.5 Техническое задание

1.5.1 Введение

1.5.1.1 Наименование программы

Наименование игры – «Найди пары».

1.5.1.2 Краткая характеристика области применения

Программа предназначена для осуществления игровой деятельности в свободное время на различных устройствах.

1.5.1.3 Цели и сроки разработки

Целью разработки является совершенствование практических навыков разработки кроссплатформенных клиент-серверных приложений. Сроки разработки: 2 квартал 2016 года.

1.5.2 Основания для разработки

1.5.2.1 Основание для проведения разработки

Основанием для проведения разработки является задание на разработку игрового приложения, в рамках подготовки выпускной квалификационной работы на соискание степени бакалавра. Являющееся частью учебного курса для студентов ТПУ кафедры АИКС обучающихся по направлению информатика и вычислительная техника.

1.5.2.2 Наименование и условное обозначение темы разработки

Наименование темы разработки – «Разработка игры «Найди пары»». Условное обозначение темы разработки (шифр темы) – «НП-007».

1.5.3 Требования к программе или программному изделию

1.5.3.1 Требования к игровому полю

Для нормального уровня сложности игровое поле состоит из 16 клеток, в виде квадрата со стороной 4 клетки. Каждая клетка хранит одну из 8 пар картинок.

Для тяжелого уровня сложности игровое поле состоит из 36 клеток, в виде квадрата со стороной 6 клеток. Каждая клетка хранит одну из 18 пар картинок.

По умолчанию картинки скрыты от игрока.

1.5.3.2 Требования к составу выполняемых функций

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

1) функция автоматической расстановки картинок на игровом поле;
2) функция выбора сложности игры: нормальный и тяжелый уровень;

3) функция сохранения результатов игры в файле;

4) функция отображения ранжированных результатов предыдущих игр;

5) функция сохранения результатов игры в глобальном хранилище на сервере;

6) функция получения результатов других игроков с сервера.

1.5.3.3 Требования к логике игры

Игра должна функционировать по следующим правилам:

1) открыть одновременно можно не более двух клеток;

2) клетки открываются кликом (касанием) по клетке;

3) если открытые клетки имеют разные картинки, то через 500 мс они скрывают свои картинки;

4) если открытые клетки имеют одинаковые картинки, то они остаются открытыми до конца игры;

5) пока не скроются две открытые различные картинки, открывать другие нельзя;

6) игра заканчивается после открытия всех клеток.

1.5.3.4 Требования к целевой платформе развертывания

Основная логика игры и клиентская часть взаимодействия с сервером должна быть совместимой со следующими платформами Windows, Windows Phone, Android, iOS. Интерфейс разрабатывается с учетом особенностей каждой платформы индивидуально.

1.5.3.5 Требования к версии платформы

Для Windows – версия 10, минимальная сборка 10240;

Для Android – версия 5.0.1;

Для Windows Phone – версия 8.1;

Для iOS – версия 7.

1.5.3.6 Требования к серверу рекордов

Сервер должен представлять собой REST-сервис, который обеспечивает доступ клиентов к глобальному хранилищу рекордов. Предоставляет возможность сохранения своего рекорда, а так же получения сведений о рекордах других игроков.

1.5.3.7 Требования к интерфейсу пользователя

Программа должна обеспечивать взаимодействие с пользователем (оператором) посредством графического пользовательского интерфейса, разработанного согласно рекомендациям компании-производителя операционной системы.

1.5.4 Требования к составу и параметрам технических средств

Устройство, на котором планируется выполнить развертывание данного приложения, должно соответствовать техническим требованиям, предъявляемым производителем ОС к аппаратной части технического средства.

1.5.5 Требования к исходным кодам и языкам программирования

Исходные коды программы должны быть реализованы на языке C#. В качестве интегрированной среды разработки программы должна быть использована среда MS Visual Studio 2015 Enterprise с XAMARIN.

1.5.6 Требования к программным средствам, используемым программой

На компьютерах под управлением Windows 10 необходимо обязательное наличие предустановленного пакета MS .NET FrameWork не ниже версии 4.5.

2 ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЯ «НАЙДИ ПАРУ»

Кроссплатформенная разработка подразумевает создание приложения для выполнения на нескольких платформах. Поскольку бизнес-логика приложения не должна отличаться от конкретной платформы, то при переносе приложения с одной платформы на другую, будет происходить дублирование большого количества кода, за исключением реализации интерфейса и некоторых других особенностях работы с данными, таймерами и др. (разные пространства имен, разные имена классов, различный механизм выполнения задержек и пр.). Поэтому целесообразно отделить бизнес-логику от интерфейса, тем самым облегчая разработку и дальнейшую поддержку приложения. При каких либо изменениях и дополнениях в бизнес-логике не нужно исправлять все версии приложения для каждой платформы, а достаточно будет скорректировать код в библиотеке бизнес-логики.

Для реализации бизнес-логики целесообразно использовать переносимую библиотеку классов. Она позволяет использовать возможности, одинаково доступные на всех целевых платформах и не позволяет использовать какие либо нативные функции конкретной платформы. Тем самым просто используя переносимую библиотеку классов с первоначальным указанием целевых платформ, уже обеспечивается ограниченный круг доступных классов, на основе которых реализуется вся бизнес-логика. Так же переносимая библиотека классов очень удобна при разработке кроссплатформенных приложений, по мере необходимости выпуска приложения под очередную целевую платформу достаточно подключить библиотеку и реализовать интерфейс.

Поскольку разрабатываемое приложение является кроссплатформенным с архитектурой клиент-сервер, целесообразно вынести в переносимую библиотеку классов не только бизнес-логику, но и модель данных, а так же реализацию процесса взаимодействия с сервером. В целом структуру взаимодействия всех компонентов отражает UML диаграмма развертывания, в соответствии с рисунком 1.

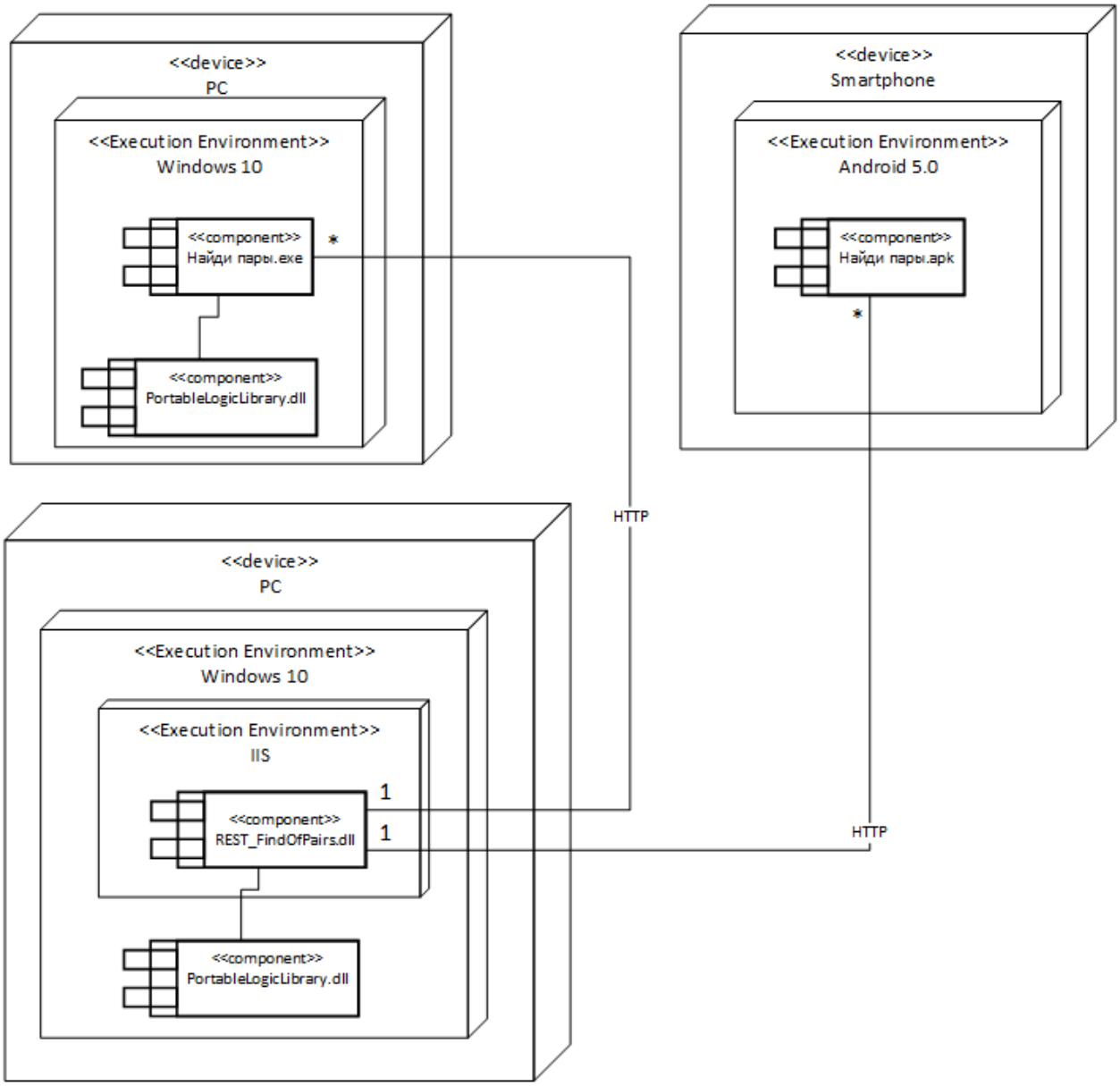

Рисунок 1 – UML диаграмма развертывания

В роли сервера выступает REST сервис, который позволяет хранить и управлять глобальным списком рекордов всех игроков. Клиент, используя API методы сервиса, может получить все установленные рекорды, получить список рекордов определенной сложности, отправить свой рекорд.

В роли клиентов выступают клиентские версии игры для платформ Windows и Android. На каждом устройстве используется переносимая библиотека классов PortableLogicLibrary.dll, которая в Windows является отдельным файлом и подгружается во время исполнения, а в Android она внедряется в исполняемый файл при компиляции.

2.1 Переносимая библиотека классов PortableLogicLibrary

Библиотека PortableLogicLibrary.dll включает в себя три пространства имен:

1) Server – содержит классы для взаимодействия с сервером;
2) Score – содержит классы, описывающие модель данных результатов игры и процедуру их подсчета;
3) Logic – содержит классы логики игры.

2.1.1 Пространство имен **Server**

Содержит один класс RestClient, который реализует все необходимые методы для доступа к REST сервису с глобальным хранилищем рекордов. Класс включает два свойства и пять методов:

1) public int KeyAccess – свойство возвращает целочисленное значение ключа доступа к серверу;
2) public string ServerURI – свойство возвращает строку URI сервера;
3) public async Task<Records> GetRecords() – асинхронный метод возвращает объект Records, содержащий все глобальные рекорды;

4) public async Task<Records> GetRecords(int id) – асинхронный метод возвращает объект Records, содержащий все рекорды игрока с номером **id**;

5) public async Task<IEnumerable<Player>> GetRecordsAsDifficulty(Difficulty dif) – асинхронный метод возвращает объект <IEnumerable<Player>, содержащий список рекордов заданной сложности **dif**;

6) public async Task<bool> PostPlayer(Difficulty dif, Player player) – асинхронный метод отправляет объект **player,** содержащий сведения об установленном рекорде, параметр **dif** определяет, на какой сложности был установлен рекорд. Метод возвращает **true**, если рекорд успешно был отправлен на сервер, и **false**, если отправка потерпела неудачу;

7) public async Task<bool> Delete(Difficulty dif, Player player) – асинхронный метод удаляет с сервера рекорд **player** из списка рекордов в зависимости от указанной сложности **dif**, и возвращает true в случае успеха и false в случае неудачи.

2.1.2 Пространство имен **Score**

Включает в себя три класса Records, Player, CalculateRecord и перечисление Difficulty.

Классы Records и Player определяют модель данных. Первый хранит все рекорды, разбитые по сложности, а второй определяет игрока, установившего рекорд. Класс Records имеет четыре свойства:

1) public List<Player> Normal – возвращает список рекордов нормального уровня сложности, только для чтения. Так же свойство отмечено атрибутом DataMember, указывающим, что свойство подлежит сериализации;

2) public List<Player> Hard - возвращает список рекордов тяжелого уровня сложности, только для чтения. Так же свойство отмечено атрибутом DataMember, указывающим, что свойство подлежит сериализации;

3) public int CountRecNormal – возвращает количество рекордов нормального уровня сложности;

4) public int CountRecHard – возвращает количество рекордов тяжелого уровня сложности.

Так же класс Records имеет два метода:

1) public void AddRecordNormal(string name, TimeSpan time) – метод добавляет новый рекорд с именем игрока **name** и результатом игры **time** в список рекордов нормального уровня сложности, а так же сортирует список по возрастанию времени прохождения уровня;

2) public void AddRecordHard(string name, TimeSpan time) – метод добавляет новый рекорд с именем игрока **name** и результатом игры **time** в список рекордов тяжелого уровня сложности, а так же сортирует список по возрастанию времени прохождения уровня.

Класс Player имеет три свойства, отмеченных атрибутом DataMember, указывающим, что они подлежат сериализации:

1) public int Id – возвращает уникальный идентификатор игрока;
2) public string Name – возвращает имя игрока;
3) public TimeSpan Time – возвращает время прохождения уровня.

Перечисление Difficulty содержит два поля: **Normal** и **Hard**, определяющие сложность игры нормальную и тяжелую соответственно. Так же отмечены атрибутом DataMember, указывающим, что они подлежат сериализации.

Ввиду того что классы Record и Player является частью модели данных, они должны быть сериализуемыми. Так же сериализуемым должно быть и перечисление Difficulty, поскольку его необходимо передавать в HTTP запросе к серверу, а значит, оно будет сериализовано. А поскольку все они расположены в переносимой библиотеке классов, это накладывает некоторые ограничения на возможности сериализации. Сериализация для выбранных платформ осуществляется с использованием механизма контрактов данных. Поэтому необходимые для сериализации классы и перечисления должны быть отмечены соответствующим атрибутом DataContract.

Класс CalculateRecords предназначен для подсчета времени прохождения уровня и выявления входит ли результат игры в десятку лучших рекордов в зависимости от сложности. Содержит одно свойство и три метода:

1) public bool IsRunning – возвращает **true** если таймер подсчета времени игры запущен, и **false** если нет;

2) public void Start() – запускает таймер для начала подсчета времени игры;

3) public bool Stop(Records rec, Difficulty dif, out TimeSpan time) – останавливает таймер и проверяет, входит ли результат в десятку лучших результатов **rec** в зависимости от сложности **dif**. Если входит, то возвращает значение **true**, если нет – **false**. В параметр **time** передается время, затраченное на прохождение уровня;

4) public void Reset() – сбрасывает таймер в начальное состояние.

2.1.3 Пространство имен **Logic**

Содержит один класс GameBoard, который представляет собой игровое поле. Содержит одно свойство и один метод:

1) public List<int> Cells – содержит номера ячеек поля и код картинки в ней;

2) public void Refresh() – перемешивает коды картинок между ячейками.

Конструктор GameBoard(int side) принимает в качестве параметра сторону игрового поля **side**, и в зависимости от нее, создает нужное количество ячеек и заполняет их кодами картинок. Максимальный размер поля 10 на 10 клеток. Если входной параметр **side** находится за пределами диапазона 1 – 10, то выбрасывается исключение. Конструктор класса без параметров инициализирует игровое поле со стороной две клетки.

UML диаграмма классов для переносимой библиотеки представлена на рисунке 2.

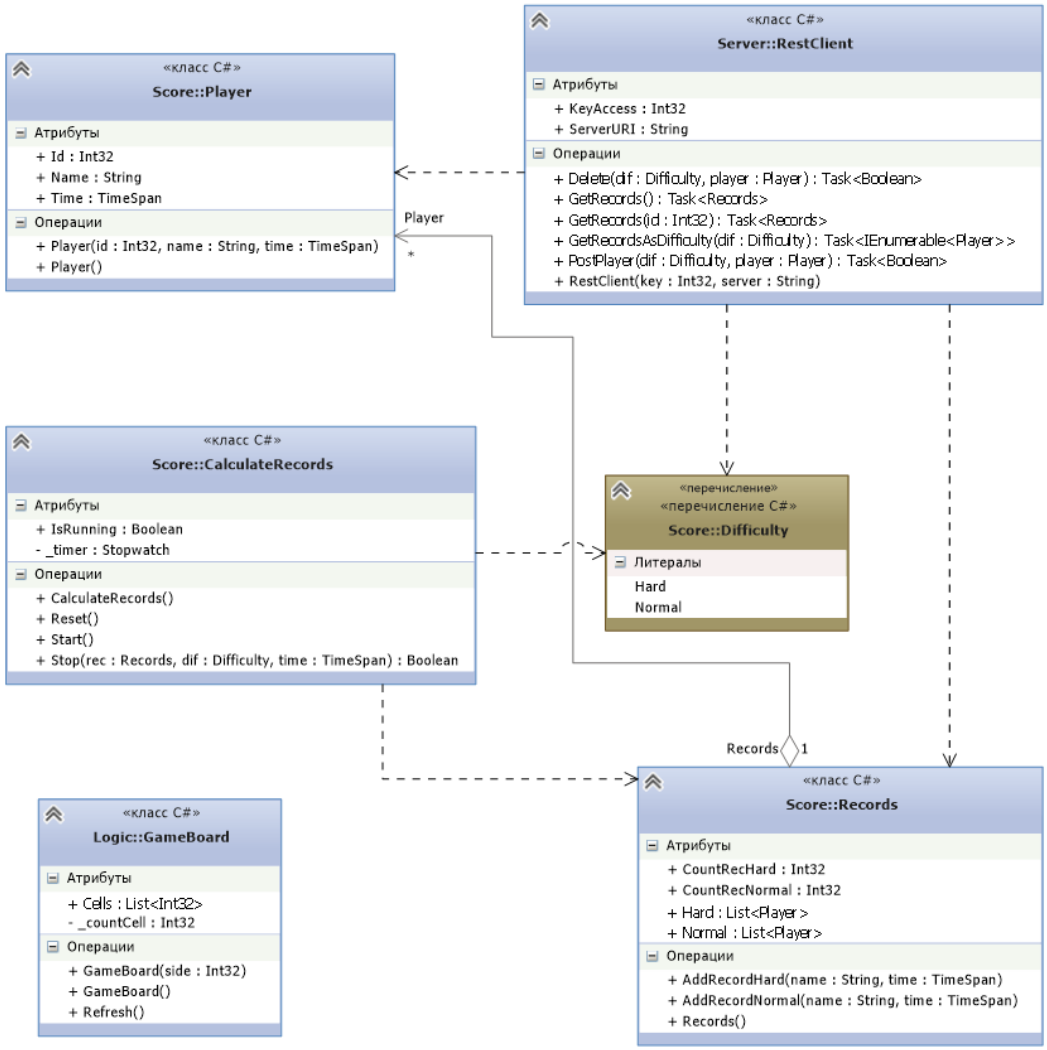

Рисунок 2 – UML диаграмма классов PortableLogicLibrary

2.2 Серверная часть

Серверная часть представляет собой REST сервис, который имеет сложную внутреннюю архитектуру, включает в себя 3 компонентные части. UML диаграмма пакетов REST сервиса приведена на рисунке 3.

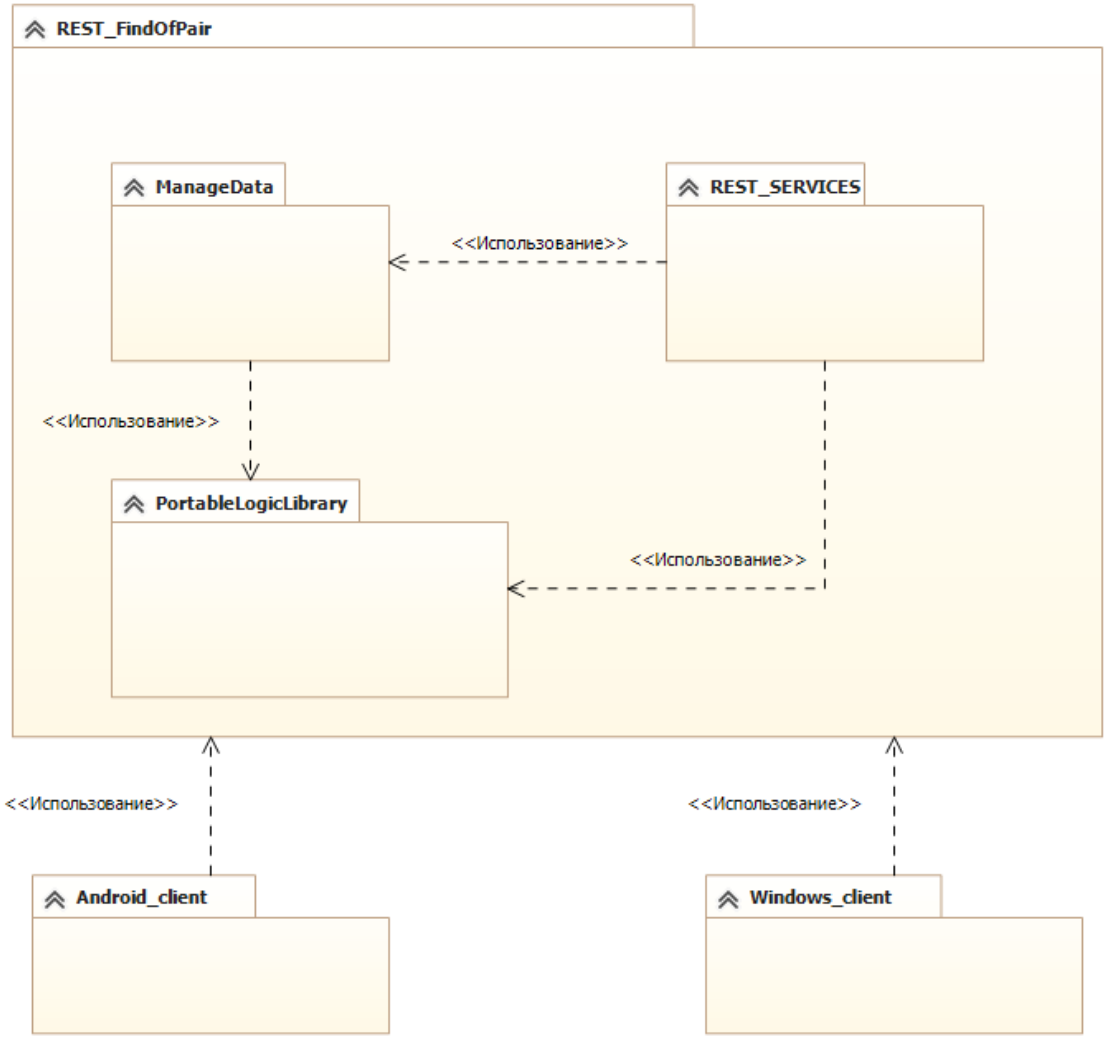

Рисунок 3 – UML диаграмма пакетов REST сервиса

Первая часть это модель данных, которая реализована в переносной библиотеке классов PortableLogicLibrary.

Вторая часть это методы доступа к данным ManageData. Содержит интерфейс IRecordsRepository и класс RecordsRepository.

Интерфейс IRecordsRepository включает в себя пять методов доступа к данным:

1) Records GetAll() – метод возвращает все рекорды;

2) Records Get(int id) – метод возвращает все рекорды игрока с идентификатором **id**;

3) void Post(Player player, Difficulty dif) – метод добавляет рекорд игрока **player** в список рекордов заданной сложности **dif**;

4) bool Delete(Player player, Difficulty dif) метод удаляет рекорд игрока **player** из списка рекордов заданной сложности **dif**, и в зависимости от успеха/неудачи операции удаления возвращает true/false;

5) IEnumerable<Player> GetNormal() – метод возвращает список рекордов нормального уровня сложности;

6) IEnumerable<Player> GetHard() – метод возвращает список рекордов тяжелого уровня сложности.

Класс RecordsRepository наследует интерфейс IRecordsRepository и соответственно реализует все требуемые методы интерфейса. Модель данных получает из библиотеки PortableLogicLibrary. Для хранения рекордов на стороне сервера используется сериализация списка отзывов в формате JSON. При создании экземпляра класса из файла десериализуется список рекордов и дальнейшая работа ведется с ним. При добавлении, удалении происходит сохранения каждого изменения в сериализованном файле.

Третья часть это методы API. Реализуется в виде библиотеки DLL. Основой библиотеки является класс WorkController, который предоставляет методы API для реализации CRUD операций:

1) Records Get(int keyAccess) – получить все рекорды с сервера;

2) Records Get(int keyAccess, int id) – получить все рекорды с сервера с идентификатором игрока **id**;

3) IEnumerable<Player> Get(int keyAccess, Difficulty dif) – получить все рекорды заданной сложности **dif**;

4) void Post(int keyAccess, Player player, Difficulty dif) – добавить игрока **player** в список рекордов заданной сложности **dif**;

5) void Delete(int keyAccess, Player player, Difficulty dif) – удалить рекорд игрока **player** из списка рекордов заданной сложности **dif**.

Отдельно следует отметить назначение параметра **keyAccess** в каждом методе API. Поскольку серверная часть реализована в виде REST сервиса, то доступ к его API возможен не только с игровых клиентов, но и с браузеров и других сетевых программ. Тем самым потенциально возникает уязвимость сервиса к несанкционированному доступу к данным, например, составив HTTP запрос с заданными параметрами можно отослать на сервер заведомо ложный результат или удалить существующий. Способ авторизации через регистрацию аккаунтов тоже не подходит, поскольку точно так же можно с браузера зарегистрировать фиктивный аккаунт и с его помощью получить доступ к данным. Поэтому решением является введение дополнительного параметра **keyAccess,** который содержит особую комбинацию цифр, являющуюся ключом доступа, известный только разработчику клиентских приложений и сервера. Этот ключ зашит в исполняемый код, и злоумышленнику извлечь его будет достаточно проблематично, как и просто подобрать, ввиду его сложности – 10 знаков. Для того чтобы злоумышленник не перехватил ключ доступа при его передаче серверу, ключ доступа хешируется и при отправке запроса к серверу клиент передает хеш ключа доступа, который сравнивается с хешем на стороне сервера и если они совпадают, то сервер выполняет запрошенную операцию.

Через интерфейс IRecordsRepository класс WorkController получает доступ к данным.

UML диаграмма классов REST_FindOfPairs приведена на рисунке 4.

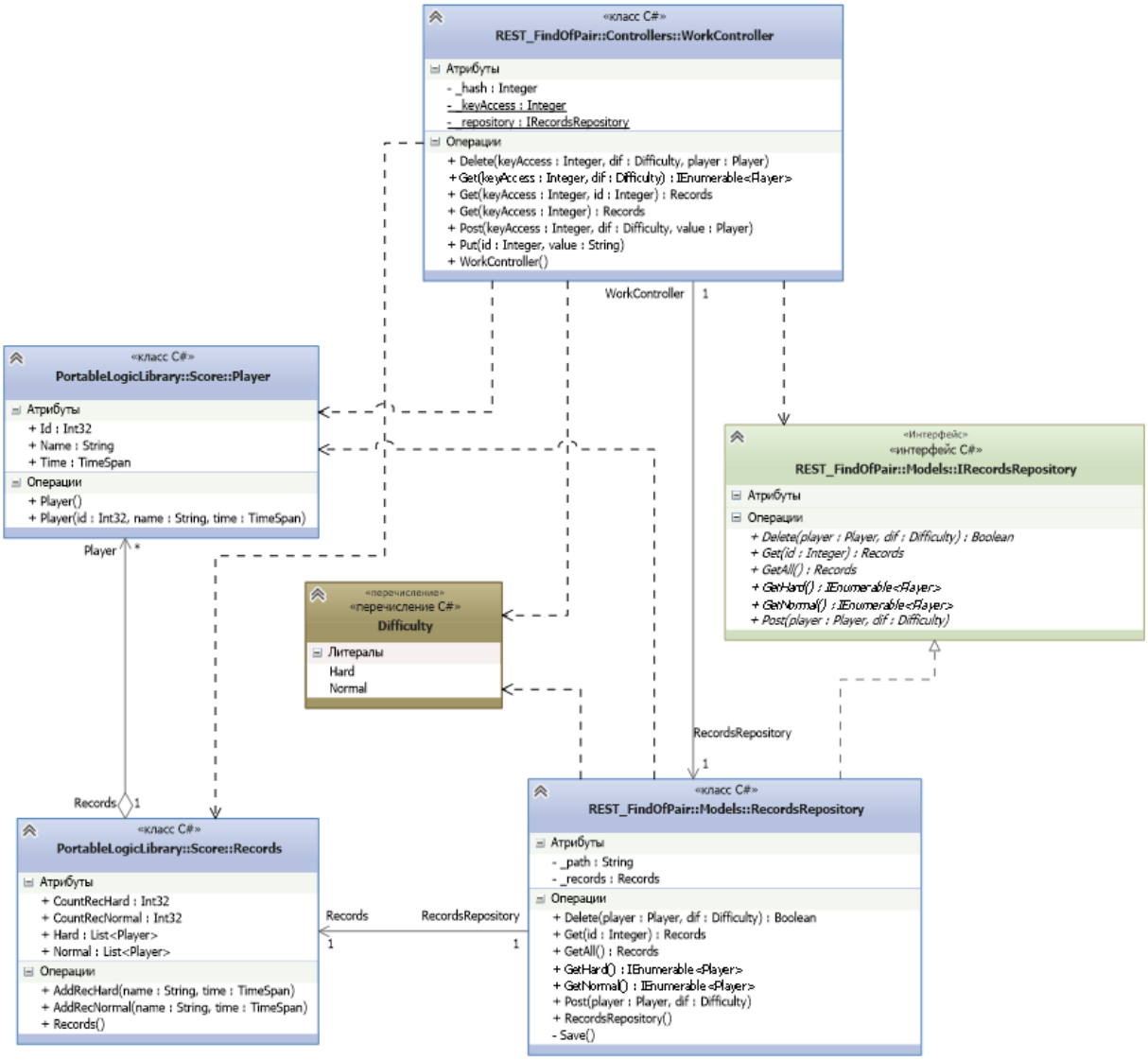

Рисунок 4 - UML диаграмма классов REST_FindOfPairs

UML диаграмма последовательностей REST_FindOfPairs приведена на рисунке 5.

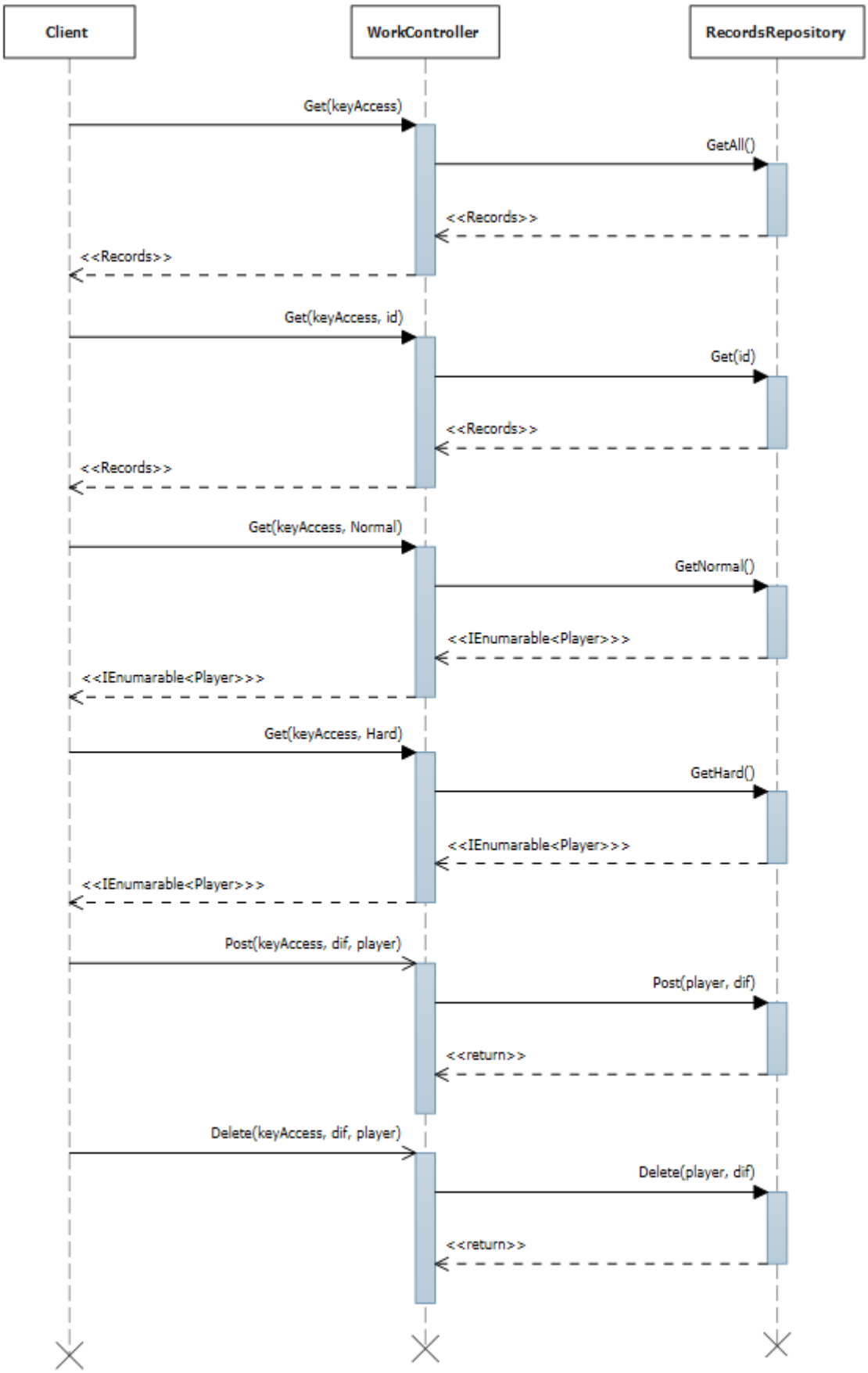

Рисунок 5 - UML диаграмма последовательностей REST_FindOfPairs

2.3 Клиентская часть на платформе Android

При проектировании приложений для Android необходимо учитывать некоторые особенности платформы и жизненного цикла приложения. Экран приложения представляет собой связку класса Activity и интерфейса на языке XAML. Тем самым интерфейс отделен от логики приложения и не имеет в себе никаких функций, а хранит лишь разметку экрана и описание элементов управления на нем. Для каждого экрана при инициализации необходимо указать слой с разметкой, которую будет иметь отрисованный экран. В процессе работы при необходимости смены экрана можно подменять этот слой, тем самым менять разметку на экране. В этом случае необходима всего один экземпляр класса Activity, но это не самое лучшее решение в проектировании приложения в целом, потому что код теряет структурированность, становится более запутанным и громоздким. Однако для исключительных ситуаций это будет разумным способом смены экрана.

Лучшим решением является создание для каждого экрана свой слой разметки и свой экземпляр класса Activity, который содержит требуемый функционал для данного экрана.

Всего в приложении предусмотрено семь экранов, соответственно семь разметок XAML:

1) главный экран – содержит кнопки навигации: играть, настройки, локальные рекорды, глобальные рекорды, правила игры;

2) игровой экран нормального уровня сложности – содержит игровое поле в виде таблицы 4 на 4 клетки (клетка представлена элементом управления ImageView), ниже расположены две кнопки: заново и назад;

3) игровой экран тяжелого уровня сложности - содержит игровое поле в виде таблицы 6 на 6 клеток (клетка представлена элементом управления ImageView), ниже расположены две кнопки: заново и назад;

4) экран локальных рекордов – содержит две кнопки: нормальный и сложный, каждая из которых отображает соответствующую таблицу с рекордами для устройства;

5) экран глобальных рекордов - содержит две кнопки: нормальный и сложный, каждая из которых отображает соответствующую таблицу с глобальными рекордами;

6) экран настроек – содержит элемент управления RadioGroup, позволяющий выбрать сложность игры;

7) экран справки – отображает правила игры.

Но классов с логикой экранов всего шесть, каждый из них наследуется от базового класса Activity:

1) MainActivity – реализует логику работы главного экрана приложения;

2) GameActivity – реализует логику работы непосредственно игрового процесса для обоих уровней сложности;

3) RecordGlobalActivity – реализует логику работы экрана глобальных рекордов;

4) RecordLocalActivity - реализует логику работы экрана локальных рекордов;

5) RulesActivity – реализует логику работы экрана с правилами игры;

6) SettingsActivity – реализует логику работы экрана настроек.

Так же для реализации всех функций необходимо четыре класса:

1) DataHelper – вспомогательный класс который обеспечивает передачу данных между экземплярами экранов и содержит ряд методов для обработки и подготовки к представлению данных;

2) AddRecord – класс, обеспечивает реализацию окна уведомления о новом рекорде с возможностью сохранения на сервере;

3) NotifyEndGame – класс, обеспечивает реализацию окна уведомления об окончании игры, когда рекорд не был установлен;

4) RecordListAdapter – класс, обеспечивает реализацию адаптера, который создает особые элементы управления для отображения строки с результатом игры в заданном виде и организует их в виде списка.

Карта кода приложения для Android, отражающая взаимодействие между классами, представлена на рисунке 6.

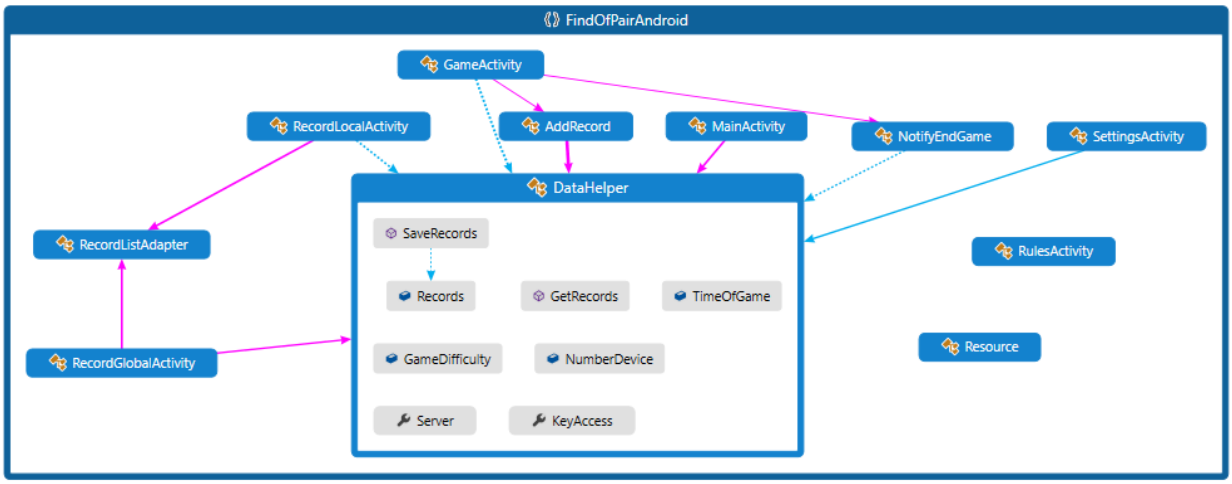

Рисунок 6 – Карта кода приложения для Android

На карте кода приняты следующие обозначения линий:

1) розовая сплошная линия – вызовы методов класса;

2) голубая сплошная линия – запись поля класса;

3) голубая пунктирная линия – чтения поля класса.

Класс **GameActivity**. При инициализации в зависимости от выбранной сложности игры устанавливает соответствующую разметку экрана и инициализирует модель игрового поля заданного размера. Здесь используется схема: один управляющий класс – две разметки. Это обусловлено тем, что отличия в логике минимальны и ограничиваются только различием в размере игрового поля. Включает в себя ряд методов:

1) private void AssignImageNormal() – метод сопоставляет картинки между элементами управления, из которых состоит игровое поле при нормальном уровне сложности;

2) private void AssignImageHard() – метод сопоставляет картинки между элементами управления, из которых состоит игровое поле при нормальном уровне сложности;

3) private bool CheckWin() – метод возвращает true, если все клетки игрового поля открыты, и false, если нет;

4) private void GameActivity_Click(object sender, EventArgs e) - основной обработчик, реализует именно логику игры с использованием модели данных и модели игрового поля из переносной библиотеки классов. Алгоритм метода представлен на рисунке 7.

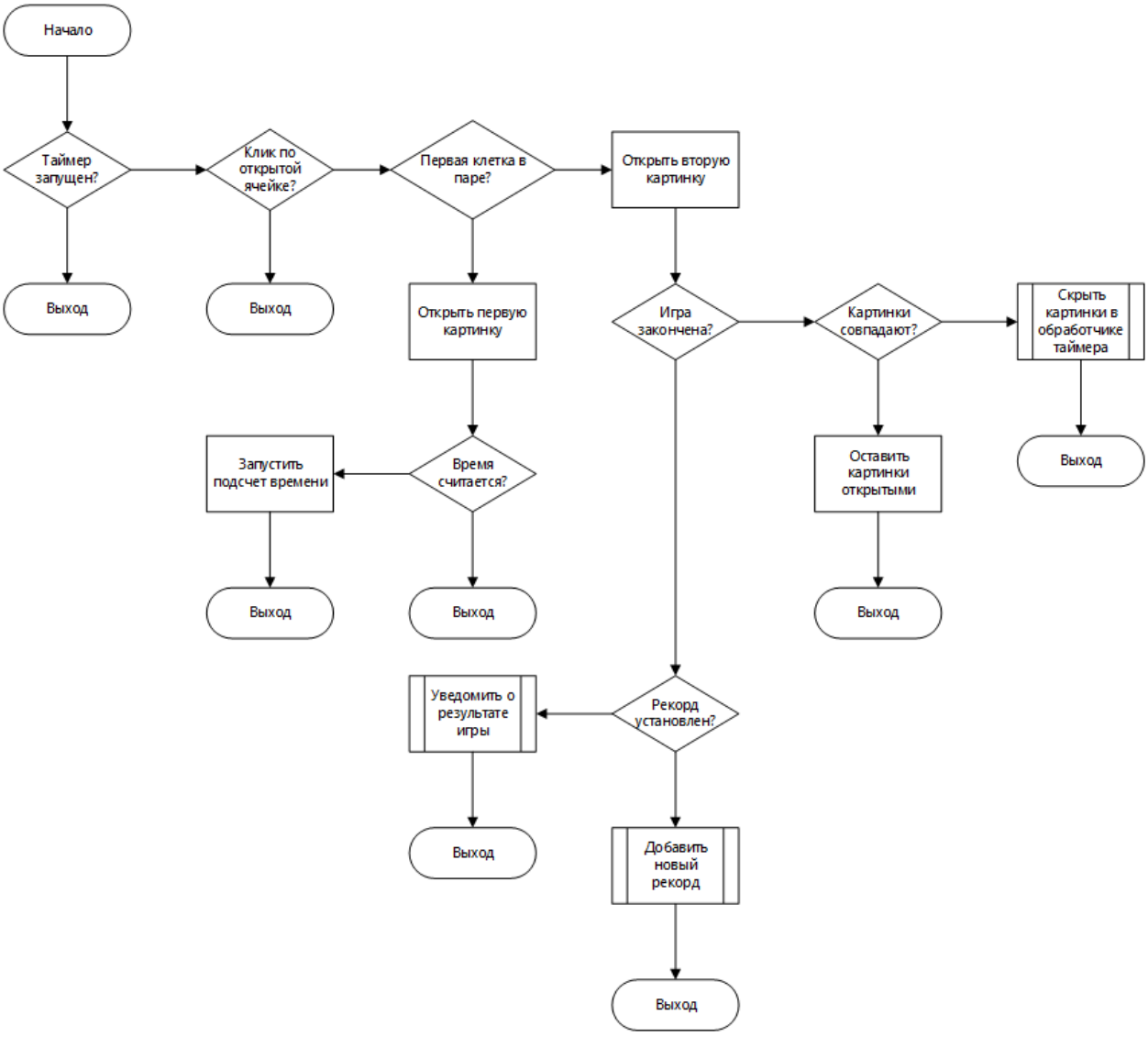

Рисунок 7 - Алгоритм обработчика клика по клетке

Использование таймера задержки скрытия пары различных картинок имеет некоторые особенности. Обработчик таймера скрывает картинки, путем подмены изображений на элементе управления, представляющим отдельную ячейку. Доступ к элементам управления (интерфейсу пользователя в целом) есть только у главного потока, а обработчик таймера выполняется в отдельном потоке и не может обновить изображение на элементах управления. Для обхода этого ограничения используется специальный объект Handler, который предоставляет доступ к очереди сообщений на выполнение в главном потоке, путем получения сообщений из других потоков, и при необходимости добавляет их в очередь сообщений (действий) для выполнения в главном потоке. Поэтому в обработчике таймера к объекту Handler отправляется специальное сообщение, а уже в обработчике этого объекта (который выполняется в главном потоке) выполняется скрытие пары различных картинок.

По результату завершения игры вызывается диалоговое окно уведомление о результатах игры, на котором отображается время прохождения уровня и, если время входит в 10 лучших для устройства, поле для ввода имени игрока и кнопка для сохранения рекорда.

Статичный вспомогательный класс **DataHelper**. Включает в себя шесть полей и два метода:

1) public static Difficulty GameDifficulty – поле хранит текущую сложность игры;

2) public static TimeSpan TimeOfGame – поле хранит результат последней игры;

3) public static Records Records – хранит таблицу рекордов;

4) public static int KeyAccess – хранит ключ доступа к серверу;

5) public static string Server – хранит адрес сервера;

6) public static int NumberDevice – хранит идентификатор устройства, который представляет id пользователя для идентификации его в глобальном хранилище рекордов среди других игроков;

7) public static Records GetRecords() – метод осуществляет чтение локальной таблицы рекордов на устройстве;

8) public static void SaveRecords(Records record) - метод осуществляет сохранение локальной таблицы рекордов на устройстве.

2.4 Клиентская часть на платформе Windows

Разработка версии игры для Windows является более простой задачей, поскольку не имеет тех ограничений, которые имели место при разработке под Android. Например, обработчик событий таймера так же выполняется в отдельном потоке, но все равно имеет доступ к элементам управления интерфейса пользователя, а значит скрывать картинки можно напрямую из кода его обработчика. Так же не требуется специального вспомогательного класса DataHelper для организации передачи данных, поскольку все требуемые экземпляры экранов находятся в запущенном состоянии, и передача нужных данных возможна простым обращением к членам класса используя ссылку на требуемый класс.

Интерфейс реализуется с использованием WindowsForm, требуется всего шесть окон:

1) главное окно (класс MainWindow) – содержит игровое поле, которое меняется в зависимости от выбранной сложности игры. Игровое поле представляет собой контейнер с элементами управления Image. В верхней части расположено меню с кнопками: меню, сложность, таблица рекордов, справка;

2) правила игры (класс Rules) – содержит информацию о правилах игры;

3) локальная таблица рекордов (класс TableOfLocalRecords) – содержит таблицу локальных рекордов;

4) глобальная таблица рекордов (класс TableOfGlobalRecords) – содержит таблицу глобальных рекордов;

5) добавить новый рекорд (класс AddRecord) – представляет окно для сохранения нового рекорда как локально, так и глобально;

6) об игре (класс AboutGame) – содержит информацию об игре.

Алгоритм обработчика клика по клетке такой же, как в версии для Android. В глобальной таблице рекордов установленный рекорд на этом компьютере выделяется цветом, чтобы его можно было отличить от остальных. Локальные рекорды хранятся сериализованом виде в каталоге развертывания приложения.

2.5 Результат разработки

В результате разработки была спроектирована переносная библиотека классов, являющаяся основой приложения, разработана серверная часть в виде REST сервиса и два клиентских приложения для ОС Windows и Android.

Рассмотрены особенности реализации интерфейсов под каждую из заданных платформ. Составлены UML диаграммы пакетов, классов, последовательностей, развертывания. Описаны методы и поля классов. Составлено руководство пользователя для обеих версий. Руководство пользователя приведено в приложении А.

3 ФИНАНСОВЫЙ МЕНЕДЖМЕНТ

3.1 Введение

В настоящее время перспективность научного исследования определяется не столько масштабом открытия, оценить которое на первых этапах жизненного цикла высокотехнологического и ресурсоэффективного продукта бывает достаточно трудно, сколько коммерческой ценностью разработки. Оценка коммерческой ценности разработки является необходимым условием при поиске источников финансирования для проведения научного исследования и коммерциализации его результатов. Особенно это важно для разработчиков коммерческого программного обеспечения, поскольку конкуренция на рынке мобильных и десктопных приложений очень высока.

Целью данного раздела является проектирование конкурентоспособной игры «Найди пары», разработанной с использованием технологии XAMARIN.

Задачами данного раздела являются оценка коммерческого потенциала и перспективности проведения разработки, а так же определение сильных и улучшения слабых сторон разрабатываемого приложения.

Процесс решения этих задач будет состоять из анализа существующих конкурентных технических решений и SWOT анализ приложения.

3.2 Анализ конкурентных технических решений

Детальный анализ конкурирующих разработок, существующих на рынке, необходимо проводить систематически, поскольку рынки пребывают в постоянном движении. Такой анализ помогает вносить коррективы в процесс разработки, чтобы успешнее противостоять своим соперникам. Важно реалистично оценить сильные и слабые стороны разработок конкурентов.

С этой целью может быть использована вся имеющаяся информация о конкурентных разработках:

- технические характеристики разработки;
- конкурентоспособность разработки;
- уровень завершенности научного исследования (наличие макета, прототипа и т.п.);
- бюджет разработки;
- уровень проникновения на рынок.

Анализ конкурентных технических решений с позиции ресурсоэффективности и ресурсосбережения позволяет провести оценку сравнительной эффективности научной разработки и определить направления для ее будущего повышения.

Данный анализ проводится с помощью оценочной карты конкурентов. В данный момент у разработки имеется несколько конкурентов: мобильное приложение для Android «Найди дубли», мобильное приложение для Windows 10 Mobile «Тренировка памяти» и браузерная игра «Memories», их анализ представлен в таблице 1.

Таблица 1 – Оценочная карта конкурентов

Критерий оценки	Вес критерия	Баллы				Конкурентоспособность			
		$Б_ф$	$Б_{к1}$	$Б_{к2}$	$Б_{к3}$	$К_ф$	$К_{к1}$	$К_{к2}$	$К_{к3}$
1) Повышение производительности труда пользователя	0,02	4	4	5	3	0,08	0,08	0,1	0,06
2) Удобство в эксплуатации	0,14	5	4	5	2	0,7	0,56	0,7	0,28
3) Помехоустойчивость	0,02	5	3	4	2	0,1	0,06	0,08	0,04
4) Энергоэкономичность	0,08	5	4	3	3	0,4	0,32	0,24	0,24
5) Надежность	0,03	5	3	4	2	0,15	0,09	0,12	0,06

Продолжение таблицы 1

Критерий оценки	Вес критерия	Баллы				Конкурентоспособность			
		$Б_ф$	$Б_{к1}$	$Б_{к2}$	$Б_{к3}$	$К_ф$	$К_{к1}$	$К_{к2}$	$К_{к3}$
6) Уровень шума	0,01	5	5	5	5	0,05	0,05	0,05	0,05
7) Безопасность	0,01	5	5	5	2	0,05	0,05	0,05	0,02
8) Потребность в ресурсах памяти	0,01	5	4	4	3	0,05	0,04	0,04	0,03
9) Функциональная мощность	0,1	5	2	4	2	0,5	0,2	0,4	0,2
10) Простота эксплуатации	0,07	5	4	4	3	0,35	0,28	0,28	0,21
11) Качество интерфейса	0,14	4	5	4	2	0,56	0,7	0,7	0,28
12) Возможности подключения в сеть ЭВМ	0,07	5	1	3	5	0,35	0,07	0,07	0,35
Экономический критерий оценки эффективности									
1) Конкурентоспособность продукта	0,1	4	3	2	1	0,4	0,3	0,2	0,1
2) Уровень проникновения на рынок	0,1	1	2	1	4	0,1	0,2	0,1	0,4
3) Цена	0,03	5	5	1	5	0,15	0,15	0,03	0,15
4) Предполагаемы срок эксплуатации	0,01	3	3	2	3	0,03	0,03	0,02	0,03
5) Послепродажное обслуживание	0,01	3	1	1	4	0,03	0,01	0,01	0,04
6) Финансирование научной разработки	0,02	1	1	1	2	0,02	0,02	0,02	0,04
7) Срок выхода на рынок	0,02	3	5	2	4	0,06	0,1	0,04	0,08
8) Наличие сертификационной разработки	0,01	1	1	1	1	0,01	0,01	0,01	0,01
Итого	**1**	**79**	**65**	**61**	**58**	**4,14**	**3,32**	**3,26**	**2,67**

Примечание:

1) Ф – данная разработка;

2) к1 - мобильное приложение для Android «Найди дубли»;

3) к2 - мобильное приложение для Windows 10 Mobile «Тренировка памяти»;

4) к3 - браузерная игра «Memories».

На основе данной оценочной карты можно сделать вывод, что данная разработка превосходит своих конкурентов по большинству параметров и в целом является лучшим решением на среди них.

Уязвимость конкурентов обусловлена несколькими позициями.

Для игры «Найди дубли» характерна низкая помехоустойчивость, ограниченный функционал, невозможность подключения к ЭВМ, слабое проникновение на рынок.

Для игры «Тренировка памяти» характерна несбалансированное энергопотребление, ограниченные возможности работы с сетью, уровень проникновения на рынок самый низкий, нет бесплатной версии.

Для игры «Memories» характерна низкая надежность, эргономика и удобство для работы пользователя оставляют желать лучшего, низкий уровень безопасности и низкая функциональность.

Разрабатываемое кроссплатформенное приложение «Найди пары» спроектировано с тем расчетом, чтобы все эти недостатки устранить и сыграть на слабых сторонах конкурентов, тем самым резко повысив конкурентоспособность приложения. Так же не обошлись без рассмотрения и сильные стороны конкурентов, которые разрабатываемое приложение реализует приблизительно на том же уровне что и конкуренты. Но, тем не менее, угрозы со стороны конкурентов существуют, поэтому необходимо рассмотреть пути развития приложения на основе сильных и слабых его сторон. Для этого используется методика рассмотрения сильных и слабых сторон, а так же возможностей и угроз с применением SWOT анализа.

3.3 SWOT анализ

SWOT – Strengths (сильные стороны), Weaknesses (слабые стороны), Opportunities (возможности) и Threats (угрозы) – представляет собой комплексный анализ разрабатываемого проекта. SWOT анализ применяют для исследования внешней и внутренней среды проекта.[14]

Матрица SWOT анализа представлена в таблице 2.

Таблица 2 - SWOT анализ

	Сильные стороны	**Слабые стороны**
	1) Кроссплатформенность 2) Поддержка современных платформ 3) Клиент-серверная архитектура 4) Отказоустойчивость клиентов 5) Постоянная техническая поддержка 6) Высокая производительность 7) Низкие системные требования	1) Один сервер 2) Малая раскрутка среди пользователей 3) Один набор картинок для игровых полей
Возможности 1) Перенос приложения на другие платформы 2) Развертывание серверов на нескольких компьютерах 3) Рекламная кампания в соцсетях, журналах, поисковиках 4) Добавление новых уровней в игру 5) Улучшение интерфейса	1) Расширить количество поддерживаемых платформ (iOS, WinPhone, Windows 10 Mobile) 2) Оптимизация работы сети 3) Доработка интерфейса	1) Активная рекламная кампания среди потенциальных пользователей 2) По мере увеличения количества игроков увеличивать количество серверов 3) Расширить набор картинок для игровых полей 4) Расширение доступных уровней
Угрозы 1) Потеря игроков из-за отказа одного сервера 2) Малый коэффициент конвертации количества показов рекламы в игроков 3) Отток игроков в связи потерей интереса к ней 4) Развитие конкурентов	1) Развернуть резервный сервер 2) Развитие игрового процесса 3) Увеличение темпов реализации планов на рост и развитие приложения	1) Увеличение стабильности работы серверной части 2) Анализ причин развития конкурентов и перенятие их опыта

3.4 Заключение

В данном разделе произведен анализ конкурентов и SWOT анализ, поставленная цель достигнута, задачи решены. Рассмотрены сильные и слабые стороны конкурентов, также выявлены свои сильные и слабые стороны. Определён путь развития в будущем.

Можно сказать, что было спроектировано конкурентоспособное приложение на базе новой современной технологии, которое готово к выходу на рынок.

4 СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ

Разрабатываемое приложение подразумевает использование в двух местах: за компьютером и на мобильном устройстве. Мобильным устройством человек может пользоваться в любом месте, а за компьютером, как правило, пользуется приложением в специально оборудованном помещении. Поэтому в данном разделе необходимо отметить особенности организации рабочего места за компьютером.

4.1 Требования к помещениям при работе за компьютером

Помещения должны иметь естественное и искусственное освещение. Расположение рабочих мест за мониторами для взрослых пользователей в подвальных помещениях не допускается.[15,16]

Площадь на одно рабочее место с компьютером для взрослых пользователей должна составлять не менее 4,5 $м^2$, а объем не менее - 15 $м^3$.[15,16]

Помещения с компьютерами должны оборудоваться системами отопления, кондиционирования воздуха или эффективной приточно-вытяжной вентиляцией.[15,16]

Для внутренней отделки интерьера помещений с компьютерами должны использоваться диффузно-отражающие материалы с коэффициентом отражения для потолка — 0,7-0,8; для стен — 0,5-0,6; для пола — 0,3-0,5.[15,16]

Поверхность пола в помещениях эксплуатации компьютеров должна быть ровной, без выбоин, нескользкой, удобной для очистки и влажной уборки, обладать антистатическими свойствами.[15,16]

В помещении должны находиться аптечка первой медицинской помощи, углекислотный огнетушитель для тушения пожара.[15,16]

4.2 Требования к микроклимату, ионному составу и концентрации вредных химических веществ в воздухе помещений

На рабочих местах пользователей персональных компьютеров должны обеспечиваться оптимальные параметры микроклимата в соответствии с СанПиН 2.2.2/2.4.1340-03. Согласно этому документу для категории тяжести работ 1а температура воздуха должна быть в холодный период года не более 22-24С, в теплый период года 20-25С. Относительная влажность должна составлять 40-60%, скорость движения воздуха — 0,1 м/с. Для поддержания оптимальных значений микроклимата используется система отопления и кондиционирования воздуха. Для повышения влажности воздуха в помещении следует применять увлажнители воздуха с дистиллированной или кипяченой питьевой водой.[15,16]

Ионный состав воздуха должен содержать следующее количество отрицательных и положительных аэройонов:

- минимально необходимый уровень 600 и 400 ионов в 1 $см^3$ воздуха;
- оптимальный уровень 3 000-5 000 и 1 500-3 000 ионов в 1 $см^3$ воздуха;
- максимально допустимый — 50 000 ионов в 1 $см^3$ воздуха.

Для поддержания оптимального ионного состава воздуха, обеспыливания и обеззараживания воздуха в помещении рекомендуется применять аппараты завода «Диод» серии «Эллион».[16]

4.3 Требования к освещению помещений и рабочих мест

В компьютерных залах должно быть естественное и искусственное освещение. Естественное освещение обеспечивается через оконные проемы с коэффициентом естественного освещения КЕО не ниже 1,2% в зонах с устойчивым снежным покровом и не ниже 1,5% на остальной территории.

Световой поток из оконного проема должен падать на рабочее место пользователя с левой стороны.[15,16]

Искусственное освещение в помещениях эксплуатации компьютеров должно осуществляться системой общего равномерного освещения.[15,16]

Освещенность на поверхности стола в зоне размещения документа должна быть 300-500 лк. Допускается установка светильников местного освещения для подсветки документов. Местное освещение не должно создавать бликов на поверхности экрана и увеличивать освещенность экрана более 300 лк. Прямую блескость от источников освещения следует ограничить. Яркость светящихся поверхностей (окна, светильники), находящихся в поле зрения, должна быть не более 200 кд/м^2.[15,16]

Отраженная блескость на рабочих поверхностях ограничивается за счет правильного выбора светильника и расположения рабочих мест по отношению к естественному источнику света. Яркость бликов на экране монитора не должна превышать 40 кд/м^2. Показатель ослепленности для источников общего искусственного освещения в помещениях должен быть не более 20, показатель дискомфорта в административно-общественных помещениях не более 40. Соотношение яркости между рабочими поверхностями не должно превышать 3:1 — 5:1, а между рабочими поверхностями и поверхностями стен и оборудования 10:1.[15,16]

Для искусственного освещения помещений с персональными компьютерами следует применять светильники с зеркализованными решетками, укомплектованные высокочастотными пускорегулирующими аппаратами. Допускается применять светильники прямого света, преимущественно отраженного света типа ЛПО13, ЛПО5, ЛСО4, ЛПО34, ЛПО31 с люминисцентными лампами типа ЛБ. Допускается применение светильников местного освещения с лампами накаливания. Светильники должны располагаться в виде сплошных или прерывистых линий сбоку от рабочих мест параллельно линии зрения пользователя при разном расположении компьютеров. При периметральном расположении — линии светильников должны располагаться

локализованно над рабочим столом ближе к его переднему краю, обращенному к оператору. Защитный угол светильников должен быть не менее 40 градусов. Светильники местного освещения должны иметь непросвечивающийся отражатель с защитным углом не менее 40 градусов.[15,16]

Для обеспечения нормативных значений освещенности в помещениях следует проводить чистку стекол оконных проемов и светильников не реже двух раз в год и проводить своевременную замену перегоревших ламп.[16]

4.4 Требования к организации и оборудованию рабочих мест

Рабочие места с персональными компьютерами по отношению к световым проемам должны располагаться так, чтобы естественный свет падал сбоку, желательно слева.[16]

Рабочий стол может быть любой конструкции, отвечающей современным требованиям эргономики и позволяющей удобно разместить на рабочей поверхности оборудование с учетом его количества, размеров и характера выполняемой работы. Целесообразно применение столов, имеющих отдельную от основной столешницы специальную рабочую поверхность для размещения клавиатуры. Используются рабочие столы с регулируемой и нерегулируемой высотой рабочей поверхности. При отсутствии регулировки высота стола должна быть в пределах от 680 до 800 мм.[15,16]

Глубина рабочей поверхности стола должна составлять 800 мм (допускаемая не менее 600 мм), ширина — соответственно 1 600 мм и 1 200 мм. Рабочая поверхность стола не должна иметь острых углов и краев, иметь матовую или полуматовую фактору.[15,16]

Рабочий стол должен иметь пространство для ног высотой не менее 600 мм, шириной — не менее 500 мм, глубиной на уровне колен — не менее 450 мм и на уровне вытянутых ног — не менее 650 мм.[15,16]

Быстрое и точное считывание информации обеспечивается при расположении плоскости экрана ниже уровня глаз пользователя, предпочтительно

перпендикулярно к нормальной линии взгляда (нормальная линия взгляда 15 градусов вниз от горизонтали).[15,16]

Клавиатура должна располагаться на поверхности стола на расстоянии 100-300 мм от края, обращенного к пользователю.[15,16]

Для удобства считывания информации с документов применяются подвижные подставки (пюпитры), размеры которых по длине и ширине соответствуют размерам устанавливаемых на них документов. Пюпитр размещается в одной плоскости и на одной высоте с экраном.[16]

Для обеспечения физиологически рациональной рабочей позы, создания условий для ее изменения в течение рабочего дня применяются подъемно-поворотные рабочие стулья с сиденьем и спинкой, регулируемыми по высоте и углам наклона, а также расстоянию спинки от переднего края сидения.

Конструкция стула должна обеспечивать:

- ширину и глубину поверхности сиденья не менее 400 мм;
- поверхность сиденья с закругленным передним краем;
- регулировку высоты поверхности сиденья в пределах 400-550 мм и углом наклона вперед до 15 градусов и назад до 5 градусов.;
- высоту опорной поверхности спинки 300±20 мм, ширину — не менее 380 мм и радиус кривизны горизонтальной плоскости 400 мм;
- угол наклона спинки в вертикальной плоскости в пределах 0±30 градусов;
- регулировку расстояния спинки от переднего края сидения в пределах 260-400 мм;
- стационарные или съемные подлокотники длиной не менее 250 мм и шириной 50-70 мм;
- регулировку подлокотников по высоте над сиденьем в пределах 230±30 мм и внутреннего расстояния между подлокотниками в пределах 350-500 мм.;

- поверхность сиденья, спинки и подлокотников должна быть полумягкой, с нескользящим неэлектризующимся, воздухонепроницаемым покрытием, легко очищаемым от загрязнения.[15,16]

Рабочее место должно быть оборудовано подставкой для ног, имеющей ширину не менее 300 мм, глубину не менее 400 мм, регулировку по высоте в пределах до 150 мм и по углу наклона опорной поверхности подставки до 20 градусов. Поверхность подставки должна быть рифленой и иметь по переднему краю бортик высотой 10 мм.[15,16]

4.5 Режим труда и отдыха при работе с компьютером

Режим труда и отдыха предусматривает соблюдение определенной длительности непрерывной работы на ПК и перерывов, регламентированных с учетом продолжительности рабочей смены, видов и категории трудовой деятельности.[16]

Виды трудовой деятельности на ПК разделяются на 3 группы: группа А — работа по считыванию информации с экрана с предварительным запросом; группа Б — работа по вводу информации; группа В — творческая работа в режиме диалога с ПК .[15,16]

Если в течение рабочей смены пользователь выполняет разные виды работ, то его деятельность относят к той группе работ, на выполнение которой тратится не менее 50% времени рабочей смены.[15,16]

Категории тяжести и напряженности работы на ПК определяются уровнем нагрузки за рабочую смену: для группы А — по суммарному числу считываемых знаков; для группы Б — по суммарному числу считываемых или вводимых знаков; для группы В — по суммарному времени непосредственной работы на ПК. В таблице 3 приведены категории тяжести и напряженности работ в зависимости от уровня нагрузки за рабочую смену.[15,16]

Таблица 3

Категория работы по тяжести и напряженности	Уровень нагрузки за рабочую смену при видах работы на ПК		
	Группа А Количество знаков	Группа Б Количество знаков	Группа В Время работы, ч
1	До 20000	До 15000	До 2,0
2	До 40000	До 30000	До 4,0
3	До 60000	До 40000	До 6,0

Количество и длительность регламентированных перерывов, их распределение в течение рабочей смены устанавливается в зависимости от категории работ на ПК и продолжительности рабочей смены.[15,16]

При 8-часовой рабочей смене и работе на ПК регламентированные перерывы следует устанавливать:

- для первой категории работ через 2 часа от начала смены и через 2 часа после обеденного перерыва продолжительностью 15 минут каждый;
- для второй категории работ — через 2 часа от начала рабочей смены и через 1,5-2,0 часа после обеденного перерыва продолжительностью 15 минут каждый или продолжительностью 10 минут через каждый час работы;
- для третьей категории работ — через 1,5- 2,0 часа от начала рабочей смены и через 1,5-2,0 часа после обеденного перерыва продолжительностью 20 минут каждый или продолжительностью 15 минут через каждый час работы.[15,16]

При 12-часовой рабочей смене регламентированные перерывы должны устанавливаться в первые 8 часов работы аналогично перерывам при 8-часовой рабочей смене, а в течение последних 4 часов работы, независимо от категории и вида работ, каждый час продолжительностью 15 минут.[15,16]

Продолжительность непрерывной работы на ПК без регламентированного перерыва не должна превышать 2 часа.[15,16]

При работе на ПК в ночную смену продолжительность регламентированных перерывов увеличивается на 60 минут независимо от категории и вида трудовой деятельности.[15,16]

Эффективными являются нерегламентированные перерывы (микропаузы) длительностью 1-3 минуты.[15,16]

Регламентированные перерывы и микропаузы целесообразно использовать для выполнения комплекса упражнений и гимнастики для глаз, пальцев рук, а также массажа. Комплексы упражнений целесообразно менять через 2-3 недели.[15,16]

Пользователям ПК, выполняющим работу с высоким уровнем напряженности, показана психологическая разгрузка во время регламентированных перерывов и в конце рабочего дня в специально оборудованных помещениях (комнатах психологической разгрузки).[15,16]

Медико-профилактические и оздоровительные мероприятия. Все профессиональные пользователи ПК должны проходить обязательные предварительные медицинские осмотры при поступлении на работу, периодические медицинские осмотры с обязательным участием терапевта, невропатолога и окулиста, а также проведением общего анализа крови и ЭКГ.[15,16]

Близорукость, дальнозоркость и другие нарушения рефракции должны быть полностью корригированы очками. Для работы должны использоваться очки, подобранные с учетом рабочего расстояния от глаз до экрана дисплея. При более серьезных нарушениях состояния зрения вопрос о возможности работы на ПК решается врачом-офтальмологом.[15,16]

Досуг рекомендуется использовать для пассивного и активного отдыха (занятия на тренажерах, плавание, езда на велосипеде, бег, игра в теннис, футбол, лыжи, аэробика, прогулки по парку, лесу, экскурсии, прослушивание музыки и т.п.). Дважды в год (весной и поздней осенью) рекомендуется проводить курс витаминотерапии в течение месяца. Следует отказаться от курения. Категорически должно быть запрещено курение на рабочих местах и в помещениях с ПК.[15,16]

4.6 Обеспечение электро-пожаробезопасности на рабочем месте

4.6.1 Электробезопасность. На рабочем месте пользователя размещены: дисплей, клавиатура и системный блок. При работе компьютера запрещается вытирать пыль с дисплея и системного блока, работать на компьютере во влажной одежде и влажными руками, поскольку существует риск электроудара.[16]

Перед началом работы следует убедиться в отсутствии свешивающихся со стола или висящих под столом проводов электропитания, в целостности вилки и провода электропитания, в отсутствии видимых повреждений аппаратуры и рабочей мебели, в отсутствии повреждений и наличии заземления приэкранного фильтра.[16]

Токи статического электричества, наведенные в процессе работы компьютера на корпусах монитора, системного блока и клавиатуры, могут приводить к разрядам при прикосновении к этим элементам. Такие разряды опасности для человека не представляют, но могут привести к выходу из строя компьютера. Для снижения величин токов статического электричества используются нейтрализаторы, местное и общее увлажнение воздуха, использование покрытия полов с антистатической пропиткой.[16]

4.6.2 Пожарная безопасность. Пожарная безопасность — состояние объекта, при котором исключается возможность пожара, а в случае его возникновения предотвращается воздействие на людей опасных его факторов и обеспечивается защита материальных ценностей.[16]

Противопожарная защита — это комплекс организационных и технических мероприятий, направленных на обеспечение безопасности людей, предотвращение пожара, ограничение его распространения, а также на создание условий для успешного тушения пожара.[16]

Пожарная безопасность обеспечивается системой предотвращения пожара и системой пожарной защиты. Во всех служебных помещениях обязательно должен быть «План эвакуации людей при пожаре», регламентирую-

щий действия персонала в случае возникновения очага возгорания и указывающий места расположения пожарной техники.[16]

Пожары в вычислительном центре представляют особую опасность, так как сопряжены с большими материальными потерями. Характерная особенность вычислительных центров — небольшие площади помещений. Как известно, пожар может возникнуть при взаимодействии горючих веществ, окислителя и источников зажигания. В помещениях вычислительных центров присутствуют все три основных фактора, необходимых для возникновения пожара.[16]

Горючими компонентами на вычислительных центрах являются: строительные материалы для акустической и эстетической отделки помещений, перегородки, двери, полы, изоляция кабелей и др.[16]

Источниками зажигания в вычислительных центрах могут быть электрические схемы компьютера, приборы, применяемые для технического обслуживания, устройства электропитания, кондиционирования воздуха, где в результате различных нарушений образуются перегретые элементы, электрические искры и дуги, способные вызвать загорания горючих материалов.[16]

В современных компьютерах очень высокая плотность размещения элементов электронных схем. В непосредственной близости друг от друга располагаются соединительные провода, кабели. При протекании по ним электрического тока выделяется значительное количество теплоты. При этом возможно оплавление изоляции. Для отвода избыточной теплоты от оборудования служат системы вентиляции и кондиционирования воздуха. При постоянном действии эти системы представляют собой дополнительную пожарную опасность.[16]

Для большинства помещений вычислительных центров установлена категория пожарной опасности В.[16]

Одна из наиболее важных задач пожарной защиты — защита строительных помещений от разрушений и обеспечение их достаточной прочности в условиях воздействия высоких температур при пожаре. Учитывая высокую стоимость электронного оборудования вычислительного центра, а также категорию его пожарной опасности, здания для вычислительного центра и части здания другого назначения, в которых предусмотрено размещение компьютеров, должны быть первой и второй степени огнестойкости. Для изготовления строительных конструкций используются, как правило, кирпич, железобетон, стекло, металл и другие негорючие материалы. Применение дерева должно быть ограничено, а в случае использования необходимо пропитывать его огнезащитными составами.[16]

ЗАКЛЮЧЕНИЕ

В процессе выполнения выпускной квалификационной работы был произведен обзор существующих интегрированных сред разработки, произведен выбор одной из них для использования в данной работе. Рассмотрена новая современная технология XAMARIN, согласно заданной предметной области составлено техническое задание для разрабатываемого приложения.

Разработана переносимая библиотека классов, содержащая основной функционал приложения, клиентская часть для платформ Android и Windows, а так же серверная часть приложения в виде REST сервиса. Составлено руководство пользователя для версий на обеих платформах.

Разработка велась на языке C# в среде разработки Visual Studio 2015 Enterprise with XAMARIN на базе .NET Framework 4.5.2 и Android SDK API level 20-23.

В разделе Финансовый менеджмент проанализированы основные конкуренты, выявлены достоинства и недостатки продукта в сравнении с конкурентами, и на основе этого выбран путь дальнейшего развития приложения.

В разделе Социальная ответственность рассмотрены вопросы организации рабочего места разработчика, требования к помещению, микроклимату в рабочей зоне, эргономике рабочего места, обеспечение электро и пожарной безопасности.

По завершению работы можно сказать, что поставленная цель была достигнута, задачи решены.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Введение в интегрированную среду разработки Eclipse. [Электронный ресурс] URI: http://www.javaportal.ru/java/ide/intro_eclipse.html, свободный. – Загл. с экрана. – Яз. рус. Дата обращения: 13.02.2016

2 About the Eclipse Foundation. [Электронный ресурс] URI: https://www.eclipse.org/org/, свободный. – Загл. с экрана. – Яз. англ. Дата обращения: 13.02.2016

3 NetBeans IDE Features. [Электронный ресурс] URI: https://netbeans.org/features/index.html, свободный. – Загл. с экрана. – Яз. англ. Дата обращения: 13.02.2016

4 NetBeans IDE - универсальная интегрированная среда разработки приложений. [Электронный ресурс] URI: http://hightech.in.ua/content/art-netbeans-ide. – Загл. с экрана. – Яз. рус. Дата обращения: 13.02.2016

5 Visual Studio Enterpise. [Электронный ресурс] URI: https://www.visualstudio.com/products/visual-studio-enterprise-vs. – Загл. с экрана. – Яз. рус. Дата обращения: 13.02.2016

6 Подробно о XAMARIN. [Электронный ресурс] URI: https://habrahabr.ru/post/188130, свободный. – Загл. с экрана. – Яз. рус. Дата обращения: 13.02.2016

7 Центр разработчиков XAMARIN. [Электронный ресурс] URI: http://developer.xamarin.com, свободный. – Загл. с экрана. – Яз. англ. Дата обращения: 13.02.2016

8 Центр разработчиков ANDROID. [Электронный ресурс] URI: http://developer.android.com, свободный. – Загл. с экрана. – Яз. рус., англ. Дата обращения: 13.02.2016

9 Центр разработчиков MSDN. [Электронный ресурс] URI: http://msdn.microsoft.com, свободный. – Загл. с экрана. – Яз. рус., англ. Дата обращения: 13.02.2016

10 Шилдт Герберт. Полный справочник по C# [Текст]: справочник / Шилдт Герберт. - Вильямс, 2005. – 752с.

11 Bewis T. C# Design Pattern Essentials. – NY: Ability First Limited, 2012. – 264 p.

12 Freeman A. Pro ASP.NET MVC 5 (Expert's Voice in ASP.Net). – NY: Apress, 2013. – 832 p.11 3. Пратт Т., Зелковиц М. Языки программирования: разработка и реализация / под общ. ред. А. Матросова. – СПб.: Питер, 2002. – 688 с.

13 Gomaa H. Software Modeling and Design: UML, Use Cases, Patterns, and software Architectures. – NY: Cambridge University Press, 2011. – 578 p.

14 Финансовый менеджмент, ресурсоэффективность и ресурсосбережение: учебно-методическое пособие / И.Г. Видяев, Г.Н. Серикова, Н.А. Гаврикова, Н.В. Шаповалова, Л.Р. Тухватулина З.В. Криницына; Томский политехнический университет. − Томск: Изд-во Томского политехнического университета, 2014. – 36 с.

15 Санитарно-эпидемиологические правила и нормативы СанПиН 2.2.2/2.4.1340-03 (с изменениями от 25 апреля 2007 г.). [Электронный ресурс] URI: http://www.infosait.ru/norma_doc/39/39082/index.htm#i222500, свободный. – Загл. с экрана. – Яз. рус. Дата обращения: 10.04.2016

16 Пособие по БЖД. [Электронный ресурс] URI: http://www.studfiles.ru/preview/434015, свободный. – Загл. с экрана. – Яз. рус. Дата обращения: 10.04.2016

Приложение А

(справочное)

Руководство пользователя

А.1 Руководство пользователя для версии под ОС Windows

«Найди пары» представляет собой реализацию игры типа Memories. Игровое поле представляет собой 16 и 64 клетки, содержащих скрытые пары картинок. Цель игры - открыть все картинки. Картинки открываются парами, сначала первая, потом вторая, если картинки одинаковые, то они обе остаются открытыми и так до тех пор, пока все поле не будет открыто.

Для начала игры запустите приложение. Откроется главное окно как на рисунке А.1. В верхней части окна расположены четыре области: «Меню», «Сложность», «Таблица рекордов» и «Справка».

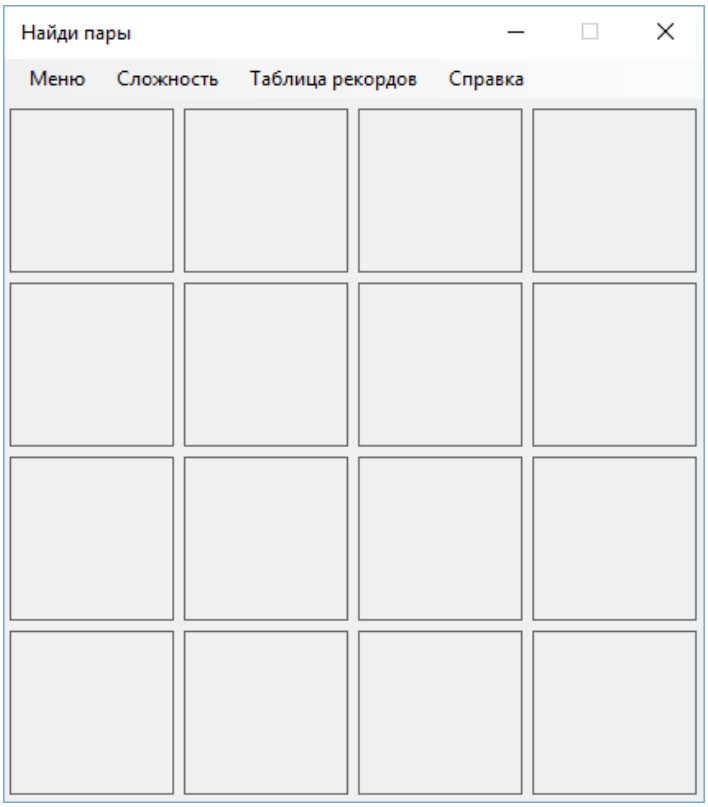

Рисунок А.1 – Главное окно приложения

В игре предусмотрено два уровня сложности:

- нормальный – игровое поле состоит из 16 клеток;
- тяжелый – игровое поле состоит из 36 клеток.

По умолчанию уровень сложности игры установлен как «Нормальный». Изменить сложность можно в соответствующем меню. Игровой процесс отражен на рисунке А.2.

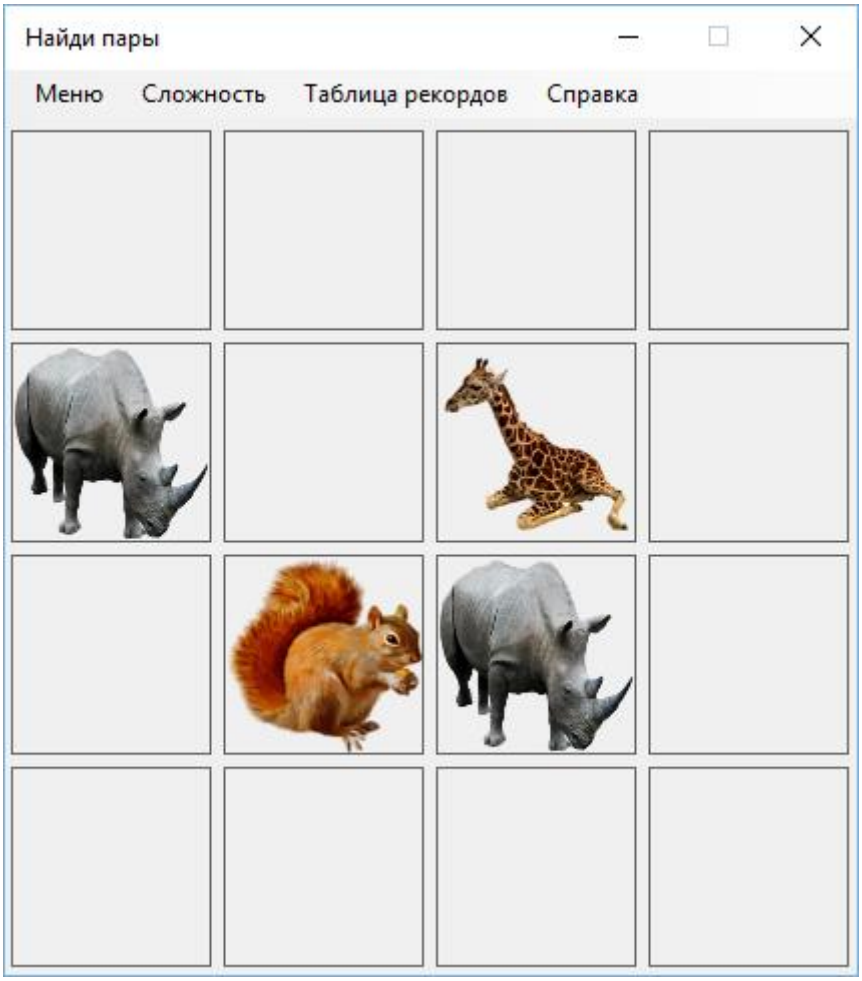

Рисунок А.2 – Игровой процесс

По окончанию игры игровое поле будет заполнено как на рисунке А.3 для нормального уровня сложности и на рисунке А.4 для тяжелого уровня сложности.

Рисунок А.3 – Заполненное игровое поле нормального уровня сложности

Рисунок А.4 – Заполненное игровое поле тяжелого уровня сложности

По окончании игры, если игрок установил рекорд, ему предлагается ввести свое имя для сохранения результата в таблице рекордов. В локальной таблице рекордов хранится десять лучших результатов. Окно сохранения рекорда представлено на рисунке А.5.

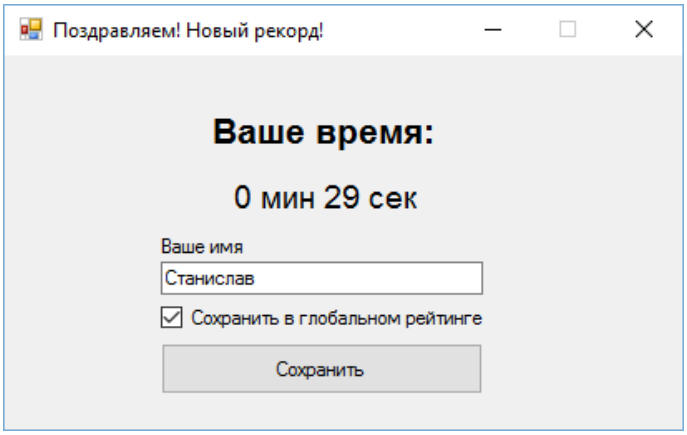

Рисунок А.5 – Окно сохранения результата игры

Чтобы посмотреть результаты текущих рекордсменов выберите соответствующий пункт в верхней области окна приложения, после чего отобразится таблица рекордов. Таблица рекордов представлена на рисунке А.6.

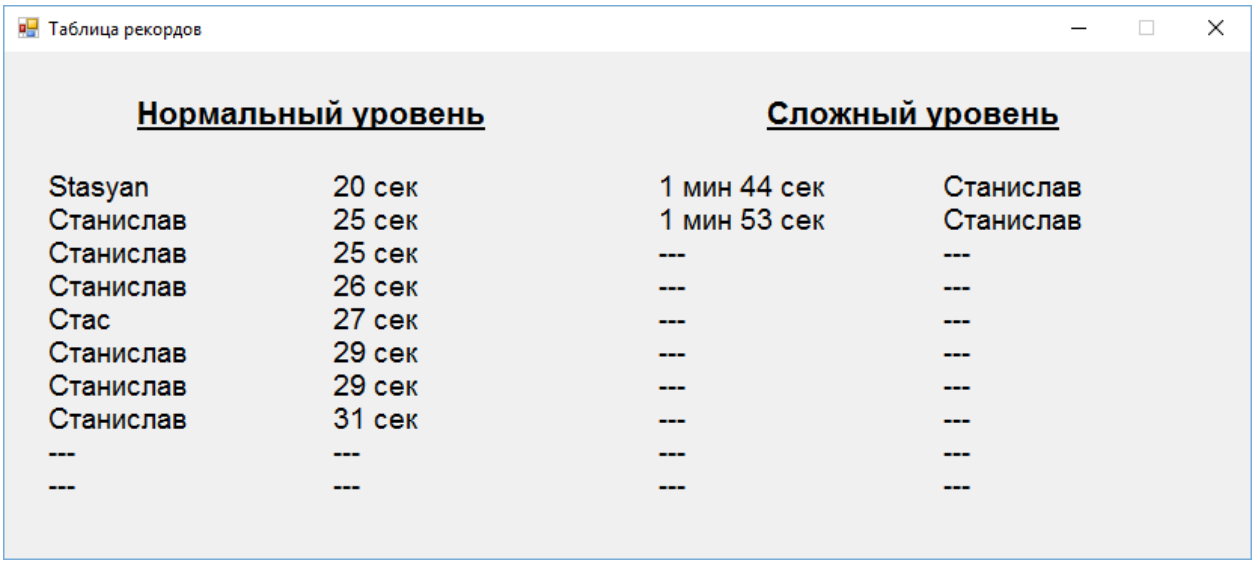

Рисунок А.6- Локальная таблица рекордов

В меню справка содержится кнопка вызова окна с правилами игры как рисунке А.7.

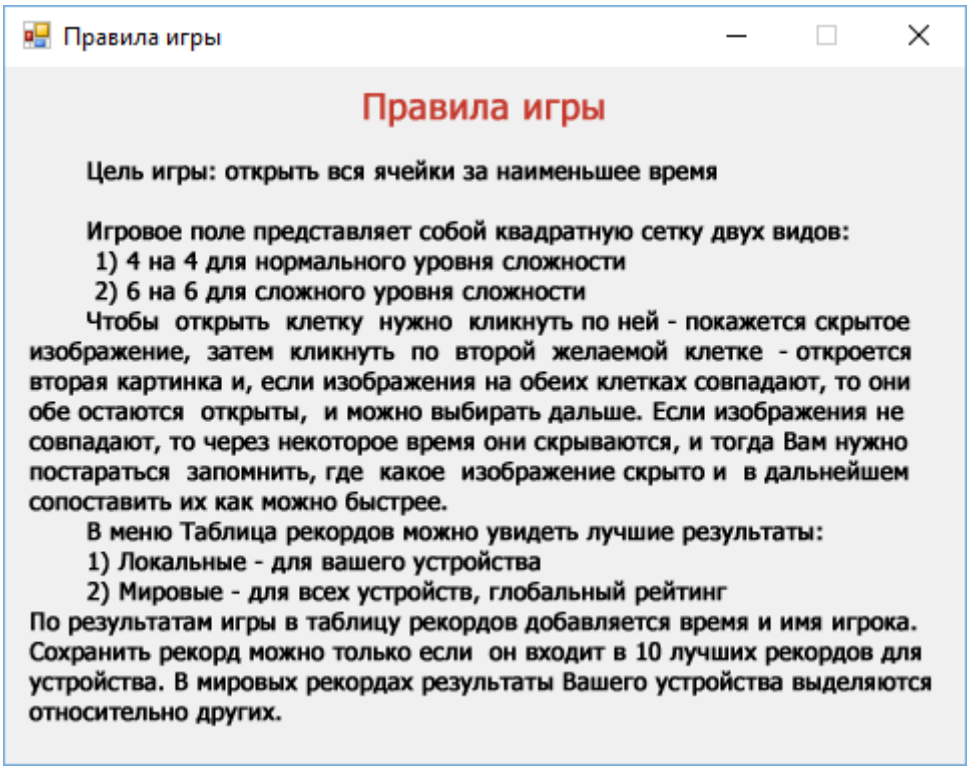

Рисунок А.7 – Окно с правила игры

А.2 Руководство пользователя для версии под ОС Android

«Найди пары» представляет собой реализацию игры типа Memories. Игровое поле представляет собой 16 и 64 клетки, содержащих скрытые пары картинок. Цель игры - открыть все картинки. Картинки открываются парами, сначала первая, потом вторая, если картинки одинаковые, то они обе остаются открытыми и так до тех пор, пока все поле не будет открыто.

Для начала игры запустите приложение. Откроется главное окно как на рисунке А.8. На экране расположены пять кнопок: «Играть», «Настройки», «Локальные рекорды», «Глобальные рекорды» и «Правила игры».

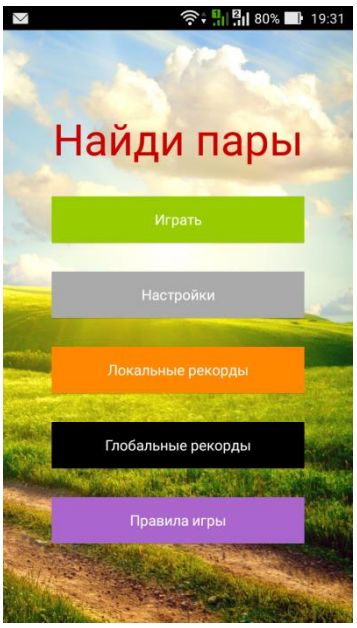

Рисунок А.8 – Главное окно приложения

В игре предусмотрено два уровня сложности:

- нормальный – игровое поле состоит из 16 клеток;
- тяжелый – игровое поле состоит из 36 клеток.

По умолчанию уровень сложности игры установлен как «Нормальный». Изменить сложность можно в соответствующем меню. Игровой процесс отражен на рисунке А.9.

Рисунок А.9 – Игровой процесс

По окончании игры, если игрок установил рекорд, ему предлагается ввести свое имя для сохранения результата в таблице рекордов. В локальной таблице рекордов хранится десять лучших результатов. Окно сохранения рекорда представлено на рисунке А.10.

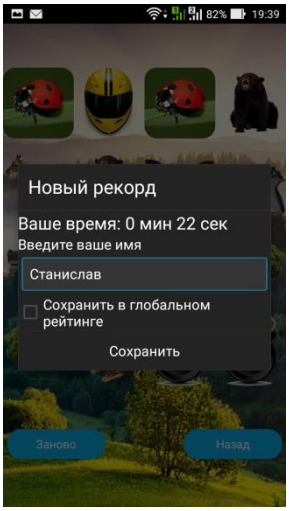

Рисунок А.10 – Окно сохранения результата игры

Чтобы посмотреть результаты текущих рекордсменов выберите соответствующий пункт в верхней области окна приложения, после чего отобразится таблица рекордов. Таблица рекордов представлена на рисунке А.11.

Рисунок А.11- Локальная таблица рекордов

Экран правил игры представлен на рисунке А.12.

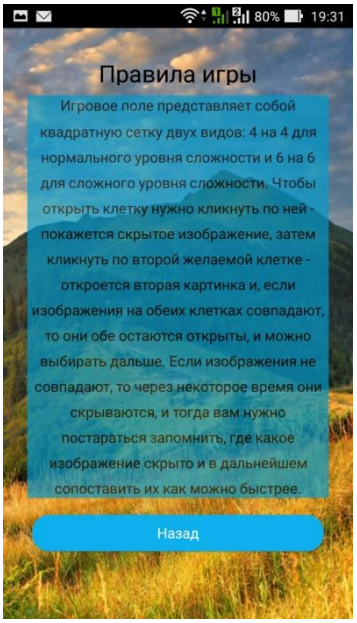

Рисунок А.12 – Экран с правилами игры