«Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)»
(СПбГЭТУ «ЛЭТИ»)

Направление	01.03.02 Прикладная математика и информатика
Профиль	Без профиля
Факультет	КТИ
Кафедра	МО ЭВМ

К защите допустить

Зав. кафедрой Кринкин К.В.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА

Тема: Создание шутера от первого лица «Страйкбол» на игровом движке Unity 3D

Студент		_____ *подпись*	Тарасенко Е.А.
Руководитель	к.т.н., доцент *(Уч. степень, уч. звание)*	_____ *подпись*	Романцев В.В.
Консультанты	ст. преподаватель *(Уч. степень, уч. звание)*	_____ *подпись*	Герасимова Т.В.
	ст. преподаватель *(Уч. степень, уч. звание)*	_____ *подпись*	Олехова Н.И.

Санкт-Петербург

2021

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ

Утверждаю
Зав. кафедрой МО ЭВМ
_____ Кринкин К.В.
«___»_____20___ г.

Студент Тарасенко Е.А. Группа 7381
Тема работы: Создание шутера от первого лица «Страйкбол» на игровом движке Unity 3D
Место выполнения ВКР: Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В.И. Ульянова (Ленина)
Исходные данные (технические требования): ОС Windows 10, Локальная сеть
Содержание ВКР: Аналитический обзор современного состояния вопроса, Разработка игрового персонажа, Разработка системы инвентаря, Описание игрового процесса и разработка игровой локации, Разработка GUI, Экономическое обоснование ВКР.
Перечень отчетных материалов: пояснительная записка, иллюстративный материал.
Дополнительные разделы: Экономическое обоснование ВКР.

Дата выдачи задания «_ _»__ _____20__ _ г.

Дата представления ВКР к защите «___»_____20___ г.

Студент		_____	Тарасенко Е.А.
Руководитель	к.т.н., доцент *(Уч. степень, уч. звание)*	_____	Романцев В.В.
Консультант	ст. преподаватель *(Уч. степень, уч. звание)*	_____	Герасимова Т.В.
Консультант	ст. преподаватель *(Уч. степень, уч. звание)*	_____	Олехова Н.И.

КАЛЕНДАРНЫЙ ПЛАН ВЫПОЛНЕНИЯ
ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

Утверждаю
Зав. кафедрой МО ЭВМ
_____ Кринкин К.В.
«___»_____20___ г.

Студент Тарасенко Е.А. Группа 7381
Тема работы: Создание шутера от первого лица «Страйкбол» на игровом движке Unity 3D

№ п/п	Наименование работ	Срок выполнения
1	Обзор литературы по теме работы	01.03 – 07.03
2	Аналитический обзор современного состояния вопроса	01.04 – 03.04
3	Разработка игрового персонажа	04.04 – 14.04
4	Разработка системы инвентаря	15.04 – 24.04
5	Описание игрового процесса. Разработка игровой локации	25.04 – 01.05
6	Разработка GUI	02.05 – 6.05
7	Экономическое обоснование ВКР	7.05 – 8.05
8	Оформление пояснительной записки	09.05 – 24.05
9	Оформление иллюстративного материала	25.05 – 28.05

Студент _____ Тарасенко Е.А.

Руководитель к.т.н., доцент _____ Романцев В.В.
(Уч. степень, уч. звание)

АННОТАЦИЯ

ШУТЕРЫ ОТ ПЕРВОГО ЛИЦА, UNITY 3D, МОДУЛЬНАЯ КАСТОМИЗАЦИЯ ОРУЖИЯ

Объектом исследования являются видеоигры жанра «шутер от первого лица».

Предметом исследования является создание шутеров от первого лица с использованием электропневматического оружия (вместо огнестрельного), что обусловливает необходимость контроля сразу за несколькими показателями, и применение к нему модульной кастомизации.

Целью работы является создание шутера от первого лица с использованием электропневматического оружия (вместо огнестрельного) и его модульной кастомизации для ПК на игровом движке Unity3D.

В ходе данной работы была создана компьютерная игра жанра «шутер от первого лица», моделирующая проведение военно-тактических игр «страйкбол». Для этого был проведен анализ предметной области и сравнение аналогов; описаны все ключевые аспекты создания игры рассматриваемого жанра на игровом движке Unity 3D. Также приведено экономическое обоснование проекта.

ABSTRACT

The object of the research is first-person shooter video games.

The subject of the research is the creation of first-person shooters using electro-pneumatic weapons (instead of firearms), which necessitates control over several indicators at once, and the application of modular customization to it.

The aim of the work is to create a first-person shooter using electro-pneumatic weapons (instead of firearms) and modular customization for PC on the Unity3D game engine.

In the course of this work, a computer game of the genre "first person shooter" was created, which simulates the conduct of military tactical games "airsoft". For this, an analysis of the subject area and comparison of analogs were carried out; describes all the key aspects of creating a game of the genre in question on the Unity 3D game engine. The economic justification of the project is also provided.

СОДЕРЖАНИЕ

Определения, обозначения и сокращения 8
1. Введение 9
1.1. Постановка задачи 9
1.2. Цель, задачи, объект и предмет исследования 10
1.3. Новизна и практическая значимость работы 11
2. Аналитический обзор современного состояния вопроса 12
2.1. Анализ и общая характеристика предметной области 12
2.2. Обзор существующих решений 13
2.3. Выводы 15
3. Разработка игрового персонажа 16
3.1. Выбор общего подхода. Структура системы управления персонажем .. 16
3.1.1. Формулировка требований к игровому персонажу 16
3.1.2. Проектирование системы управления игровым персонажем 18
3.2. Управление персонажем 19
3.2.1. Обработка событий нажатий клавиш 19
3.2.2. Обзор реализованных возможностей управления персонажем 20
3.3. Система анимаций персонажа 22
3.3.1. Общая архитектура системы анимаций персонажа 22
3.3.2. Анимации, связанные с передвижением персонажа 23
3.3.3. Анимации, связанные с использованием оружия 24
3.4. Реализация стрельбы. Определение попаданий 25
3.5. Взаимодействие игроков по локальной сети 28
3.6. Выводы 29
4. Разработка системы инвентаря 30
4.1. Формулировка требований к возможностям игрового инвентаря 30
4.2. Система представления предметов в игровом инвентаре 31
4.3. Реализация модульной кастомизации оружия 33
4.3.1. Древовидная структура конфигурации оружия 33
4.3.2. Добавление и удаление модулей из конфигурации оружия 34

4.4. Обзор реализованных возможностей игрового инвентаря 36
4.5. Выводы 38
5. Описание игрового процесса и Разработка игровой локации 39
5.1. Игровые сценарии и их реализация 39
5.1.1. Обзор общих правил и игровых сценариев страйкбола 39
5.1.2. Обзор реализованных игровых сценариев 40
5.2. Специфика создания игровой локации 41
5.2.1. Формулировка требований к игровой локации 41
5.2.2. Создание игровой локации 42
5.3. Выводы 44
6. Разработка GUI 45
6.1. Формулировка требований к GUI. Его общая структура 45
6.2. Проектирование и реализация неигровой сцены 46
6.2.1. Проектирование и реализация страницы главного меню игры 46
6.2.2. Проектирование и реализация игрового лобби 47
6.2.3. Проектирование и реализация страницы игрового инвентаря 49
6.2.4. Проектирование и реализация страницы игрового магазина 50
6.2.5. Проектирование и реализация меню настроек 51
6.3. Проектирование и реализация игровой сцены 52
6.4. Выводы 54
7. Экономическое обоснование ВКР 55
7.1. Расчет затрат на оплату труда и накладных расходов 55
7.2. Расходы на материалы 57
7.3. Издержки на амортизацию ПК и оргтехники 57
7.4. Расходы на услуги сторонних организаций 58
7.5. Себестоимость выполнения ВКР 58
7.6. Анализ конкурентов 59
7.7. Выводы 61
Заключение 62
Список использованных источников 64

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

В настоящей пояснительной записке применяются следующие сокращения и термины с соответствующими определениями:

ВКР – выпускная квалификационная работа.

ПК – персональный компьютер.

Игровой движок – базовое программное обеспечение компьютерной игры.

Скрипт – последовательность действий, описанная с помощью определенного скриптового языка программирования (в игровом движке Unity 3D это язык программирования C#).

Игровое лобби – страница настроек сетевой игры: выбора и настроек игрового режима (игрового сценария), распределения игроков по командам или позициям в рамках одной команды и пр.

Коллайдер – физическая оболочка игрового объекта.

GUI – Graphical User Interface – графический пользовательский интерфейс приложения.

UNET – Unity Networking – система, предназначенная для организации сетевого взаимодействия в игровом движке Unity 3D, представляющая собой набор сетевых инструментов, технологий и служб.

Слот – ячейка, определенное место (позиция) для размещения соответствующего компонента.

Военно-тактические игры – командные соревновательные игры с использованием игрового оружия, зачастую являющегося точной копией реальных образцов.

Страйкбол – военно-тактическая игра, в которой используются макеты игрового оружия, максимально точно копирующие реальные (боевые) прототипы, а поражающим элементом являются пластиковые шары диаметром 6 мм и массой 0.12 – 0.48 г.

ВВЕДЕНИЕ

На сегодняшний день разработка видеоигр является одной из самых обширных отраслей индустрии развлечений, по своим масштабам сопоставимой, например, с киноиндустрией. Разработка игр может требовать привлечения IT-специалистов по нескольким направлениям: программисты игровой архитектуры, программисты инструментов, скриптеры, тестировщики и пр.

Одним из наиболее коммерчески успешных жанров компьютерных игр считается «шутер от первого лица». Игры данного жанра основываются на проведении виртуальных сражений с использованием, в основном, огнестрельного оружия; игрок наблюдает за происходящим глазами протагониста. Однако со временем возрастает тенденция развития привычных механик шутера или поиска им альтернатив. Так в некоторых играх данного жанра появилась возможность использования не огнестрельного оружия или пользовательской модификации огнестрельного.

Данная работа посвящена разработке видеоигры жанра «шутер от первого лица», сражения в которой моделируют проведение военно-тактической игры «страйкбол», в которой используется электропневматическое оружие, вместо огнестрельного.

1.1. Постановка задачи

В ходе разработки игры необходимо реализовать следующий функционал:

1. Игровой персонаж: предоставление возможности передвижения и взаимодействия с предметами и другими игроками.
2. Механики использования оружия: стрельба, различные виды перезарядок, заклинивание и пр.

3. Модульная кастомизация оружия: возможность удаления и добавления модулей как непосредственно на само оружие, так и на другие модули, уже на него установленные.
4. Игровой процесс: возможность выбора и проведения сражений по одному из доступных сценариев.
5. Игровой интерфейс: доступность и удобство использования игроками всех вышеперечисленных возможностей.

1.2. Цель, задачи, объект и предмет исследования

Цель работы: создать шутер от первого лица с использованием электропневматического оружия (вместо огнестрельного) и его модульной кастомизации для ПК на игровом движке Unity3D.

Для достижения поставленной цели необходимо решить следующие **задачи**:

1. Провести обзор предметной области
2. Обозначить ключевые аспекты и технологии, необходимые для разработки приложения
3. Создать и настроить игрового персонажа (управление, анимации)
4. Разработать пользовательский инвентарь и систему для обеспечения модульной кастомизации оружия
5. Создать игровую локацию и набор игровых сценариев
6. Реализовать графический пользовательский интерфейс игры

Объектом исследования являются видеоигры жанра «шутер от первого лица».

Предметом исследования является создание шутеров от первого лица с использованием электропневматического оружия (вместо огнестрельного), что обусловливает необходимость контроля сразу за несколькими показателями, и применение к нему модульной кастомизации.

1.3. Новизна и практическая значимость работы

Игровая индустрия быстро растет и развивается, поэтому остается актуальным объектом для исследований. Жанр шутеров является одним из самых коммерчески успешных игровых жанров. Но среди шутеров (в т. ч. от первого лица) нет примеров, которые бы являлись симуляторами военно-тактической игры «страйкбол» (airsoft), т. е. с использованием электропневматического оружия. Таким образом, именно это направление наиболее перспективно для исследования.

В процессе выполнения работы будет получена компьютерная игра – шутер от первого лица, посвященный военно-тактической игре «страйкбол» с возможностью проведения модульной кастомизации используемого оружия и поддерживающий несколько режимов сражений между игроками.

Практическая новизна заключается в том, что полученная в результате выполнения работы игра значительно расширит понятие жанра шутера благодаря использованным в процессе разработки решениям.

Практическая значимость состоит в возможности использования полученной игры в качестве полноценного коммерческого проекта (релиза на популярных игровых платформах). Коммерческий успех игры будет обеспечен ее уникальным подходом к реализации некоторых базовых принципов рассматриваемого жанра.

2. АНАЛИТИЧЕСКИЙ ОБЗОР СОВРЕМЕННОГО СОСТОЯНИЯ ВОПРОСА

2.1. Анализ и общая характеристика предметной области

Компьютерная игра – это компьютерная программа, служащая для организации игрового процесса, также она осуществляет связь между игровыми партнерами или выступает сама в качестве такового.

Существует несколько жанров компьютерных игр, среди которых одним из самых коммерчески успешных является жанр «шутеров», так как, например, в США в 2018-м году на их долю пришлось около 21% от всех продаж компьютерных игр [1]. Игры данного жанра основываются на сражениях между игроками с использованием огнестрельного или любого другого оружия. Шутеры от первого лица характеризуются таким положением камеры, при котором пользователь воспринимает происходящее в виртуальном мире глазами протагониста.

Игрой, которая заложила базовые принципы рассматриваемого жанра, принято считать вышедшую в 1992 году Wolfenstein 3D. Популяризация шутеров от первого лица началась с проектов Doom (1993), Half-Life (1998) и Half-Life 2 (2004). Также стоит отметить и многопользовательский шутер, разработанный для ОС Microsoft Windows американской компанией Valve Corp., Counter-Strike (1999), бывший на тот момент лишь модификацией оригинальной Half-Life, однако с годами переросший в самостоятельную игровую серию (начиная с 2000-го года) [2].

Несмотря на возрастающие с каждым годом популярность и разнообразие игр жанра «шутер от первого лица», подавляющее большинство ограничено использованием именно огнестрельного оружия, а игры, в которых рассматриваются альтернативные виды вооружения, за редким исключением, с технической стороны реализации игрового процесса не отличаются от «огнестрельных» шутеров, что накладывает ограничение и на реализуемые игровые сценарии (режимы сражений).

Также со временем стала актуальна и кастомизация используемого в игре оружия в целях повышения удобства его использования или изменения стандартных характеристик: возможность демонтировать или установить на оружие тот или иной модуль (прицел, дополнительную рукоять, фонарик и пр.). И по сей день многие игры-представители рассматриваемого жанра либо не поддерживают так называемую модульную кастомизацию, либо сильно ограничивают ее возможности и разнообразие. На момент 2021-го года примерами игр, принадлежащих к жанру шутеров от первого лица, с лучшей кастомизацией используемого оружия являются Escape from Tarkov (2016), Call of Duty: Modern Warfare (2019) и Payday 2 (2013) [3].

2.2. Обзор существующих решений

Поиск аналогов осуществлялся в соответствие с объектом исследования, поэтому далее рассмотрены игры, принадлежащие к шутерам от первого лица.

Counter-Strike: Global Offensive [4] – многопользовательский онлайн-шутер от первого лица, предоставляющий возможность проводить командные сражения по двум сценариям (закладка бомбы и освобождение заложников) и одиночные в режиме «королевской битвы». Инвентарь в игре сильно ограничен по количеству переносимых предметов.

Escape from Tarkov [5] – также многопользовательский онлайн-шутер от первого лица, поддерживающий несколько игровых сценариев, в котором реализована модульная кастомизация оружия. Инвентарь типа «тетрис» (каждый предмет занимает определенное количество отведенных ячеек) ограничен по количеству доступных для переноса предметов. Используется огнестрельное оружие, в котором единственным параметром для отслеживания остается боезапас.

В **S.T.A.L.K.E.R.: Зов Припяти** [6] реализован также инвентарь типа «тетрис», не ограниченный по количеству предметов. Загруженность снаряжения отражается на характере передвижения игрока. Присутствует модульная кастомизация оружия.

Metro Exodus [7] имеет ограниченный по количеству переносимых предметов инвентарь, однако в игре реализована пневматическая винтовка, при использовании которой игроку необходимо осуществлять контроль не только за боезапасом, но и за давлением воздуха в резервуаре. Не все оружие поддается кастомизации; все модули можно установить только на строго определенное оружие в специально отведенное для того место.

Таким образом, были выявлены как положительные, так и отрицательные стороны каждой игры, снижающие возможности кастомизации оружия, выбора режима сражения или ограниченные использованием только огнестрельного оружия. В качестве критериев сравнения были выбраны:

1. Использование не огнестрельного оружия;
2. Вариативность кастомизации оружия;
3. Вариативность игровых сценариев;
4. Неограниченный инвентарь.

Использование не огнестрельного оружия вносит разнообразие в уже сформировавшийся набор базовых принципов шутера. Например, внедрение в процесс использования оружия дополнительного параметра, помимо количества боеприпасов, за которым необходимо будет следить игроку.

Вариативность кастомизации оружия предоставляет игроку не просто возможность его изменять визуально и технически, но и обеспечивает достаточное разнообразие в этом процессе (например, тот или иной модуль может устанавливаться в разных местах одной винтовки или подходить для установки на разные).

Вариативность игровых сценариев достигается возможностью выбрать более чем из двух режимов сражений.

Неограниченный инвентарь также расширяет возможности игрового процесса, позволяя игрокам переносить с собой большее количество предметов (не имеет ограничений по их количеству). Такие факторы, как, например, нагрузка по массе переносимого снаряжения, не ограничивают, а лишь затрудняют некоторые действия игроков.

Таблица 2.1 – Сравнение аналогов по отобранным критериям

Наименование критерия	Counter Strike: Global Offensive	Escape from Tarkov	S.T.A.L.K.E.R.: Зов Припяти	Metro Exodus
Использование не огнестрельного оружия	–	–	–	+
Вариативность кастомизации оружия	–	+	+	–
Вариативность игровых сценариев	+	+	–	–
Неограниченный инвентарь	–	–	+	–

Символ «+» означает соответствие заданному критерию, «–» – нет.

В результате анализа рынка видеоигр жанра «шутер от первого лица» не найдено игры, которая имела бы возможность использования не огнестрельного оружия, отличающегося от огнестрельного в плане игрового процесса, обладала бы достаточным разнообразием его кастомизации и доступных игровых сценариев (режимов сражений). Это подтверждает актуальность работы.

2.3. Выводы

Таким образом, в данном разделе ВКР была изучена рассматриваемая предметная область: проанализированы видеоигры жанра «шутер от первого лица». Был произведен поиск соответствующих аналогов. В результате было произведено их сравнение на предмет выявления сильных и слабых сторон каждого.

3. РАЗРАБОТКА ИГРОВОГО ПЕРСОНАЖА

В данном разделе описан процесс разработки игрового персонажа – система управления его передвижениями, соответствующими анимациями и использование игровых предметов, например, оружия.

3.1. Выбор общего подхода. Структура системы управления персонажем

3.1.1. Формулировка требований к игровому персонажу

Видеоигры жанра «шутер от первого лица» характеризуются проведением виртуальных сражений (перестрелок) с видом камеры от лица протагониста, то есть игрокам предлагается наблюдать за происходящим в виртуальном мире глазами главного героя. Далее представлен, основанный на базовых принципах данного жанра, список требований к игровому персонажу:

- Передвижения. Игровой персонаж должен иметь возможность перемещаться в нескольких направлениях: вперед, назад, влево, вправо – а также их комбинировать (перемещение по диагонали).
- Преодоление препятствий и использование укрытий. Во многих шутерах реализованы способности прыжка и приседания персонажа. В работе необходимо реализовать данные способности для преодоления препятствий и использования укрытий.
- Взаимодействие с предметами. Одним из базовых принципов игр рассматриваемого жанра является постоянная смена используемого оружия, при которой игрок «поднимает» находящееся на локации и «выбрасывает» уже имеющееся. Требуется реализовать возможность «подбора» и «выбрасывания» различных игровых предметов.
- Стрельба из оружия. Необходимо реализовать возможность стрельбы из используемого оружия. В рамках использования электропневматического оружия она должна вестись пластиковыми шариками при условии наличия заряда в батарее аккумулятора и

непустого боекомплекта. Для усложнения игрового процесса индикация заряда аккумулятора и состояния боезапаса должна отсутствовать. Должна присутствовать возможность прицеливания.

- Перезарядка оружия. Игровой персонаж должен иметь возможность перезарядить оружейный аккумулятор (так как в игре будет отсутствовать графическая индикация, то о наличии заряда должен свидетельствовать звук выстрела). С разряженным аккумулятором стрельба невозможна. В игре должна быть реализована перезарядка оружейного магазина. Выполняющие роль боеприпасов шарики должны быть достаточно заметны в момент выстрела для того, чтобы игроки могли контролировать ведение стрельбы без индикации.
- Повреждения оружия и заклинивание. В современных шутерах также реализовывается заклинивание оружия. В разрабатываемой игре необходимо создать возможность застревания боеприпасов во время стрельбы, шанс которого возрастает с уровнем поломки оружия. Уровень поломки, в свою очередь, возрастает быстрее при «холостой стрельбе» (с пустым магазином). В реальных образцах электропневматического оружия при такой стрельбе также быстрее разряжается аккумулятор. Устранение заклинивания совершается перезарядкой магазина.
- Все действия игрового персонажа должны быть проиллюстрированы соответствующими анимациями.
- Игровой прогресс. Виртуальные сражения должны вестись по локальной сети, так как подобный подход к организации игрового процесса позволит реализовать множество игровых сценариев, а персонажи, управляемые реальными людьми, будут наиболее быстро и грамотно реагировать на любые изменения игрового процесса.

3.1.2. Проектирование системы управления игровым персонажем

Игровая сцена – область игрового движка Unity 3D, на которой размещаются все необходимые объекты окружения и интерфейса [8].

Все объекты, находящееся на сцене, принадлежат типу GameObject (игровой объект). Свойства каждого объекта определяются наличием у него соответствующих компонентов – ими могут являться как уже реализованные компоненты движка, так и пользовательские скрипты.

Разработанный игровой персонаж управляется сразу несколькими ключевыми компонентами (пользовательскими скриптами):

- PlayerSetup. Общий контроль над поведением персонажа и управляющими остальными компонентами. В нем содержатся флаги, отвечающие за доступность стрельбы, прицеливания, возможности передвижения и т. п. Также компонент отвечает за связь с GUI и файлами игры (сохраненные данные о, например, количестве оставшегося боезапаса).
- FPSController. Стандартный скрипт из подключаемого пакета Unity Standard Assets, осуществляющий контроль над передвижениями персонажа. В рамках выполнения данной работы был модифицирован для хранения информации о выносливости персонажа – долгий бег и прыжки снижают уровень выносливости, а при низком ее уровне всяческие передвижения затрудняются. Также была добавлена возможность приседания для использования укрытий.
- PlayerAnimations. Компонент управления анимациями персонажа (передвижения, перезарядки оружия, прицеливания и пр.).
- Shooting. Реализация стрельбы – механика стрельбы, считывание и обработка попаданий, повреждение и заклинивание оружия.
- PlayerStatus. Сетевое взаимодействие игроков, учет прогресса в ходе сражения (как личного, так и командного). Отслеживание основных этапов проведения сражений в зависимости от выбранного сценария,

включая подведение итогов. Также компонент располагает данными о текущем статусе игрока.

На рис. 3.1 приведена UML-диаграмма сотрудничества, иллюстрирующая зависимости между данными управляющими компонентами.

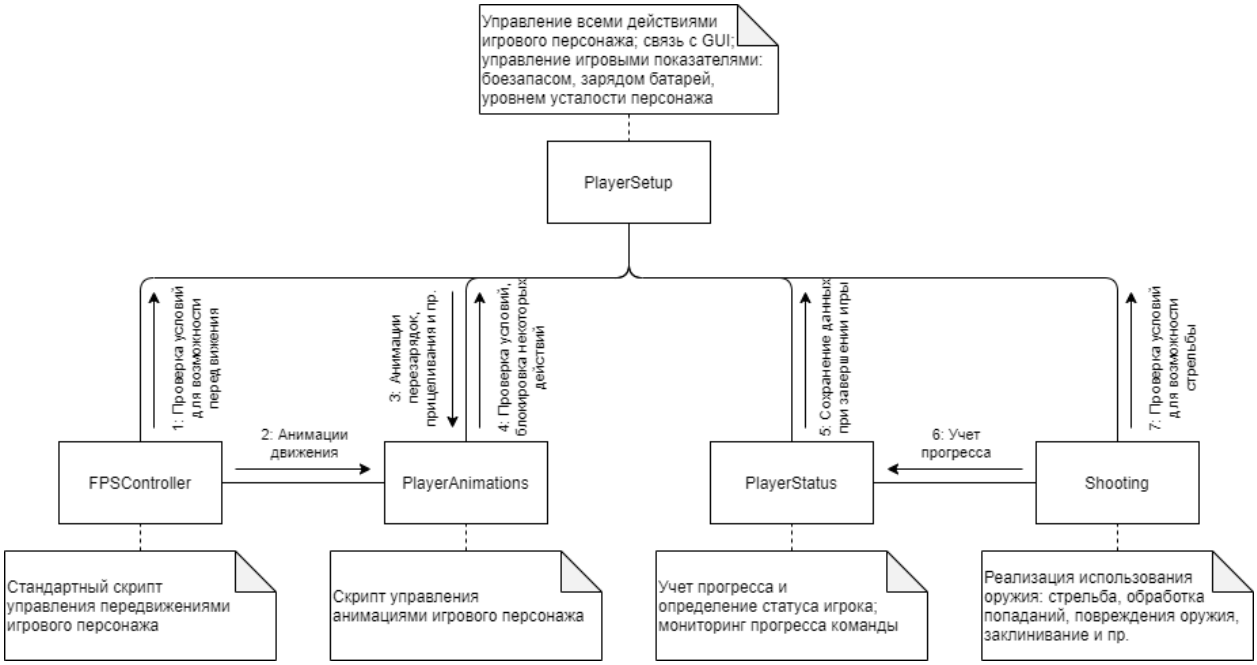

Рисунок 3.1 – UML-диаграмма сотрудничества разработанных управляющих компонентов игрового персонажа

3.2. Управление персонажем

3.2.1. Обработка событий нажатий клавиш

Перемещение персонажа; использование бега, прыжка и приседания реализовано на базе стандартного скрипта FPSController. Все манипуляции поведением игрового персонажа осуществляются нажатиями соответствующих клавиш клавиатуры или мыши.

В игровом движке Unity 3D существует несколько способов обработки событий нажатий клавиш: напрямую с помощью технологии Input.GetKey или с использованием так называемых виртуальных кнопок (Input.GetButton и Input.GetAxis [8]), определяемых в окне Edit -> ProjectSettings -> Input редактора Unity.

В данной работе при обработке событий нажатий клавиш клавиатуры и мыши была использована технология виртуальных кнопок, так как в отличие от обращения к конкретным клавишам напрямую, данный подход позволяет, при необходимости, реализовать интерфейс для их переназначения по желанию пользователя. Также переназначить клавиши управления при такой реализации можно будет и через стартовое окно Unity, появляющееся при каждом запуске собранного на данном движке приложения.

3.2.2. Обзор реализованных возможностей управления персонажем

В табл. 3.1 представлена сводная информация по всем реализованным в ходе разработки игры возможностям управления игровым персонажем на основе выдвинутых ранее требований к его функционалу.

Таблица 3.1 – Сводная информация по реализованному функционалу игрового персонажа

Клавиши по умолчанию	Назначение (способность персонажа)	Примечание
Управляющий компонент PlayerSetup		
Escape	Пауза	Команда открытия/закрытия меню паузы
I	Инвентарь	Открытие/закрытие игрового инвентаря
L	Фонарик	Срабатывает только если на текущее оружие установлен соответствующий модуль «тактический фонарь»
F	Подбор предмета	Подбор предмета, находящегося на локации, происходит, если персонаж находится в непосредственной близости к нему
1, 2, 3	Смена текущего оружия	Соответственно выбор первичного или вторичного оружия и убрать оружие

Продолжение таблицы 3.1

Клавиши по умолчанию	Назначение (способность персонажа)	Примечание
Управляющий компонент FPSController		
W, A, S, D	Направления движения персонажа	Передвижение соответственно вперед, влево, назад, вправо
Space	Использование прыжка	Способность недоступна при низком уровне выносливости персонажа
Left shift	Режим бега (если зажата)	
Left ctrl	Присесть (если зажата)	Снижает высоту силуэта персонажа
Управляющий компонент PlayerAnimations		
R	Перезарядка магазина	Доступна при непустом боекомплекте в инвентаре и в случае, когда персонаж неподвижен
T	Перезарядка аккумулятора	Срабатывает, если персонаж неподвижен
RMB	Прицеливание	
Управляющий компонент Shooting		
LMB	Выстрел	При заряженном аккумуляторе происходит проигрывание звука выстрела. При наличии непустого боезапаса производится еще и сам выстрел.

3.3. Система анимаций персонажа

3.3.1. Общая архитектура системы анимаций персонажа

Редактор Unity позволяет создавать и редактировать анимации. Композиции из созданных анимаций представляются в виде конечного (или бесконечного) автомата. Сам автомат и набор условий к нему для перехода по его состояниям называется AnimatorController [8]. Компонент управления анимациями игрового объекта (GameObject) и всех его дочерних в Unity – Animator.

Ввиду того, что, согласно требованиям, игра должна вестись по локальной сети, то локальный персонаж будет заметен каждому из подключенных игроков. Также в современных шутерах от первого лица помимо рук персонажа, имеется возможность наблюдать и за его походкой. Несмотря на возрастающую популярность данного подхода к реализации игровых персонажей, в достаточно большом количестве игр рассматриваемого жанра возможности наблюдать за походкой персонажа не присутствует. Следовательно, ее реализация повышает актуальность разрабатываемого решения на рынке видеоигр.

Таким образом, в целях удовлетворения данных условий игровой персонаж был смоделирован и анимирован полностью. При этом нижняя часть его туловища (ноги) подчиняется командам по передвижению и проигрывает анимации ходьбы, бега и т. п., а верхняя (руки и голова, за которой закреплена игровая камера) – анимации использования различного оружия (перезарядки, прицеливания и пр.). На рис. 3.2 представлен внешний вид разработанного игрового персонажа при разных положениях камеры.

Рисунок 3.2 – Внешний вид разработанного игрового персонажа при разных положениях камеры

3.3.2. Анимации, связанные с передвижением персонажа

Для управления анимациями передвижения, которые проигрывает нижняя часть туловища разработанного персонажа, были созданы следующие условия для переходов по состояниям соответствующего автомата окна аниматора:

- down_action. Целое число, отвечающее за осуществление движения: 0 – бездействие, 1 – движение.
- down_moving_mode. Целое число, отвечающее за характер передвижения (но не за его наличие): 0 – ходьба (режим по умолчанию), 1 – прыжок, 2 – режим бега, 3 – режим в приседе.

На рис. 3.3 представлена общая схема, иллюстрирующая состояния аниматора и условия перехода между ними для нижней части туловища персонажа, связанной с проигрыванием анимаций его передвижения.

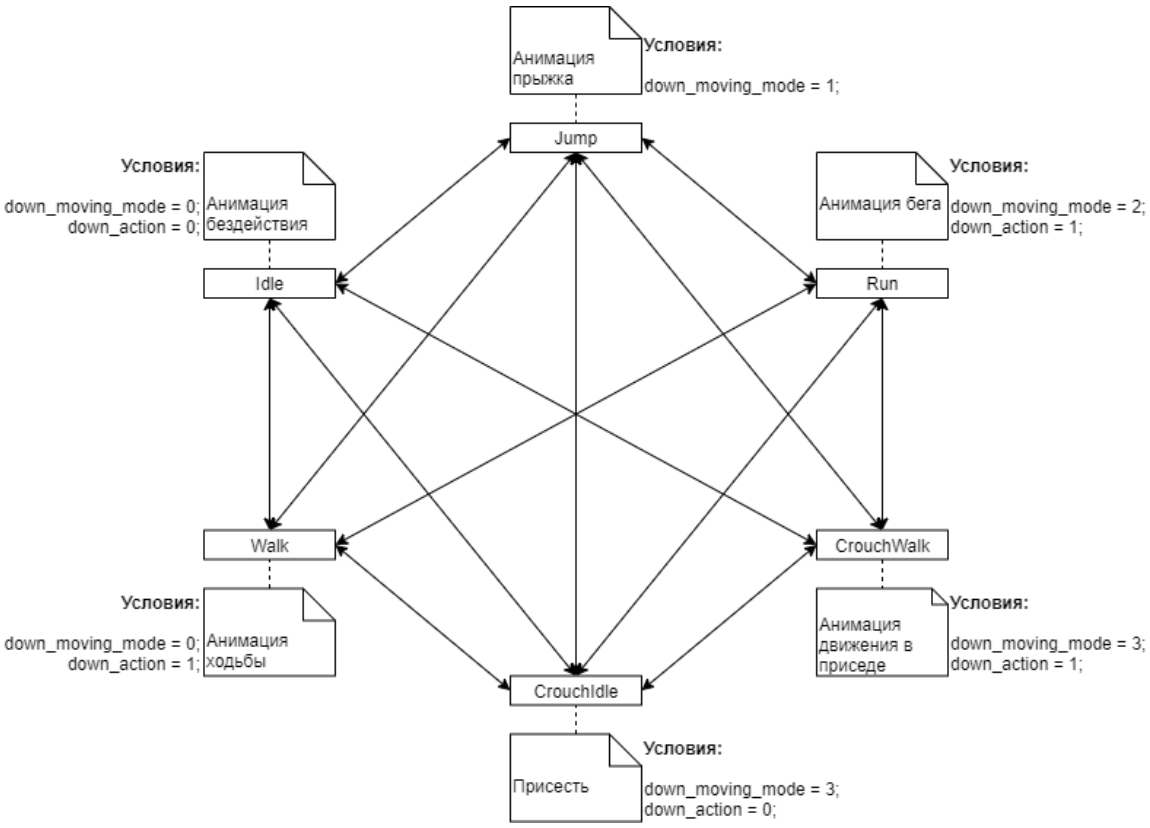

Рисунок 3.3 – Общая схема состояний и условий переходов между ними для нижней части туловища персонажа, связанной с проигрыванием анимаций его передвижения

3.3.3. Анимации, связанные с использованием оружия

Условия для перехода по состояниям анимаций, которые проигрываются верхней частью туловища персонажа (для демонстрации использования оружия; параллельно анимациям нижней части):

- down_action. Целое число, отвечающее за изменение поведения оружия в руках персонажа в зависимости от его перемещения: 0 – бездействие, 1 – движение.
- down_moving_mode. Целое число, отвечающее за характер передвижения: 0 – ходьба (режим по умолчанию), 1 – прыжок, 2 – режим бега, 3 – режим в приседе.
- up_action. Целое число, отвечающее за действие, производимое с оружием: 0 – по умолчанию, 1 – перезарядка магазина, 2 – перезарядка аккумулятора, 3 – прицеливание.

Для каждого вида оружия необходим свой набор анимаций (в т. ч. и для безоружного передвижения). На рис. 3.4 представлена общая схема системы «верхних» анимаций для разработанного тестового образца «AK_series».

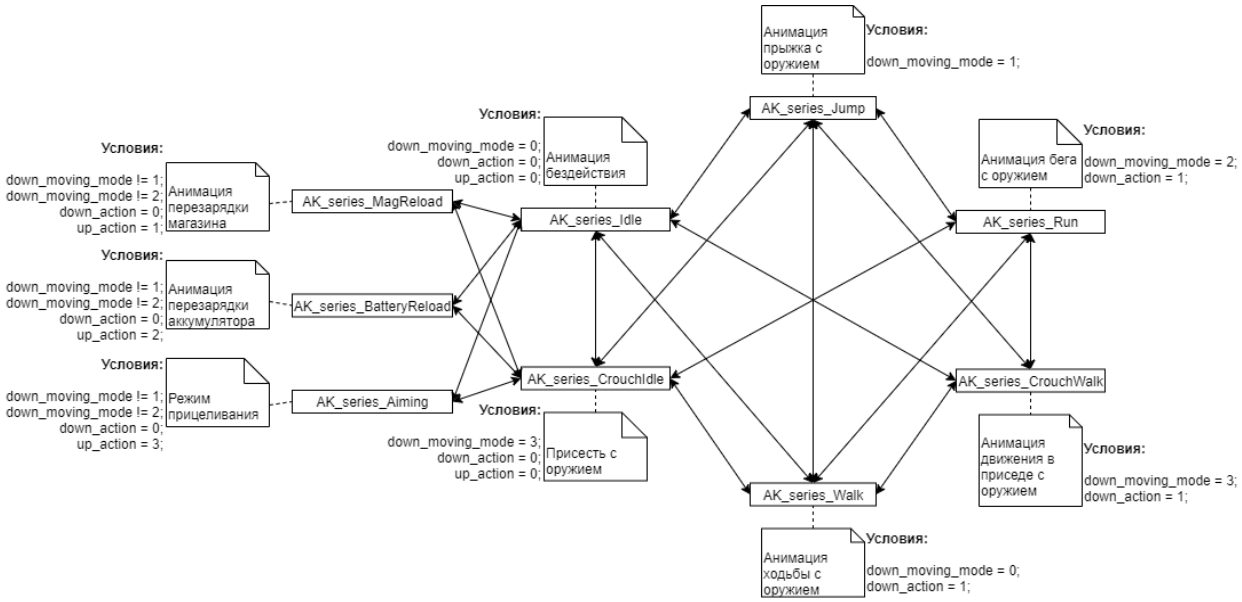

Рисунок 3.4 – Общая схема состояний и условий переходов между ними для верхней части туловища персонажа, связанной с проигрыванием анимаций использования оружия (тестового образца «AK_series»)

3.4. Реализация стрельбы. Определение попаданий

Для реализации стрельбы в шутерах (в т. ч. и от первого лица) используется две технологии: бросание лучей (Raycasting) и использование баллистики пуль [9].

Raycasting. Из точки, откуда должен производится выстрел (чаще всего это центр экрана или конец ствола игрового оружия), в направлении выстрела выбрасывается мнимый луч. Точка пересечения луча с коллайдером какого-либо игрового объекта – точка попадания условной «пули» в этот объект.

Преимущества:

- Не требует расчетов физики поведения объектов, следовательно, способствует экономии ресурсов пользовательского компьютера.
- Эффективную дальность полета пули можно отрегулировать значением дистанции, на которую можно пустить луч.

Недостатки:

- Попадание обрабатывается мгновенно.
- Направление полета боеприпаса всегда прямолинейно, что делает невозможным стрельбу по настильной траектории (криволинейной траектории, характер которой формируется преимущественно под действием силы тяжести).
- Отсутствие возможности реализации визуализации боеприпасов в полете.

Реализация баллистики пуль. В момент выстрела создается игровой объект «пули», к которому прилагается сила в направлении выстрела. Далее его траектория и время полета зависит от действующих на игровой сцене сил и параметров используемого физического движка. То есть в данном случае используется просчет физики летящего объекта. Точка соприкосновения коллайдеров «пули» и какого-либо объекта считается точкой попадания.

Преимущества:

- Возможность реализации настильной траектории полета пули.
- Возможность регулирования времени полета пули, изменяя значение ее массы и силы, прикладываемой к ней в момент выстрела.
- Возможность визуализировать летящую пулю после выстрела до ее попадания в какой-либо объект.

Недостатки:

- Расчеты физики для быстро движущихся объектов могут снизить производительность пользовательского компьютера.

Так как шары, выпущенные из электропневматического оружия, например, реплики штурмовой винтовки АК-74М – CM040C (Cyma AK-74M AEG) летят с начальной скоростью в 7.2 раза меньшей начальной скорости полета пули (для боевого образца начальная скорость полета пули – 900 м/с; для реплики – 125 м/с), то в разрабатываемой игре присутствует

необходимость визуализации летящего боеприпаса и обеспечение настильной траектории полета (эффективная дальность стрельбы боевого образца АК-74М – 1000 м; реплики – 150 – 200 м) [10][11]. Также визуализация боеприпасов необходима для обеспечения требуемого игрового процесса, при котором в игре должна отсутствовать индикация боекомплекта и игрок должен следить за наличием вылетающих из ствола игрового оружия шаров в ходе стрельбы.

Таким образом, для достижения поставленных целей была применена технология расчета баллистики пуль. В игровом движке Unity 3D это реализовывается наделением игровых объектов, к которым планируется приложение сил, компонентом Rigidbody. В момент выстрела с учетом рандомизированного вектора отклонений (его направление обусловлено точностью используемого оружия) создается объект шара (боеприпаса), наделенного компонентом Rigidbody, после чего к нему прилагается сила в направлении выстрела методом Rigidbody.AddForce(). Попадание шара в какой-либо объект на игровой сцене фиксируется благодаря стандартному методу Unity – OnCollisionEnter(). Его определение приведено на рис. 3.5. В данном методе получается и обрабатывается необходимая для дальнейшей работы информация об объекте, в который совершено попадание.

```
void OnCollisionEnter(Collision collision) {
    // Если это первое столкновение шара с объектом на сцене; определен лок. игрок, который произвел выстрел;
    // Также объект не прин. к игнорируемым слоям
    if (!hitted && playerShootingScript && playerShootingScript.isLocalPlayer &&
        (mask.value & (1 << collision.gameObject.layer)) != 0) {
        // Попадание совершено - ост. коллизии будут проигнорированы
        hitted = true;
        // Ссылка на точку коллизии (соприкосновения)
        ContactPoint collisionContactPoint = collision.GetContact(0);
        // Расчет позиции для размещения и поворота объекта "трещины" от попадания
        Vector3 pos = collisionContactPoint.point + collisionContactPoint.normal * 0.03f;
        Quaternion rot = Quaternion.LookRotation(-collisionContactPoint.normal);
        // Команда лок. игроку с данными об объекте, в который сов. попадание
        playerShootingScript.LocalHittedObjectProccessing(collision.gameObject, pos, rot);
    }
}
```

Рисунок 3.5 – Определение попадания шара в объект на сцене путем считывания соприкосновения их коллайдеров

3.5. Взаимодействие игроков по локальной сети

Для реализации виртуальных сражений по нескольким игровым сценариям (что требуется для выполнения поставленных задач) они в разрабатываемой игре должны вестись по локальной сети, так как персонажи, управляемые реальными людьми, смогут наиболее быстро и грамотно реагировать на любые изменения в ходе игрового процесса. Также реализация возможности сетевых сражений, вместо сражений с процедурно управляемыми персонажами, призвана ускорить процесс разработки игры без ущерба всем требуемым особенностям игрового процесса.

Организация игр по локальной сети в игровом движке Unity 3D обеспечивается системой UNET. Для этого на игровую сцену добавляется объект со специальным компонентом NetworkManager или NetworkLobbyManager [8]. В данных компонентах содержатся методы для создания игрового лобби и подключения к уже созданному в пределах локальной сети, в которой находится ПК пользователя. Все игровые объекты, над которыми в процессе сетевого взаимодействия будут производится манипуляции, наделяются компонентом сетевой идентификации – NetworkIdentity.

В разрабатываемой игре использовался NetworkLobbyManager, так как он, в отличие от компонента NetworkManager, дает возможность реализации интерфейса игрового лобби (путем разделения игры на офлайн-сцену, например, с меню и онлайн-сцену с самой игрой). Для взаимодействия сетевых объектов (для добавления или удаления объектов со сцены в процессе игры, для изменения параметров компонентов других игроков) используются методы с атрибутами Client, Command, ClientRpc и др. Общая схема процедурного взаимодействия сетевых объектов приведена на рис. 3.5. На схеме: клиент (client) – компьютер подключенного игрока; сервер (server) – компьютер, на котором создано игровое лобби; хост (host) – компьютер подключенного игрока, на котором было создано игровое лобби (одновременно выполняет функции сервера и клиента).

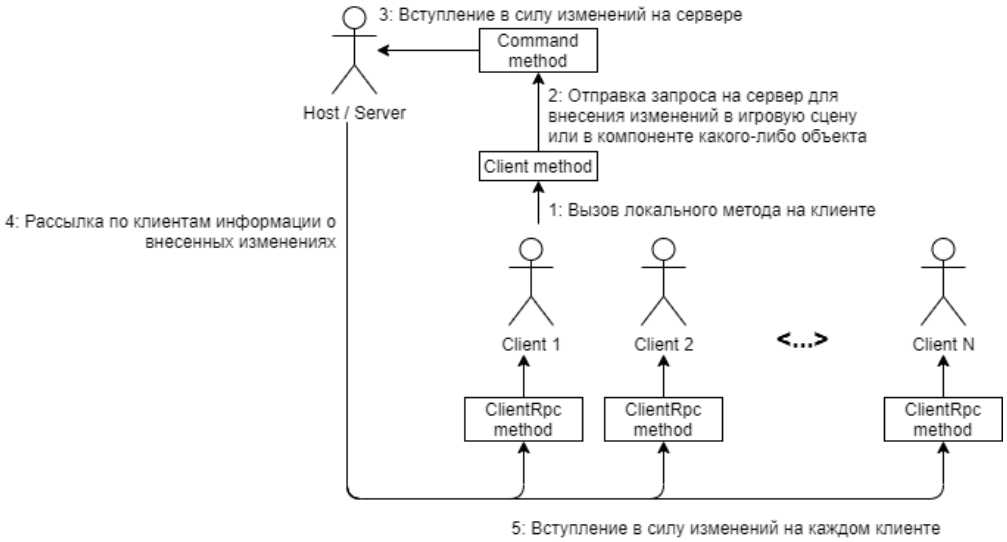

Рисунок 3.5 – Общая схема процедурного взаимодействия сетевых объектов с использованием системы UNET

3.6. Выводы

В рамках данного раздела была описана реализованная система управления функционалом игрового персонажа.

У игроков имеется возможность перемещения по локации, использования укрытий и преодоления препятствий. Для обеспечения взаимодействия пользователя с его персонажем использована технология виртуальных кнопок Unity.

Также была реализована система стрельбы и определения попаданий в рамках использования персонажем игрового оружия. Стрельба реализована с использованием физики (расчета баллистики пуль).

Все действия игрового персонажа иллюстрируются соответствующими анимациями, разделенными на нижние (воспроизведение способов передвижений) и верхние (иллюстрация взаимодействия с оружием).

Игра ведется по локальной сети, а взаимодействие игроков в ней организовано с помощью системы UNET.

4. РАЗРАБОТКА СИСТЕМЫ ИНВЕНТАРЯ

В данном разделе описан процесс разработки системы игрового инвентаря – специфика представления игровых предметов и особенности реализованной системы модификации игрового оружия.

4.1. Формулировка требований к возможностям игрового инвентаря

Использование электропневматического оружия порождает необходимость его наиболее детальной кастомизации. Помимо различных типов аккумуляторов и магазинов, на подобного рода оружие можно устанавливать предназначенные для боевых образцов прицелы, рукояти, упоры и т. п. Это обусловливается тем, что оружие, использующееся в военно-тактических играх, как правило, является (вплоть до используемых в изготовлении материалов) точной копией боевого. Далее представлены подробные требования к итоговому функционалу игрового инвентаря:

- Общая структура инвентаря. Пользовательские предметы могут сохраняться в одном из двух хранилищ – доступном для использования только вне игры (игровой «склад») и доступном всегда (непосредственно игровой инвентарь, «амуниция»). Вне игры пользователи должны иметь возможность переносить предметы из одного хранилища в другое и назначать первичное и вторичное оружие.
- Ресурсы. В игре должно быть реализовано два вида ресурсов: игровая валюта (деньги) и шары, выполняющие роль боеприпасов. В меню должна присутствовать возможность приобретения боеприпасов и предметов за игровую валюту, также должна быть реализована возможность продажи предметов из инвентаря и платная починка поврежденного оружия.
- Модульная кастомизация. Система программного представления конфигурации оружия (с учетом всех установленных на него модулей) должна предоставлять возможность реализации функционала по

удалению уже установленных на оружие модулей и добавлению новых.

Модульная кастомизация подразумевает представление любого предмета в виде совокупности модулей с возможностью формирования пользовательской конфигурации (редактирования сборки предмета путем замены конкретных модулей в его составе). Все модули, составляющие рассматриваемый предмет, образуют между собой иерархическую систему.

В рамках разработки игрового инвентаря необходимо реализовать многоуровневую модульную кастомизацию оружия, выступающего в данном случае в роли предмета, конфигурацию которого пользователь (игрок) должен иметь возможность настраивать. Таким образом, оружие представимо в виде многоуровневой иерархической системы составляющих его модулей.

4.2. Система представления предметов в игровом инвентаре

Для выполнения поставленных задач, каждый модуль наделяется уникальным идентификатором, который является именем соответствующего объекта на игровой сцене, и компонентом (скриптом) Module, который содержит торговую информацию о модуле, его технические характеристики (степени влияния на те или иные характеристики оружия, если он будет на него установлен) и методы для проведения над ним манипуляций по его установке или демонтажу (удалению) из конфигурации оружия.

Структура компонента Module приведена на рис. 4.1. Здесь:

- slots – список экземпляров класса Slot, в каждом из которых хранится информация о месте на модуле, куда возможно установить другой (transform_info). К каждому слоту прилагается соответствующий объект указателя (для отображения в GUI) pointer и массив доступных для размещения модулей avail_modules.
- specificationsChange – экземпляр класса SpecificationsChange; набор переменных, отвечающих за то, как установка данного модуля повлияет на общие технические характеристики оружия: емкость

магазина (mag_capacity), батареи (battery_capacity), скорость повреждения при выстреле (destruction_rate), масса (weight), разброс при стрельбе (deviation) и интервал между выстрелами в секундах (shottime_delta).

- cost – стоимость предмета (модуля, целого оружия) в игровом магазине.
- shop_name – торговое наименование предмета.
- manufacturer – наименование производителя.
- category – категория, к которой принадлежит предмет.
- icon – изображение предмета для его иллюстрации в GUI инвентаря и игрового магазина.

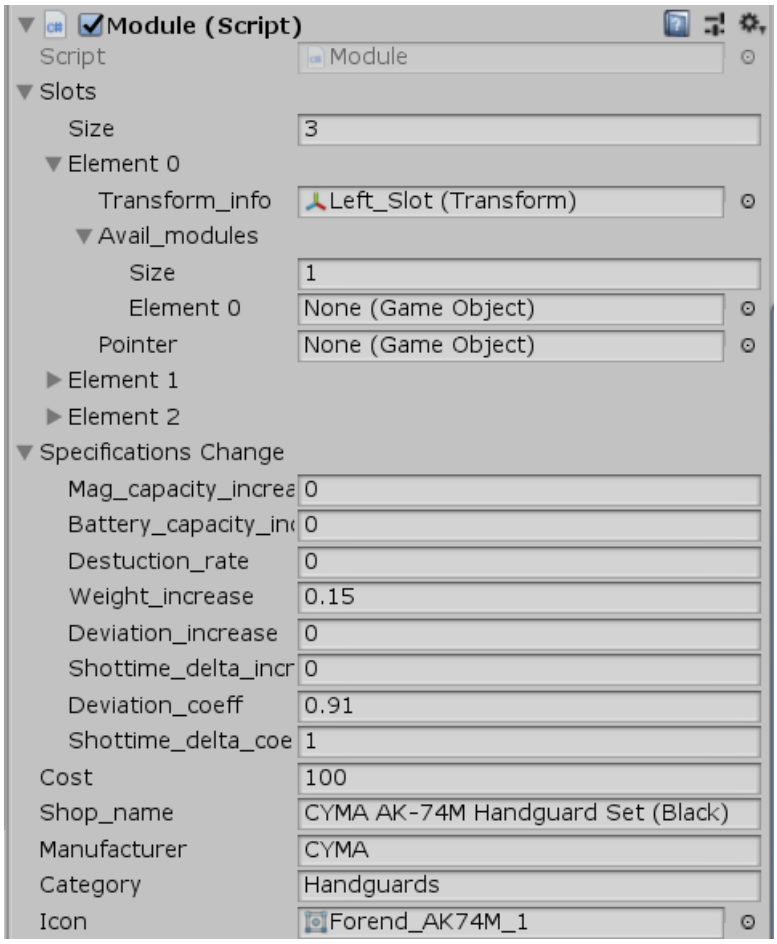

Рисунок 4.1 – Структура компонента Module, содержащего информацию о модуле и методы для его внедрения и удаления из конфигурации оружия

4.3. Реализация модульной кастомизации оружия

4.3.1. Древовидная структура конфигурации оружия

Многоуровневая модульная кастомизация подразумевает представление оружия в виде многоуровневой иерархической структуры доступных для установки модулей, что процедурно наиболее эффективно можно отразить в виде дерева. Таким образом, выбор древовидной структуры представления конфигурации оружия обусловлен тем, что на корпус может быть установлено несколько модулей (на определенные позиции), на которые, в свою очередь, могут быть установлены другие.

На рис. 4.2 приведен пример дерева, соответствующего некоторой конфигурации сборки тестового образца игрового оружия, и структура идентификатора этого образца, соответствующего построенному дереву.

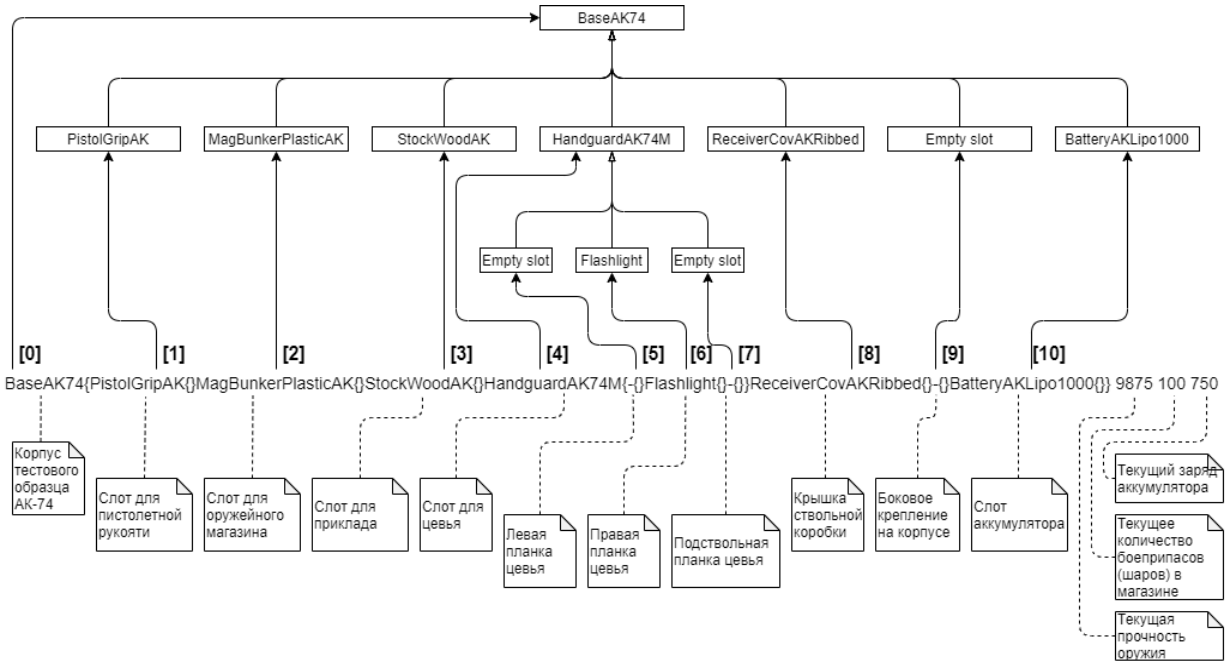

Рисунок 4.2 – Пример дерева, иллюстрирующего конфигурацию тестового образца игрового оружия, и соответствующая структура идентификатора данного образца с учетом позиции в нем каждого модуля

4.3.2. Добавление и удаление модулей из конфигурации оружия

В скрипте Module находятся определения для методов добавления (AddModule()), удаления (RemoveModule()) модулей из общей конфигурации оружия, также там присутствует определение метода для его сборки (отображения) по заданной конфигурации (BuildWeapon()). Во всех вышеуказанных методах производится обход дерева, которое иллюстрирует соответствующую конфигурацию. В общем случае алгоритмы для решения подобных задач делятся на обход в глубину и в ширину.

Обход в ширину осуществляется по иерархическим уровням, а в глубину, рассматривая каждый узел дерева с его дочерними элементами. Так как в разрабатываемой игре каждый модуль содержит информацию о своих дочерних, то для обработки конфигурации оружия необходимо использовать именно обход в глубину. Типы обходов в глубину: прямой, центрированный и обратный. Иллюстрации работы этих типов приведены на рис. 4.3.

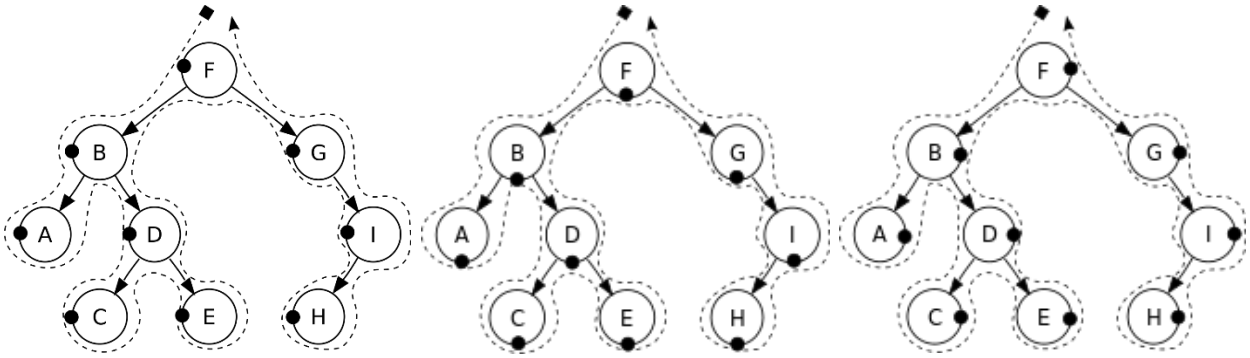

Рисунок 4.3 - Прямой, центрированный и обратный обходы дерева в глубину соответственно

Все данные варианты обхода в глубину имеют алгоритмическую сложность в худшем случае $O(N)$, где N – количество вершин в дереве, а в лучшем – $O(logN)$. Построение (для последующего отображения и использования) игрового оружия по заданной конфигурации должно вестись от родительских элементов к дочерним (доступ к дочерним элементам будет возможен только после обработки и включения отображения родительского), а удаление элемента чаще будет относится именно к узлу дерева

конфигурации (все слоты у соответствующего модуля могут быть пустыми), следовательно, в данной работе предпочтительнее использовать именно прямой обход дерева, при котором сначала обрабатывается узел, затем поочередно рекурсивно обрабатываются все дочерние элементы.

Вспомогательный метод поиска позиции модуля для удаления из конфигурации оружия представлен на рис. 4.4 (module – управляющий компонент искомого модуля для его точной идентификации при совершении обхода; moduleNumber – ссылка на текущую позицию искомого модуля в идентификаторе оружия, при вызове метода значение переменной по ссылке равно 0).

На рис. 4.5 приведен вспомогательный метод для определения позиции слота для добавления модуля (targetSlot – информация об искомом слоте, в который будет помещен модуль, для его точной идентификации при совершении обхода; moduleNumber – ссылка на текущую позицию искомого слота в идентификаторе оружия, при вызове метода значение переменной по ссылке равно 0).

Специфика нумерации позиций слотов игрового оружия в его идентификаторе была проиллюстрирована на рис. 4.2.

```
public bool FindModulePositionToRemove(Module module, ref int moduleNumber) {
    // Обход слотов текущего объекта (по умолчанию - корпуса оружия)
    foreach (Slot slot in slots) {
        moduleNumber++; // Накопление счетчика позиции, на которой наход. модуль для удаления
        // Поиск установленного на рассматриваемый слот модуля (он будет включен)
        foreach (GameObject avail_module in slot.avail_modules) if (avail_module.activeSelf) {
                if (avail_module.GetComponent<Module>() == module) return true; // Искомый модуль для удаления найден
                if (avail_module.GetComponent<Module>().FindModulePositionToRemove(module, ref moduleNumber)) return true;
                // Искомый модуль для удаления найден у одного из дочерних (рекурсивный обход)
                break;
            }
    }
    return false; // У данного род. объекта и всех его дочерних искомого модуля не обнаружено
}
```

Рисунок 4.4 – Вспомогательный метод для поиска позиции для удаления модуля из конфигурации оружия

```
public bool FindModulePositionToAdd(Transform targetSlot, ref int moduleNumber) {
    // Обход слотов текущего объекта (по умолчанию - корпуса оружия)
    foreach (Slot slot in slots) {
        moduleNumber++; // Накопление счетчика позиции, на которой наход. слот для добавления модуля
        if (slot.transform_info == targetSlot) return true; // Искомый слот для добавления модуля найден
        // Рекурсивный обход всех дочерних объектов (если такие есть - они будут включены) для поиска необходимого слота
        foreach (GameObject avail_module in slot.avail_modules) if (avail_module.activeSelf) {
                if (avail_module.GetComponent<Module>().FindModulePositionToAdd(targetSlot, ref moduleNumber)) return true;
                // Искомый слот найден у дочернего объекта
                break;
            }
    }
    return false; // У данного род. объекта и всех его дочерних искомый слот не обнаружен
}
```

Рисунок 4.5 – Вспомогательный метод для поиска позиции слота для добавления модуля в конфигурацию оружия

4.4. Обзор реализованных возможностей игрового инвентаря

Игровой инвентарь, согласно выдвинутым к нему требованиям, подразделен на 2 основных хранилища предметов: storage (игровой склад), куда сохраняются вне игры снятые с оружия модули и купленные в магазине предметы, и ammunition (амуниция), инвентарь, который игрок должен использовать непосредственно во время проведения виртуальных сражений (во время них у игроков нет доступа к складу).

В свою очередь амуниция состоит из неограниченного по количеству предметов хранилища данных и определенных слотов для первичного и вторичного оружия. Для амуниции при каждом изменении ее состава подсчитывается масса всех расположенных там предметов – уровень загруженности инвентаря в процессе сражения влияет на скорость потери выносливости игровым персонажем.

Игроки имеют возможность переносить предметы из хранилища storage в ammunition и наоборот, а также назначать первичное и вторичное оружие. Методы-обработчики соответствующих событий определены в скрипте Inventory, назначенном специально созданному пустому объекту, всегда присутствующему на игровой сцене.

Сохранение пользовательских данных о составе хранилищ и слотов для оружия происходит автоматически при каждом их изменении с использованием метода BinaryFormatter.Serialize(), записывающего данные пользовательского инвентаря в бинарный файл data.sv (рис. 4.6). Также это

позволяет игрокам получать доступ к некоторым возможностям инвентаря в процессе проведения виртуальных сражений.

```
public void SaveUserData() {
    if (!Directory.Exists(Application.dataPath + "/User")) Directory.CreateDirectory(Application.dataPath + "/User");
    FileStream fs = new FileStream(Application.dataPath + "/User/data.sv", FileMode.Create);
    BinaryFormatter formatter = new BinaryFormatter();
    formatter.Serialize(fs, current_data);
    fs.Close();
}
```

Рисунок 4.6 – Сохранение данных пользовательского инвентаря в бинарный файл

В разрабатываемой игре реализовано 2 вида ресурсов: виртуальная валюта и счетчик общего количества боеприпасов (шаров) для использования их в процессе игры. Также был реализован игровой магазин, где пользователь за виртуальную валюту может приобрести любой модуль или определенное количество боеприпасов (шаров). Присутствует возможность продажи (за 95% от стоимости покупки) любого предмета из инвентаря и платной починки оружия (в зависимости от уровня повреждения).

Благодаря разработанной древовидной системе программного представления конфигурации оружия и соответствующей ее шифровке в идентификатор, игроки имеют возможность размещения имеющихся в инвентаре модулей на доступные слоты первичного или вторичного оружия и снятия уже установленного модуля – в таком случае выбранный модуль и все, закрепленные за ним (если такие имеются), переместятся в хранилище storage (вне игры) или ammunition (во время проведения виртуального сражения).

На рис. 4.7 представлены примеры отображения в игре тестового образца оружия при различных конфигурациях его сборки.

Рисунок 4.7 – Примеры отображения тестового образца оружия в игре при различных конфигурациях его сборки

4.5. Выводы

В данном разделе были описаны функционал созданного игрового инвентаря и особенности программного представления предметов. Описана специфика древовидного представления конфигурации игрового оружия и обоснован выбор алгоритма обхода дерева, который лег в основу проектирования и реализации всех возможных манипуляций над рассматриваемыми конфигурациями.

Игровое оружие процедурно представляется в виде совокупности модулей, организованных в виде многоуровневой иерархической (древовидной) структуры. В качестве алгоритма для обхода дерева конфигурации оружия для добавления и удаления из него модулей был выбран прямой обход в глубину.

5. ОПИСАНИЕ ИГРОВОГО ПРОЦЕССА И РАЗРАБОТКА ИГРОВОЙ ЛОКАЦИИ

В данной главе описана специфика реализованного игрового процесса на основе реальных правил проведения военно-тактических игр и процесс разработки игровой локации, которая позволила бы проводить виртуальные сражения по каждому из отобранных игровых сценариев.

5.1. Игровые сценарии и их реализация

5.1.1. Обзор общих правил и игровых сценариев страйкбола

Одним из основных отличий страйкбола от других военно-тактических игр является точное следование (высокая степень копирования) реальным аналогам как в плане макетов оружия, так и в плане экипировки участников. Также отличительной чертой считается характер игровых полигонов (площадок), на которых и проводятся «сражения». Ими зачастую являются заброшенные базы отдыха, детские лагеря или военные городки (полигон «Ольгино», Санкт-Петербург, Российская Федерация [12]); огражденные лесные территории (полигон «Заходское», Ленинградская область, Российская Федерация [12]) и т. п.

Подобного рода игры проводятся по подготовленным и установленным заранее игровым сценариям с участием преимущественно двух команд. Структурно игровой полигон подразделяется на непосредственно зону сражений и точки возрождения для команд-участниц (или общая точка для всех игроков). После начала участники занимают свои позиции согласно сценарию для выполнения поставленной в рамках игры задачи. Условной «смертью» игрока считается любое попадание в него поражающим элементом (шаром), после чего он должен, выбирая кратчайший путь, наиболее быстро проследовать на точку возрождения и снова выдвинуться на позицию по истечение специально отведенного времени для возврата в игру, которое отсчитывается от момента прихода участника на точку возрождения [13].

В качестве игровых сценариев в страйкболе наиболее часто используются: встречный бой – противостояние команд на время или с ограничением по количеству возрождений; «каждый сам за себя» – аналогичный режим, при котором отсутствует разделение на команды; захват флага (флагов) – игроки команд должны доставить на свои точки возрождения либо нейтральный флаг, либо захваченный флаг условного противника (противоположной команды); штурм – в задачу одной команды входит удержание определенной точки (здания) на игровом полигоне от ее захвата другой командой [14].

5.1.2. Обзор реализованных игровых сценариев

Таким образом, в компьютерной игре, разрабатываемой в рамках выполнения ВКР, каждый участник виртуального сражения может прибывать в одном из трех статусов: alive (активен), dead (выбыл, следует к точке возрождения) и waiting (ожидает возрождения). Были реализованы следующие игровые режимы (сценарии):

- Encounter battle. Встречный бой; командная схватка без возрождений. Условие победы: все игроки противоположной команды пребывают в статусе dead. Условия поражения: все союзники и сам игрок находятся в статусе dead, а в команде противника есть хотя бы один игрок со статусом alive.
- Capturing flags. Захват флагов. Безграничное число возрождений. Условие победы: флаг противоположной команды захвачен и доставлен на точку возрождения (выбывшие игроки не могут переносить флаги). Условие поражения: команда противника выполнила задачу раньше.
- Capturing flag. Захват нейтрального флага. Аналогично предыдущему режиму, только с использованием общего нейтрального флага, вместо командных.

- Grand battle. Режим игры «каждый сам за себя» – отсутствует разделение на команды, нет возрождений. Условие победы: игрок остается единственный в статусе alive. Условие поражения: игрок выбывает.
- Assault. Штурм точки на локации. Первая команда должна достигнуть определенной точки (позиции) на игровой локации, вторая команда должна воспрепятствовать этому. Игра ограничена определенным количеством возрождений каждого игрока (в данном случае 3).
- Training. Режим тренировки, при котором разделение на команды носит условный характер; безграничное число возрождений. Отсутствует подведение итогов (нет победивших и проигравших).

5.2. Специфика создания игровой локации

5.2.1. Формулировка требований к игровой локации

В соответствие с описанными ранее правилами проведения страйкбольных состязаний и определенными игровыми сценариями разрабатываемой компьютерной игры были сформулированы следующие требования к общей структуре игровой локации, ее оформлению и ключевым объектам на ней:

- Общая структура локации должна состоять из непосредственно боевой зоны и двух точек возрождения для состязающихся команд (кроме режима Grand battle). Игроки, ожидающие возможности выдвижения на позиции, не должны покидать территории точек возрождения до истечения отведенного временного промежутка. Участники сражений не должны иметь возможности попасть за пределы игровой локации.
- Характер ландшафта. Так как чаще всего страйкбольный полигон – это комплекс заброшенных построек или лесная территория, то наиболее универсальным решением будет сочетание на игровой

локации открытого пространства, лесистой местности и области, занятой разнообразными постройками.

- Для проведения игр по сценарию Capture flag (захват нейтрального флага) одной из ключевых точек на локации должна быть оборудованная различного рода ограждениями («оборонительными позициями») для игроков обеих команд центральная зона, где и будет появляться нейтральный флаг, захват которого входит в условия победы команд при данном сценарии.
- Для проведения игр по сценарию Assault (штурм некоторого объекта на локации; чаще всего в его роли выступает определенная постройка на игровом полигоне) необходимо наличие, в качестве ключевой точки, строения (здания), находящегося в непосредственной близости от точки возрождения первой команды (обороняющейся) и расположенное на достаточной дистанции от точки возрождения второй (атакующей).

5.2.2. Создание игровой локации.

Основой для игровой локации является площадка, на которой необходимо сформировать ландшафт местности. В игровом движке Unity 3D ландшафт моделируется при помощи объекта, снабженного компонентом Terrain. Данный компонент позволяет не только произвести моделирование местности, но и покрыть ее соответствующими текстурами и растительностью.

Согласно выдвинутым требованиям касательно ландшафта, в качестве основы для игровой локации была смоделирована, представленная на рис. 5.1, местность, где 1 – вытянутое вдоль всей игровой локации открытое пространство (пляж); 2 - центральная территория, на которой должны располагаться разнообразные постройки; 3 – занимающая большую часть полигона лесная зона.

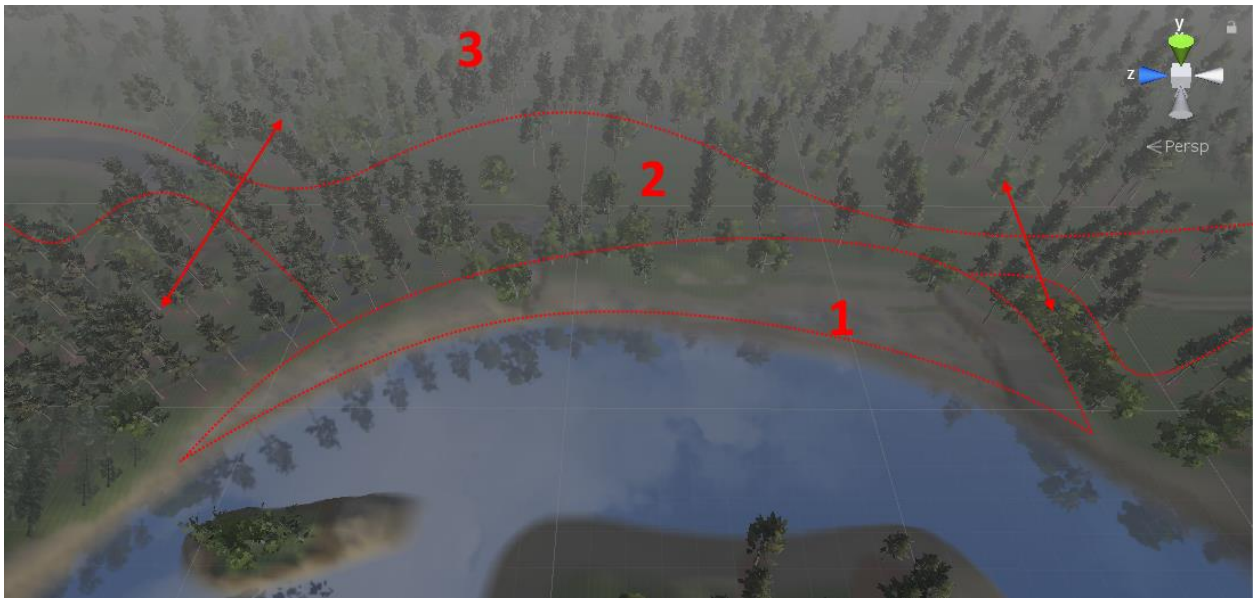

Рисунок 5.1 – Смоделированная в качестве основы для игровой локации местность

Постройки, ограждения и более мелкие объекты окружения и наполнения локации должны всячески подчеркивать или полностью формировать требуемые ключевые точки, необходимые для проведения сражений по реализованным игровым сценариям: как командные точки возрождения, так и объекты для штурма или размещения флагов. Таким образом, все объекты на созданной локации были сгруппированы определенным образом для удовлетворения данным требованиям.

Для наиболее качественного наполнения игровой локации сами постройки и предметы окружения были смоделированы в нескольких вариациях как самой модели, так и ее оформления (различные степени повреждения объекта; использованные для оформления цвета).

Также были созданы и размещены ограждения (как смоделированные, так и невидимые) для того, чтобы участники виртуальных сражений не имели возможности попасть за пределы непосредственно игровой локации, а выбывшие и ожидающие возрождения игроки – за пределы территории точки возрождения своей команды.

Таким образом, в рамках выполнения поставленных задач была создана игровая локация, проиллюстрированная на рис. 5.2 и соответствующая всем

требованиям, касающимся ее общей структуры и размещения на ней ключевых точек для проведения игр по определенным сценариям. Здесь: 1 – территория точки возрождения первой команды (еще и точка размещения флага в режиме Capturing flags); 2 – территория точки возрождения второй команды (еще и точка размещения флага в режиме Capturing flags); 3 – область проведения сражений; 4 – точка появления нейтрального флага (в режиме Capturing flag); 5 – область для захвата (в режиме Assault).

Рисунок 5.2 – Созданная игровая локация

5.3. Выводы

В рамках данного раздела ВКР были рассмотрены общие правила проведения страйкбольных состязаний, были описаны наиболее часто используемые на подобных мероприятиях игровые сценарии, также рассмотрена их реализация в разрабатываемой компьютерной игре.

На основе сформулированных требований и 6 определенных игровых режимов (сценариев) была создана локация с применением технических средств игрового движка Unity 3D по формированию ландшафта местности.

6. РАЗРАБОТКА GUI

В данном разделе описана общая архитектура и процесс реализации графического пользовательского интерфейса игры.

6.1. Формулировка требований к GUI. Его общая структура

Графический пользовательский интерфейс должен обеспечивать возможность доступа пользователя (игрока) ко всем аспектам функционала разрабатываемой игры: созданию игрового лобби в локальной сети, подключению к уже существующему, работе с игровым инвентарем, использованию магазина, возможности обзора пользовательских модулей и оружия, пользовательской кастомизации последнего.

Использование технологии UNET требует такой структуры организации игры, при которой она будет подразделена на офлайн и онлайн игровые сцены. Таким образом, на рис. 6.1 представлена UML-диаграмма архитектуры GUI с описанием назначений каждой входящей в него страницы.

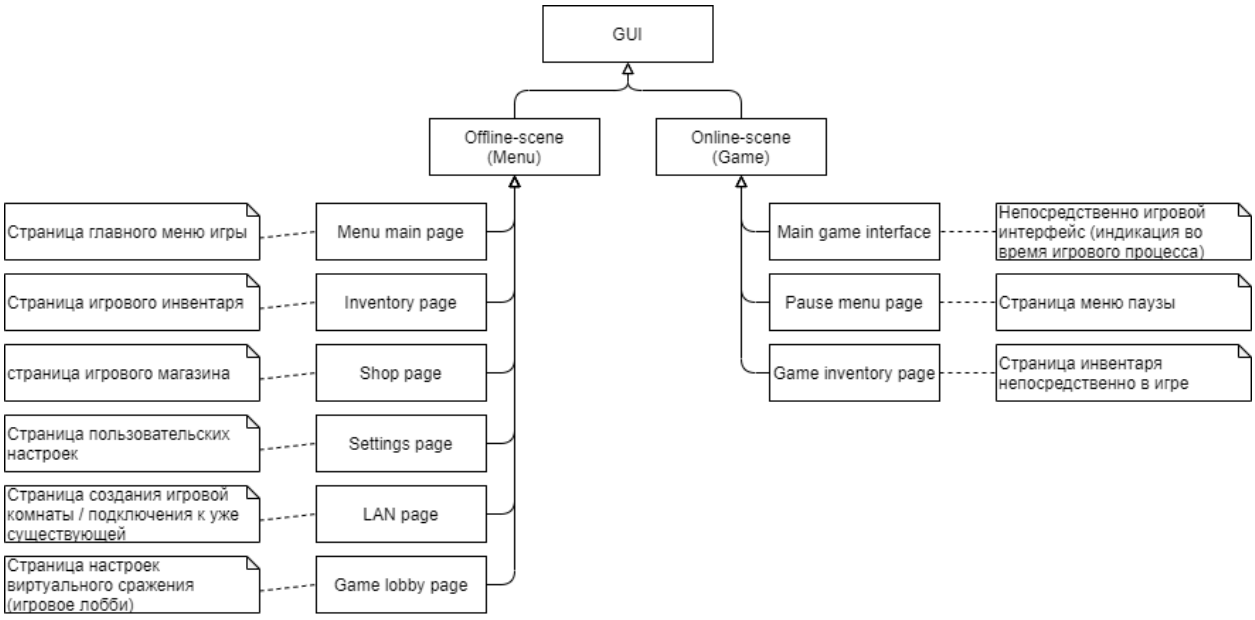

Рисунок 6.1 – UML-диаграмма архитектуры GUI разрабатываемой игры с описанием назначений каждой входящей в него страницы

6.2. Проектирование и реализация неигровой сцены

6.2.1. Проектирование и реализация страницы главного меню игры

После запуска игры открывается страница главного меню, расположенная на офлайн (неигровой) сцене. Она обеспечивает навигацию по остальным доступным страницам игрового меню. Внешний вид разработанной стартовой страницы игры (страницы главного меню) приведен на рис. 6.2. Здесь:

1. Логотип игры.
2. Показатели количества шаров (боеприпасов) и игровой валюты соответственно.
3. Подпись.
4. Кнопка начала игры (переход на страницу создания игрового лобби в локальной сети или подключения к уже существующему).
5. Кнопка перехода на страницу пользовательского игрового инвентаря.
6. Кнопка перехода на страницу игрового магазина.
7. Кнопка перехода на страницу настроек игры.
8. Кнопка выхода из игры (закрытия приложения).

Рисунок 6.2 – Внешний вид интерфейса страницы главного меню игры

6.2.2. Проектирование и реализация игрового лобби

Для начала игры необходимо из главного меню перейти на страницу, где пользователь (игрок) может создать игровое лобби в своей локальной сети или подключиться к уже существующему. Внешний вид данной страницы представлен на рис. 6.3. Здесь:

1. Логотип игры.
2. Показатели количества шаров (боеприпасов) и игровой валюты соответственно.
3. Подпись.
4. Кнопка создания игрового лобби в локальной сети, в которой находится пользовательский ПК (открывает страницу настроек игрового лобби).
5. Кнопка подключения к уже созданному в локальной сети игровому лобби по IP-адресу (значение по умолчанию – localhost, адрес пользовательского ПК); открывает страницу игрового лобби.
6. Кнопка возврата в главное меню.

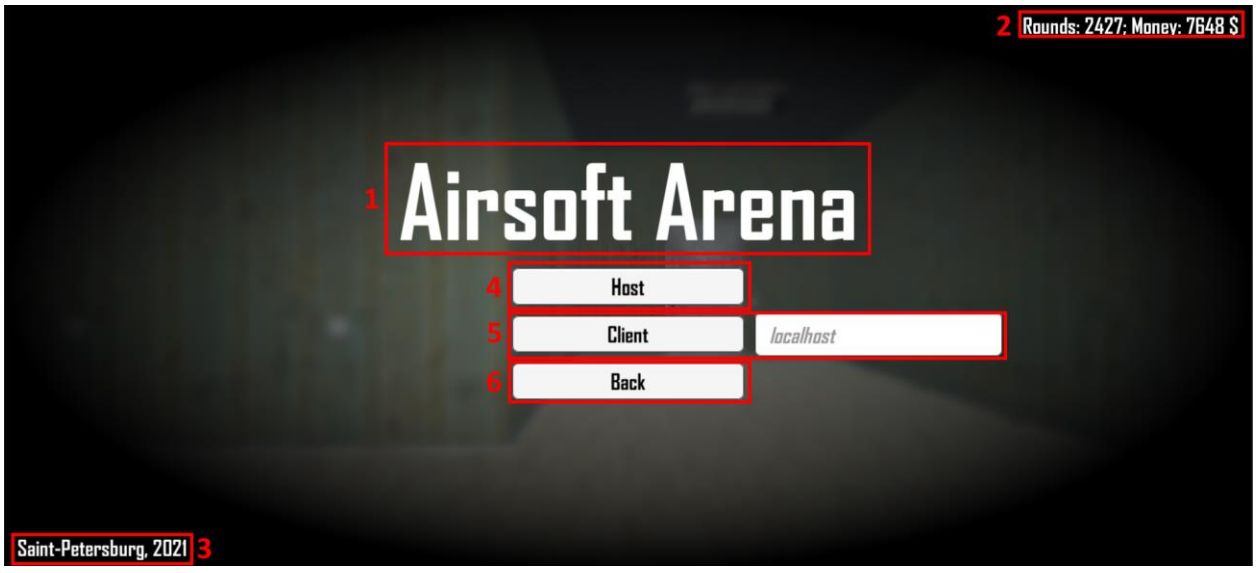

Рисунок 6.3 – Внешний вид интерфейса страницы локальной сети

Игровое лобби содержит инструмент для выбора режима проведения виртуального сражения, а также информацию о выбранном режиме и порядке

размещения подключенных игроков в обеих доступных командах. Внешний вид соответствующей страницы представлен на рис. 6.4. Здесь:

1. Показатели количества шаров (боеприпасов) и игровой валюты соответственно.
2. Окно первой команды.
3. Окно второй команды.
4. Пример занятой некоторым подключенным игроком позиции в команде.
5. Пример свободной позиции.
6. Выпадающий список для выбора игрового сценария (режима виртуального сражения; недоступен для изменения у игроков, не являющихся создателями лобби).
7. Описание выбранного игрового сценария.
8. Кнопка запуска сражения по выбранному игровому сценарию (доступна только у игрока, являющегося создателем лобби; запускает онлайн-сцену).
9. Кнопка отключения от игрового лобби и выхода в главное меню.
10. Подпись.

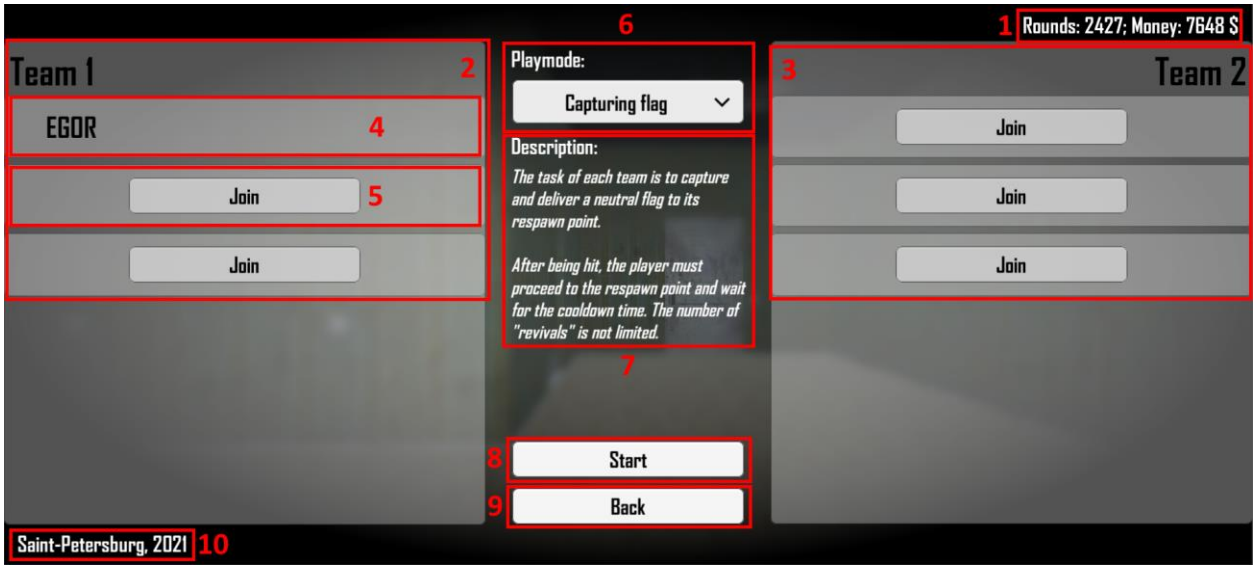

Рисунок 6.4 – Внешний вид интерфейса страницы настроек игрового лобби

6.2.3. Проектирование и реализация страницы игрового инвентаря

Страница игрового инвентаря доступна для перехода из главного меню. В ней реализованы функции по обзору текущего состояния хранилищ пользовательского инвентаря и элементы для манипуляции над находящимися в них предметами. Внешний вид соответствующей страницы приведен на рис. 6.5. Здесь:

1. Показатели количества шаров (боеприпасов) и игровой валюты соответственно.
2. Информация о текущем отображаемом предмете (наименование в игровом магазине, производитель, цены продажи и починки, технические характеристики).
3. Доступные действия над выделенной позицией на оружии (удаление или добавление модуля на выделенную позицию).
4. Доступные действия над выделенным предметом в зависимости от его категории: его перемещение между хранилищами, размещение в слоты для первичного или вторичного оружия, починка и продажа.
5. Обзорная область выделенного предмета (доступно для вращения; на оружии с помощью специальных указателей отмечаются наименования установленных на него модулей – выделено зеленым цветом, свободные позиции «Empty slot» выделены красным).
6. Текущее состояние игрового «склада» (хранилища storage). Все предметы, находящееся в нем, проиллюстрированы соответствующими иконками.
7. Текущее состояние слотов для первичного и вторичного оружия.
8. Текущее состояние загруженности инвентаря, которое вычисляется как сумма масс всех предметов из хранилища ammunition и масс первичного и вторичного оружия.
9. Текущее состояние игрового хранилища ammunition, предметы из которого могут использоваться во время проведения виртуальных сражений.

10. Кнопка перехода на страницу игрового магазина.
11. Кнопка выхода в главное меню.
12. Подпись.

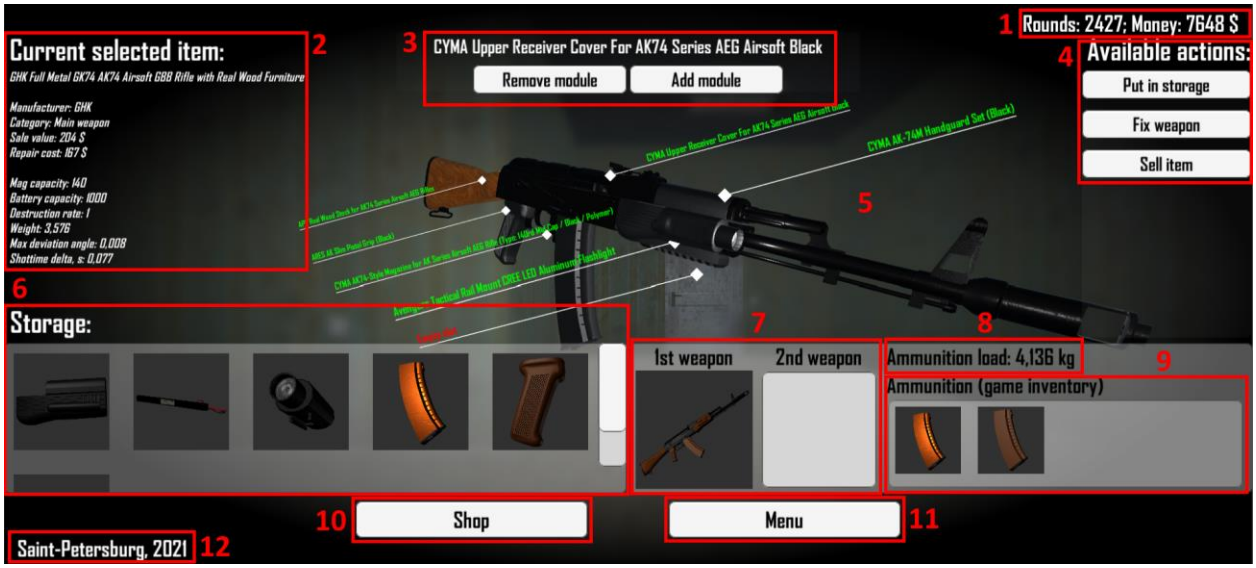

Рисунок 6.5 – Внешний вид интерфейса страницы игрового инвентаря

6.2.4. Проектирование и реализация страницы игрового магазина

Страница игрового магазина доступна для перехода из главного меню и инвентаря. На ней расположены элементы для ознакомления и покупки различных игровых предметов и боеприпасов. Внешний вид страницы представлен на рис. 6.6. Здесь:

1. Показатели количества шаров (боеприпасов) и игровой валюты соответственно.
2. Основное окно игрового магазина, в котором каждый продаваемый предмет проиллюстрирован соответствующей иконкой.
3. Информация о выбранном для покупки предмете (торговое наименование, производитель, цена покупки и технические характеристики).
4. Кнопка покупки выбранного предмета.
5. Меню пополнения количества боеприпасов (шаров) на 100, 1000 и 10000 штук соответственно.
6. Кнопка перехода в игровой инвентарь.

7. Кнопка возврата в главное меню.
8. Подпись.

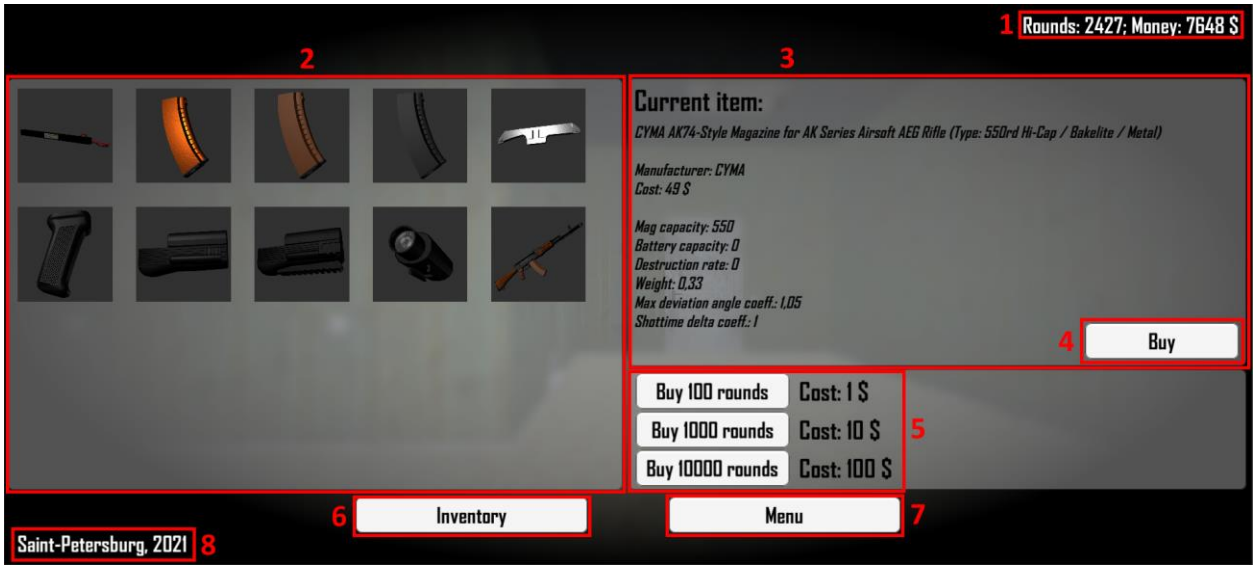

Рисунок 6.6 – Внешний вид интерфейса страницы игрового магазина

6.2.5. Проектирование и реализация меню настроек

Из главного меню также доступна для перехода страница с некоторыми игровыми настройками. Внешний вид страницы настроек приведен на рис. 6.7. Здесь:

1. Показатели количества шаров (боеприпасов) и игровой валюты соответственно.
2. Сетевые пользовательские настройки – указание имени пользователя, которое будет отображаться в игровом лобби.
3. Базовые графические настройки: выбор одной из предустановленных настроек качества (текстур, теней, освещения и т. п.) – доступно 6 вариантов: very low, low, medium, high, very high и ultra; включение или отключение оконного режима.
4. Кнопка сохранения настроек (здесь, как и при сохранении данных инвентаря, используется метод BinaryFormatter.Serialize() для записи в соответствующий файл settings.sv).
5. Кнопка выхода в главное меню.
6. Подпись.

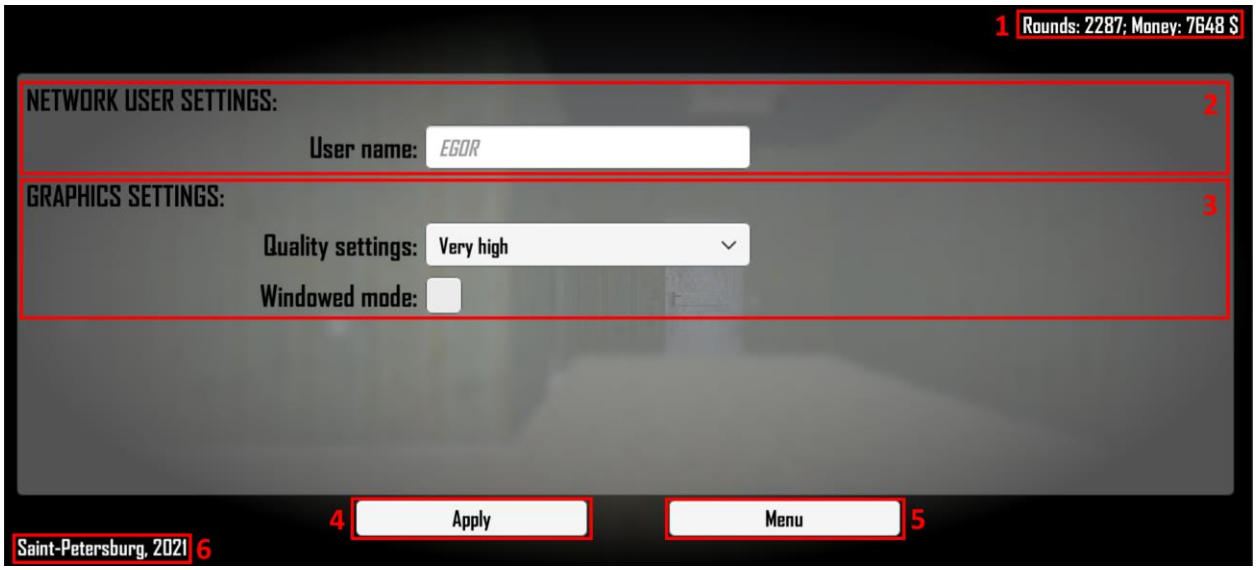

Рисунок 6.7 – Внешний вид интерфейса страницы игровых настроек

6.3. Проектирование и реализация игровой сцены

В целях усложнения игрового процесса, непосредственно в ходе игры у пользователя отсутствует индикация текущего состояния боезапаса, заряда аккумулятора и пр. Также отсутствует указатель прицела в центре экрана. Внешний вид непосредственно игрового интерфейса проиллюстрирован на рис. 6.8, где 1 – обзорная область; 2 – текущее состояние игрового статуса; 3 – доступность «подбора» игрового предмета; 4 – навигация по игровой локации.

Рисунок 6.8 – Внешний вид интерфейса непосредственно в процессе игры

В случае вызова меню паузы блокируются все возможные команды, связанные с персонажем. Рис. 6.9 – внешний вид меню паузы, где 1 – кнопка продолжения игры; 2 – кнопка разрыва соединения и выхода в главное меню.

Рисунок 6.9 – Внешний вид интерфейса меню паузы

На рис. 6.10 представлен внешний вид интерфейса непосредственно игрового инвентаря, где 1 – индикация игровых ресурсов; 2 – торговая информация о текущем выбранном предмете и его технические характеристики; 3 – кнопка «сброса» текущего предмета; 4 – текущее состояние слотов первичного и вторичного оружия и единственного доступного в процессе самой игры хранилища ammunition; 5 – текущая загруженность игрового инвентаря; 6 – доступные действия над выделенной позицией на оружии: удаление или добавление модуля на выделенную позицию; 7 – обзорная область текущего выбранного предмета, аналогична по своей структуре и функционалу обзорной области в инвентаре неигровой офлайн-сцены.

На рис. 6.11 представлен пример итогов игры, где 1 – общий командный результат (победа или поражение); 2 – заработок в игровой валюте, рассчитывающийся на основе совокупности личного и командного прогресса в зависимости от игрового сценария; 3 – количество попаданий по условным противникам; 4 – кнопка разрыва соединения и выхода в главное меню.

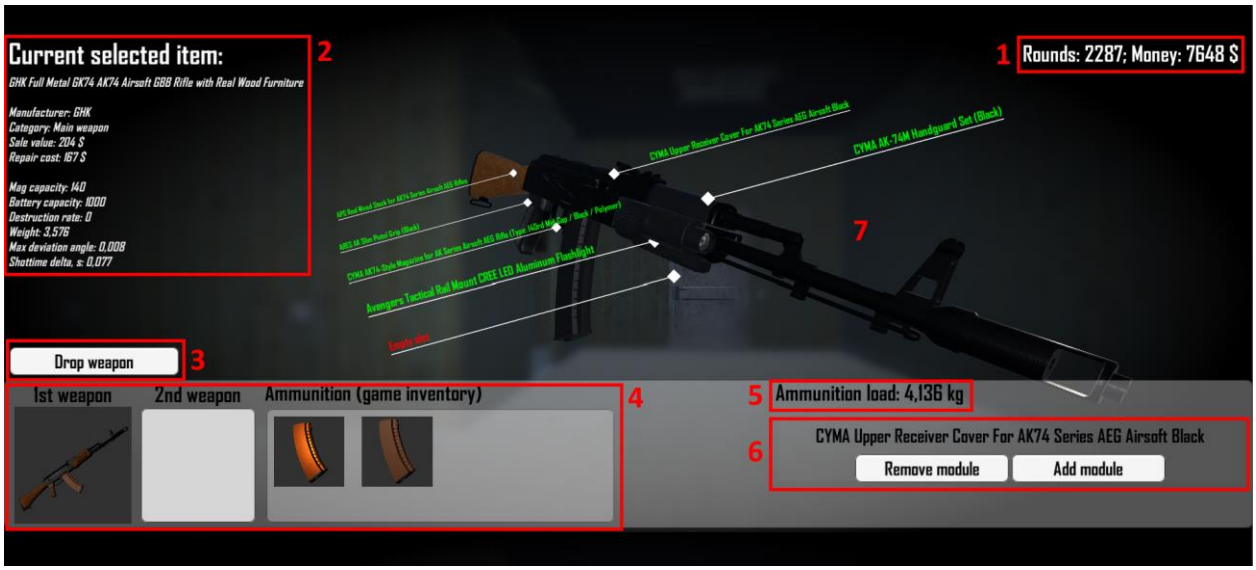

Рисунок 6.10 – Внешний вид интерфейса непосредственно игрового инвентаря

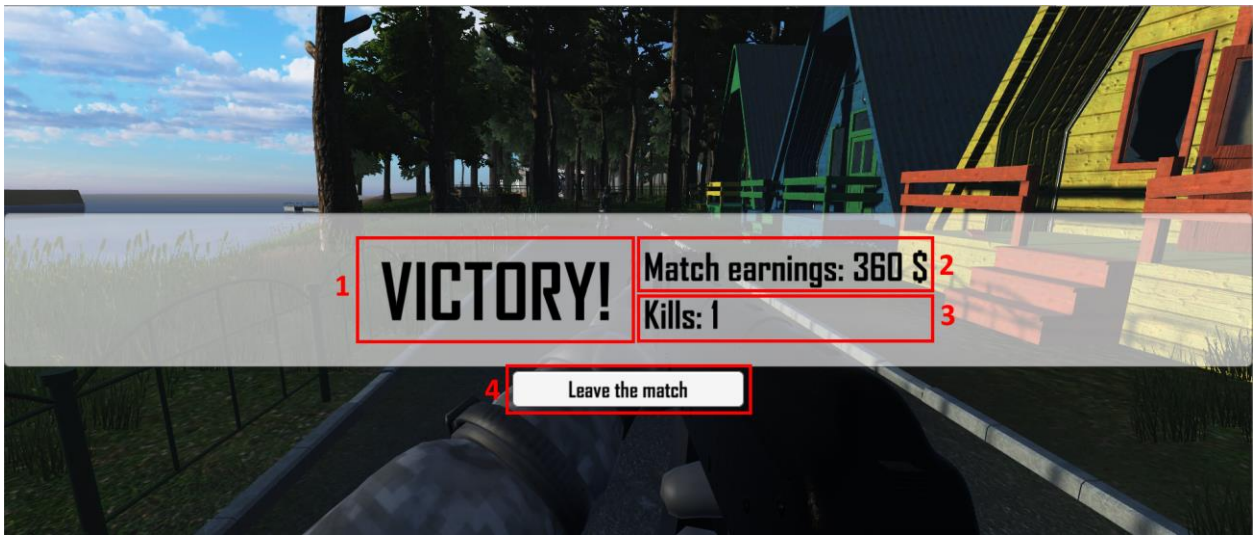

Рисунок 6.11 – Внешний вид окна итогов игры

6.4. Выводы

В данном разделе подробно описана структура игровых сцен (онлайн и офлайн), обусловленная использующейся системой для обеспечения сетевого взаимодействия игроков (UNET), а также структура и реализация всех связанных с рассмотренными сценами страниц созданного графического пользовательского интерфейса.

7. ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ ВКР

Результатом выполнения данной работы является шутер от первого лица, использующий тематику военно-тактической игры «страйкбол» (airsoft) для значительного расширения базовых принципов игр рассматриваемого жанра, что делает данную игру уникальной, инновационной и, как следствие, потенциально коммерчески успешной.

В данном разделе производятся расчеты расходов на основную и дополнительную заработную плату работников, на социальные отчисления, на материалы и на деятельность сторонних организаций. Также вычисляется величина амортизационных отчислений за период разработки и производится анализ аналогичных игровых приложений конкурентов.

7.1. Расчет затрат на оплату труда и накладных расходов

Сначала необходимо сформировать план проводимых работ, на основе которого определяется их трудоемкость, далее рассчитывается ставка заработной платы в день.

В Санкт-Петербурге и Ленинградской области открыто 55 вакансий по профессии «Программист стажер». Средняя заработная плата составляет 33542 рубля [15]. По профессии «Профессор» открыто 19 вакансий, средняя заработная плата при этом составляет 46354 рублей [15]. На основе этой информации рассчитываются дневные ставки программиста и профессора:

$$C_{\text{прог}} = \frac{33542 \text{ руб.}}{21 \text{ день}} \approx 1597.24 \text{ руб./день}$$

$$C_{\text{проф}} = \frac{46354 \text{ руб.}}{21 \text{ день}} \approx 2207.33 \text{ руб./день}$$

В табл. 7.1 и 7.2 соответственно приведены данные о трудоемкости выполнения работ, указанные в человеко-днях.

Таблица 7.1 – План работ программиста

№	Этапы и содержание выполняемых работ	Трудоемкость, чел./день	Ставка, руб./день
1	Составление ТЗ	1	1597.24
2	Анализ предметной области	8	1597.24
3	Разработка приложения	29	1597.24
4	Тестирование и отладка приложения	3	1597.24
5	Согласование с руководителем и исправление ошибок	5	1597.24
6	Оформление дипломного проекта	16	1597.24

Таблица 7.2 – План работ научного руководителя

№	Этапы и содержание выполняемых работ	Трудоемкость, чел./день	Ставка, руб./день
1	Составление ТЗ	1	2207.33
2	Согласование с руководителем и исправление ошибок	5	2207.33

Трудозатраты научного руководителя: $\mathrm{Т_{рук}} = 6$ чел./дней.

Трудозатраты программиста: $\mathrm{Т_{прог}} = 62$ чел./дней.

На основе данных о трудоемкости работ и ставок исполнителей, участвующих в проекте, рассчитываются расходы на заработную плату. Процент отчислений на социальные нужды 30,2%, а также процент дополнительной заработной платы 8,3%. Основная заработная плата рассчитывается по формуле: $\mathrm{З_{осн.з/пл}} = \mathrm{Т_{рук}} \cdot \mathrm{С_{рук}} + \mathrm{Т_{прог}} \cdot \mathrm{С_{прог}}$, где $\mathrm{Т}_i$ – время, затраченное i-м исполнителем на проведение работ (в днях); $\mathrm{С}_i$ – ставка i-го исполнителя (в руб./день).

$$\mathrm{З_{осн.з/пл}} = 2207.33 \cdot 6 + 1597.24 \cdot 62 \approx 112272.75 \text{ руб.}$$

Дополнительная заработная плата составляет:

$$\mathrm{З_{доп.з/пл}} = \mathrm{З_{осн.з/пл}} \cdot \frac{\mathrm{Н_{доп}}}{100} = 112272.75 \cdot 0.083 \approx 9318.63 \text{ руб.}$$

Отчисления на социальные нужды от основной и дополнительной заработной платы равны: $\mathrm{З_{соц}} = \left(\mathrm{З_{осн./пл}} + \mathrm{З_{доп.з/пл}}\right) \cdot \frac{\mathrm{Н_{соц}}}{100}$

$$З_{соц} = (112272.75 + 9318.63) \cdot 0.302 \approx 36720.6 \text{ руб.}$$

В данной работе норматив накладных расходов равен 20% от суммы основной и дополнительной заработной платы. Накладные расходы рассчитываются по формуле: $С_{нр} = \left(З_{осн./пл} + З_{доп.з/пл}\right) \cdot \frac{Н_{нак}}{100}$

$$С_{нр} = (112272.75 + 9318.63) \cdot 0.2 \approx 24318.28 \text{ руб.}$$

7.2. Расходы на материалы

Данные при расчете количества и стоимости материалов с учетом транспортно-заготовительных расходов занесены в табл. 7.3. К расходам на материалы относят затраты на основные и вспомогательные материалы. Норма транспортно-заготовительных расходов $Н_{т.з} = 10\%$.

Таблица 7.3 – Затраты на материалы

Материалы	Кол-во	Цена, руб.	Сумма, руб.
Бумага офисная SvetoCopy A4, 500 листов [16]	1	295	295
Картридж для принтера Canon PGI-450PGBK XL [17]	1	1450	1450
Набор картриджей для принтера Canon CLI-451 C/M/Y/BK [18]	1	3549	3549
ИТОГО:			5294
Транспортно-заготовительные расходы (10%)			529.4
ВСЕГО:			5823.4

7.3. Издержки на амортизацию ПК и оргтехники

В процессе разработки игры использовался ПК и принтер. Стоимость ноутбука Lenovo IdeaPad Y700 14 – 54775.21 рубля [19], а стоимость принтера Canon PIXMA iP7240 составляет 4748 рублей [20].

Сумма амортизации за год вычисляется по формуле: $А_г = К_{об} * \frac{Н_{АМ}}{100}$, где $К_{об}$ – стоимость оборудования; $Н_{АМ}$ – норма амортизации.

Согласно постановлению правительства РФ от 01.01.2002 №1 (ред. от 28.04.2018) «О Классификации основных средств, включаемых в

амортизационные группы», персональные компьютеры и печатающие устройства к ним относятся ко второй группе. Срок полезного использования оборудования равен от двух до трех лет включительно. Срок эксплуатации три года [21]. Норма амортизации $Н_{АМ}$ = 33,3%.

$$А_г = (54775.21 + 4748) \cdot \frac{33.3}{100} \approx 19821.23 \text{ руб.}$$

Сумма амортизации за рабочий день равна: $А_д = \frac{А_г}{N}$. В 2021 году 247 рабочих дней, соответственно: $А_д = \frac{19821.23}{247} \approx 80 \text{ руб. } 25 \text{ коп.}$ Амортизация оборудования за время проектирования дипломного проекта:

$$А_{вкр} = А_д \cdot 62 \approx 4975 \text{ руб. } 37 \text{ коп.}$$

7.4. Расходы на услуги сторонних организаций

В процессе разработки использовалась бесплатная Personal версия движка Unity 3D, а также интегрированная среда разработки Visual Studio 2017, которая также распространяется бесплатно.

Для разработки игры был необходим доступ в интернет, поэтому была подключена соответствующая интернет услуга компании ООО «Ростелеком», стоимость которой составляет 500 рублей в месяц. В табл. 7.4 приведены затраты по работам, выполняемым сторонними организациями.

Таблица 7.4 – Затраты по работам, выполняемым сторонними организациями

Наименование	Кол-во	Цена, руб.	Сумма, руб.
Доступ в интернет, мес.	3	500	1500
ИТОГО:			1500

7.5. Себестоимость выполнения ВКР

Себестоимость ВКР по статьям расхода и в целом указана в таблице 7.5. Таким образом, себестоимость ВКР оценивается в 194929 руб. 4 коп.

Таблица 7.5 – Смета затрат на ВКР

№ п/п	Наименование статьи	Сумма, руб.	Структура себестоимости, %
1	Материалы	5823.4	2.99
2	Расходы на оплату труда	121591.39	62.38
3	Отчисления на социальные нужды	36720.6	18.84
4	Издержки на амортизацию ПК и оргтехники	4975.37	2.55
5	Расходы на услуги сторонних организаций	1500	0.77
6	Накладные расходы	24318.28	12.48
ИТОГО:		194929.04	100

7.6. Анализ конкурентов

Далее был проведен анализ конкурентов, разрабатывающих аналогичные игры для ПК жанра «шутер от первого лица». Результаты проведенного анализа представлены в табл. 7.6.

Counter Strike: Global Offensive [4] поддерживает наибольшее количество языков, имеет достаточное число доступных игровых сценариев (режимов) и простые игровые механики для проведения виртуальных сражений; поддерживает несколько операционных систем. Для удержания уровня популярности проекта (примерно 24 млн активных пользователей ежемесячно) распространяется бесплатно.

Escape from Tarkov [5] обладает самой продуманной системой пользовательской кастомизации игрового оружия, однако игра узко локализована, реализована только для операционной системы Windows, также имеет высокий возрастной рейтинг из-за наличия сцен насилия и использования персонажами нецензурной лексики. Имеет несколько завышенную цену.

S.T.A.L.K.E.R.: Зов Припяти [6] обладает продуманной системой игрового инвентаря, более мягким возрастным рейтингом и большим количеством поддерживаемых языков, однако также реализован только для операционной системы Windows.

Metro Exodus [7] поддерживает несколько операционных систем и языков интерфейса, однако обладает высокой ценой и возрастным рейтингом.

Разрабатываемая игра, благодаря инструментам сборки в Unity 3D, поддерживает сразу несколько операционных систем, однако ее интерфейс представлен только одним языком – английским, так как одним из обязательных требований для игр при размещении на самых популярных платформах (Steam; Epic Games Store) является наличие именно английской локализации. В следствие использования не огнестрельного оружия в рамках моделирования состязаний военно-тактической игры «страйкбол», в игре отсутствуют сцены насилия, что обеспечивает самый низкий из всех рассмотренных шутеров возрастной рейтинг по системе PEGI [22].

Таблица 7.6 – Анализ конкурентов

Наименование критерия	Разрабатываемое решение	Counter Strike: Global Offensive	Escape from Tarkov	S.T.A.L.K.E.R.: Зов Припяти	Metro Exodus
Операционные системы	Windows, Mac OS, Linux	Windows, Mac OS, Linux	Windows	Windows	Windows, Mac OS, Linux
Локализация, количество языков	1	28	4	8	14
Возрастной рейтинг	7+	18+	18+	16+	18+
Цена, руб.	1024.5	0	1600	899	1599

Цену на разрабатываемую игру формировалась на основе анализа аналогов при задании весов ценам аналогичных игр по степени приближенности:

$$P = 0.25 \cdot 0 + 0.25 \cdot 1600 + 0.25 \cdot 899 + 0.25 \cdot 1599 = 1024.5 \text{ руб.}$$

В такой случае, минимальный объем продаж, который позволит окупить затраты на разработку, составляет:

$$Q = \frac{194929.04}{1024.5} \approx 190 \text{ копий}$$

7.7. Выводы

В данном разделе в рамках экономического обоснования ВКР для разрабатываемого решения были рассчитаны расходы на основную и дополнительную заработные платы работников, на социальные отчисления, на материалы и на деятельность сторонних организаций, а также вычислена величина амортизационных отчислений за весь период разработки игры. Затем были проанализированы аналогичные игровые приложения конкурентов.

Исходя из расчета, себестоимость конечного продукта составляет 194929.04 рубля. Основную часть затрат составляют расходы на оплату труда (62.38%), отчисления на социальные нужды (18.84%) и накладные расходы (12.48%). Цена одной копии игрового приложения составляет 1024.5 рубля. Минимальный объем продаж, который позволяет окупить затраты на разработку игры равен 190 копий.

ЗАКЛЮЧЕНИЕ

В рамках выполнения данной работы была создана компьютерная игра, принадлежащая к жанру «шутер от первого лица» и моделирующая проведение страйкбольных военно-тактических игр с использованием электропневматического оружия, вместо огнестрельного, для ПК (операционные системы Windows, Linux и Mac OS) на игровом движке Unity 3D. Использование подобного оружия обусловливает необходимость контроля сразу за несколькими показателями, и применение к нему модульной кастомизации (представление игрового оружия в качестве многоуровневой иерархической системы модулей и обеспечение возможности внесения изменений в его конфигурацию).

Для достижения поставленной цели был проведен анализ предметной области, в ходе которого был рассмотрен ряд аналогов – других игр данного жанра, наиболее близких к разрабатываемой в плане использованных механик. На основе выявленных у рассмотренных аналогов достоинств и недостатков были сделаны соответствующие выводы касательно разрабатываемого решения, сформулированы подробные требования к каждому этапу разработки и выбраны методы решения соответствующих задач на каждом из этапов.

Таким образом, в разрабатываемой игре был реализован персонаж с возможностью перемещения по локации и использования игрового оружия (все действия персонажа иллюстрируются соответствующими анимациями); инвентарь с возможностью модульной кастомизации игрового оружия; были определены и реализованы 6 игровых сценариев для проведения виртуальных сражений по локальной сети; создана локация, позволяющая проводить состязания по любому из сценариев. Графический пользовательский интерфейс предоставляет игрокам возможность взаимодействия со всем реализованным функционалом.

Созданный в ходе выполнения ВКР шутер от первого лица значительно расширил базовые принципы игр данного жанра, касающиеся игрового процесса. Однако необходимо продолжать исследования в данной области. Дальнейшие исследования могут быть связаны как с наполнением уже разработанной игры (добавление новых образцов игрового оружия и модулей, новых локаций и пр.), так и с созданием новых «экспериментальных» шутеров, моделирующих проведение других командных военно-тактических игр (Пейнтбол, Лазертаг), в т. ч. и с возможностью применения разработанной в ходе данной ВКР системы модульной кастомизации к используемому игровому оружию (Фаертаг), которое, в свою очередь, будет отличаться в плане специфики использования и принципа работы от общепринятого в играх рассматриваемого жанра огнестрельного.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Video game sales in the United States in 2018, by genre [Электронный ресурс]. URL: https://www.statista.com/statistics/189592/breakdown-of-us-video-game-sales-2009-by-genre/ (дата обращения: 06.04.2021).
2. Headshot: A visual history of first-person shooters [Электронный ресурс]. URL: https://arstechnica.com/gaming/2016/02/headshot-a-visual-history-of-first-person-shooters/ (дата обращения: 06.04.2021).
3. Лучшие игры с кастомизацией оружия [Электронный ресурс]. URL: https://cubiq.ru/luchshie-igry-s-kastomizatsiej-oruzhiya/ (дата обращения: 06.04.2021).
4. Counter-Strike: Global Offensive [Электронный ресурс]. URL: https://store.steampowered.com/app/730/CounterStrike_Global_Offensive/ (дата обращения: 06.04.2021).
5. Escape from Tarkov [Электронный ресурс]. URL: https://www.escapefromtarkov.com/?lang=ru (дата обращения: 06.04.2021).
6. S.T.A.L.K.E.R.: Call of Pripyat [Электронный ресурс]. URL: https://store.steampowered.com/app/41700/STALKER_Call_of_Pripyat/ (дата обращения: 06.04.2021).
7. Metro: Exodus [Электронный ресурс]. URL: https://store.steampowered.com/app/412020/Metro_Exodus/ (дата обращения: 06.04.2021).
8. Unity Documentation [Электронный ресурс]. URL: https://docs.unity3d.com/ru/2019.4/Manual/ (дата обращения: 09.05.2021).
9. How Do Bullets Work in Video Games [Электронный ресурс]. URL: https://medium.com/@3stan/how-do-bullets-work-in-video-games-d153f1e496a8 (дата обращения: 12.05.2021).
10. Страйкбольный автомат (Cyma) CM040C S-74MN [Электронный ресурс]. URL: https://airsoftsports.ru/products/model-av-oma-a-cyma-cm040c-aks-74mn/ (дата обращения: 12.05.2021).

11. 5,45-мм автомат Калашникова АК-74М [Электронный ресурс]. URL: https://structure.mil.ru/structure/forces/ground/weapons/arma/more.htm?id=10344196@morfMilitaryModel (дата обращения: 12.05.2021).
12. Полигоны для страйкбола (СПб и ЛО) [Электронный ресурс]. URL: http://airsoftpiter.ru/wiki/%D0%9A%D0%B0%D1%82%D0%B5%D0%B3%D0%BE%D1%80%D0%B8%D1%8F:%D0%9F%D0%BE%D0%BB%D0%B8%D0%B3%D0%BE%D0%BD%D1%8B_%D0%B4%D0%BB%D1%8F_%D1%81%D1%82%D1%80%D0%B0%D0%B9%D0%BA%D0%B1%D0%BE%D0%BB%D0%B0_(%D0%A1%D0%9F%D0%B1_%D0%B8_%D0%9B%D0%9E) (дата обращения: 17.05.2021).
13. Правила страйкбола [Электронный ресурс]. URL: http://airsoftgun.ru/rules (дата обращения: 17.05.2021).
14. Сценарии страйкбольных игр [Электронный ресурс]. URL: https://aoharuxmachinegun.fandom.com/ru/wiki/%D0%A1%D1%86%D0%B5%D0%BD%D0%B0%D1%80%D0%B8%D0%B8_%D1%81%D1%82%D1%80%D0%B0%D0%B9%D0%BA%D0%B1%D0%BE%D0%BB%D1%8C%D0%BD%D1%8B%D1%85_%D0%B8%D0%B3%D1%80 (дата обращения: 17.05.2021).
15. Статистика зарплат в Санкт-Петербурге [Электронный ресурс]. URL: https://sankt-peterburg.trud.com/salary/865.html (дата обращения: 22.05.2021).
16. Бумага SvetoCopy Classic A4 500 шт [Электронный ресурс]. URL: https://www.onlinetrade.ru/catalogue/bumaga_dlya_ofisnoy_pechati-c1585/international_paper/bumaga_svetocopy_a4_80g_m2_146_cie_500l_ipsc a4-72421.html (дата обращения: 22.05.2021).
17. Картридж Canon PGI-450PGBK XL черный пигмент (pigment black) повышенной емкости (6434B001) [Электронный ресурс]. URL: https://www.onlinetrade.ru/catalogue/kartridzhi_dlya_struynoy_pechati-c70/canon/kartridzh_canon_pgi_450pgbk_xl_chernyy_pigment_pigment_black

_t_povyshennoy_emkosti_6434b001-106192.html (дата обращения: 22.05.2021).

18. Комплект картриджей для струйного принтера Canon CLI-451C/M/Y/BK (6524B004), черный, голубой, пурпурный, желтый [Электронный ресурс]. URL: https://www.ozon.ru/context/detail/id/147435712/ (дата обращения: 22.05.2021).
19. Ноутбук Lenovo IdeaPad Y700 14 [Электронный ресурс]. URL: https://www.e-katalog.ru/LENOVO-Y700-14ISK-80NU0029US.htm (дата обращения: 22.05.2021).
20. Принтер Canon Pixma iP7240 [Электронный ресурс] URL: https://www.ozon.ru/product/printer-canon-pixma-ip7240-struynyy-20103138/?utm_source=google&utm_medium=cpc&utm_campaign=2_SPB_Product_DSA_NewClients&gclid=Cj0KCQjw16KFBhCgARIsALB0g8JiqjNWlF1FXZLwkOuZJEsL91oqSJIekbHpMc3W_1Rc3KufoPC-pZYaAr-kEALw_wcB (дата обращения: 22.05.2021).
21. О классификации основных средств, включаемых в амортизационные группы: постановление Правительства Рос. Федерации от 01.01.2002 №1 (ред. от 28.04.2018) // Собрание законодательства Российской Федерации.
22. Возрастные метки PEGI [Электронный ресурс]. URL: https://pegi.info/ (дата обращения: 23.05.2021).