МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное
учреждение высшего образования
«Южно-Уральский государственный университет»
(национальный исследовательский университет)
Высшая школа экономики и управления
Кафедра «Информационные технологии в экономике»

ДОПУСТИТЬ К ЗАЩИТЕ
Зав. кафедрой, д.т.н., с.н.с.
_____/ Б.М. Суховилов /
«_____»_____20____ г.

Разработка компьютерной игры в жанре альтернативной истории с элементами интеграции сторонних пользователей

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 09.03.03.2021.028.ВКР

Руководитель, доцент
_____/ Е.М. Сартасов /
«_____»_____2021 г.

Автор
студент группы ЭУ-402
_____/ Е.А. Дружинин /
«_____» _____2021 г.

Нормоконтролер, доцент
_____/Е.А. Конова/
«_____»_____2021 г.

Челябинск 2021

АННОТАЦИЯ

Дружинин Е.А. Разработка компьютерной игры в жанре альтернативной истории с элементами интеграции сторонних пользователей.– Челябинск: ЮУрГУ, ЭУ-402, 51 с., 20 ил., 12 табл., библиогр. список – 12 наим., 1 прил.

Дипломный проект выполнен с целью разработки компьютерной игры в жанре альтернативной истории с элементами интеграции сторонних пользователей.

В дипломном проекте проведён сравнительный анализ компьютерных игр в жанре Roguelite с целью определения наиболее распространенных геймдизайнерских решений, необходимого функционала, принципов и приемов дизайна в построении пользовательского интерфейса. По рассмотренным существующим решениям, сделан вывод, на основе которого построен план реализации компьютерной игры в данной предметной области.

С помощью игрового движка Unity, языка программирования C#, библиотеки TwitchLib, реализована компьютерная игра «Steampunk», выступающая в жанре «Roguelite» и одноименном литературном жанре альтернативной истории «Steampunk». В дополнении к игровому процессу осуществлена интеграция сторонних пользователей через интернет-сервис прямых трансляций Twitch.

Полученные данные, в ходе исследования и разработки компьютерной игры позволят внести вклад в развитие жанра Roguelite и в развитие концепции внедрения сторонних пользователей в игровой процесс.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Компьютерные игры давно уже стали неотъемлемой частью досуга многих людей в мире, начиная от самых простых мобильных игр в казуальных жанрах до сложных AAA-проектов (неформальный термин, обозначающий класс высокобюджетных игр), разрабатываемых компаниями в течение долгого времени.

Индустрия компьютерных игр довольно обширна и находится в постоянном развитии. Чтобы заинтересовать потенциальных игроков в своем продукте, разработчики и геймдизайнеры зачастую используют уже готовые, проверенные временем пути создания игры. Однако чтобы выделиться на фоне других проектов, требуются уникальные решения, эксперименты в смешивании разных жанров. Одним из таких уникальных решений является интеграция с различными видеостриминговыми сервисами, такими как Twitch, YouTube и т.д.

Данные сервисы специализируются на предоставлении пользователям возможности проводить потоковые трансляции в режиме реального времени с минимальным количеством требуемых инструментов. Чтобы запустить прямой эфир, человеку могут потребоваться веб-камера, микрофон, программное обеспечение для захвата изображения и высокоскоростное подключение к сети Интернет.

Интеграция со зрителями прямого эфира осуществляется посредством создания интернет-соединения между интерфейсом сервиса и компьютерной игрой.

Цель выпускной квалификационной работы – разработать компьютерную игру в жанре альтернативной истории с элементами интеграции сторонних пользователей в игровой процесс.

Задачи работы.

1. Выполнить сравнительный анализ аналогов – популярных компьютерных игр в выбранном жанре.

2. Рассмотреть интернет-сервисы, специализирующиеся на прямых трансляциях в тематике компьютерных игр, и их функционал.

3. Выбрать игровой движок для разработки, и определить дизайн пользовательского интерфейса.

4. Разработать компьютерную игру;

5. Отладить и протестировать разработанную игру.

6. Подготовить проект к выпуску.

Объект исследования – процесс разработки компьютерной игры с внедрением сторонних пользователей интернет-сервисов.

Разработанная компьютерная игра может быть применена как для проведения досуга одного пользователя, так и для развлечения и увеличения аудитории стримеров, проводящих прямые трансляции в сети Интернет.

1 ПОСТАНОВКА ЗАДАЧИ И АНАЛИЗ СУЩЕСТВУЮЩИХ РЕШЕНИЙ

1.1 Описание выбранного жанра

Во время концептуальной разработки игры, когда формировались основные идеи и способы их реализации, выбран жанр компьютерной игры под названием Roguelite. Он является небольшим ответвлением от жанра Roguelike, который имеет определенные характерные особенности, такие как генерируемые случайным образом уровни, враги, предметы и необратимость смерти персонажа. В случае гибели, игрок не может загрузить игру перед моментом своей смерти и должен начать свое прохождение заново.

Ключевые факторы, позволяющие определить игру, как Roguelike.

1. Игра должна быть пошаговой, каждая команда должна соответствовать одному действию и одному ходу.
2. Игровые уровни должны генерироваться случайным образом, будучи уникальными для каждого прохождения.
3. Игра должна содержать «перманентную смерть», не позволяя игроку продолжить прохождение после гибели персонажа.
4. Игра должна иметь единый режим и единый набор команд для всех игровых ситуаций, не допуская каких-либо дополнительных меню, головоломок или мини-игр.
5. Игра должна предоставлять игроку не какой-то единый линейный путь, а свободу с множеством вариантов прохождения.
6. Игрок должен самостоятельно исследовать найденные предметы и открывать их свойства.

Этот перечень факторов выбрали в 2008 году на International Roguelike Development Conference в Берлине как определяющий жанр.

Однако в отличие от своего предка, жанр Roguelite имеет некоторые отличительные особенности. Это направление игростроя характеризуется некоторым осовремениванием и упрощением каноничного Roguelike. Неизменными особенностями обычно остаются перманентная смерть и процедурная генерация уровней, остальное ограничивается лишь полетом мысли разработчика. Механика игры может быть любой: от классической пошаговой в духе жанра Roguelike до платформера или ритм-игры.

1.2 Анализ и сравнение существующих игр в выбранном жанре

В настоящее время существует немало игр Roguelite, большинство из них содержат в себе помесь других жанров, создавая уникальный опыт, как для игрока, так и для разработчика. Эксперименты с различными игровыми механиками позволяют выявить сильные и слабые стороны игры, одни уникальные реализации отсеиваются, другие остаются. Данный процесс можно даже в какой-то степени сравнить с эволюцией, где в итоге выживает сильнейшая особь, а менее приспособленная погибает. Так и происходит с идеями во время разработки.

Для создания концепта игровой механики, иногда приходится жертвовать ключевыми факторами жанра Roguelike, например перманентной смертью, пошаговостью процесса или отсутствием дополнительных интерфейсов и головоломок.

В качестве объектов сравнения выбраны следующие компьютерные игры.

The Binding of Isaac: Rebirth

Случайно сгенерированный ролевой шутер с тяжелыми элементами Roguelite. Интерфейс приведен на Рисунок 1.

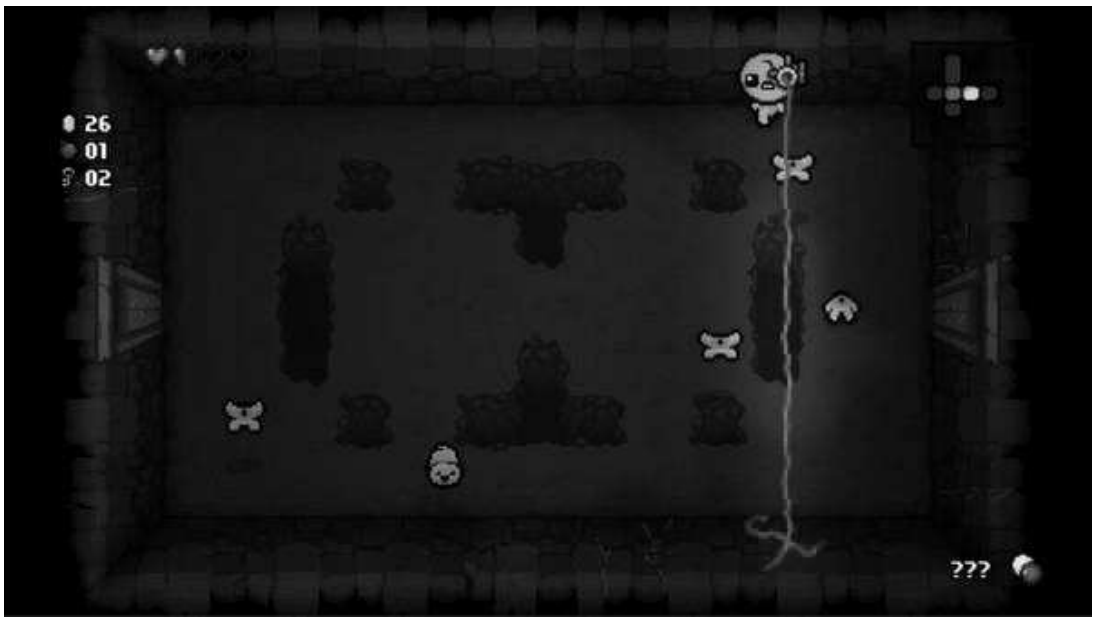

Рисунок 1 – Компьютерная игра The Binding of Isaac: Rebirth

Следуя за Исааком в его путешествии, игроки будут искать причудливые сокровища, которые изменяют форму Исаака, давая ему сверхчеловеческие способности и позволяя ему отбиваться от толп таинственных существ, открывать секреты и пробиваться к безопасному пути дальше по подземелью.

Hades

Это компьютерная игра в жанрах Roguelite и Action/RPG, сюжет которой основан на древнегреческой мифологии. Интерфейс приведен на Рисунок 2.

Рисунок 2 – Компьютерная игра Hades

Главным героем является Загрей, сын Аида. Он пытается сбежать из подземного царства мертвых и добраться до горы Олимп. Во время его странствий ему помогают боги-олимпийцы, посылающие Загрею те или иные дары.

Enter the Gungeon

Игра жанра «пулевая завеса в подземелье». Интерфейс приведен на Рисунок 3.

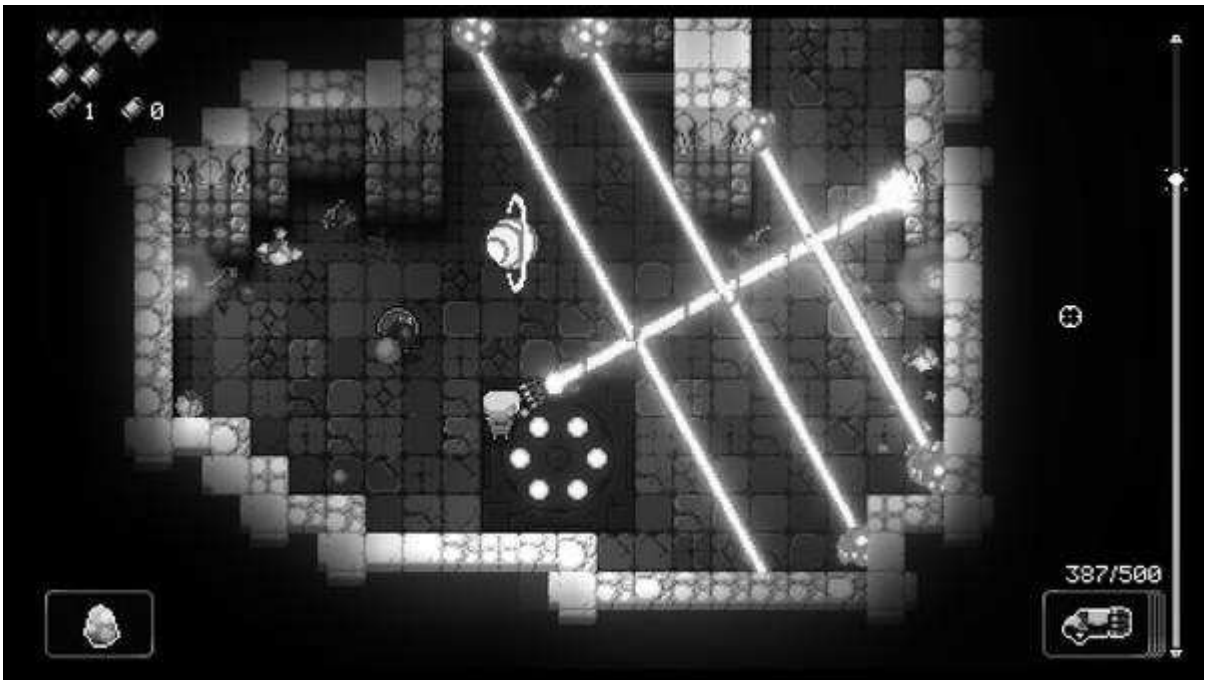

Рисунок 3 – Компьютерная игра Enter the Gungeon

В Enter the Gungeon ищущие спасения неудачники стреляют, грабят, перекатываются и опрокидывают столы, чтобы добраться до легендарного сокровища Оружелья: оружия, которое может убить прошлое. Для этого им нужно суметь остаться в живых на каждом из сложных уровней, где будут встречаться толпы жутко симпатичных оживших мертвецов и грозных, вооруженных до зубов боссов.

Noita

Магический боевик с элементами «Roguelike» в мире с физическим моделированием каждого пикселя. Интерфейс приведен на Рисунок 4.

Рисунок 4 – Компьютерная игра Noita

Игроку предстоит сражаться с врагами, исследовать различные места, расплавлять, сжигать, замораживать и испарять препятствия на пути по процедурно генерируемому миру, используя самостоятельно созданные заклинания. В поисках неизвестных тайн игрок будет погружаться всё глубже – в самые разные места: от угольных шахт до замерзших пустынь.

FTL: Faster Than Light

В FTL игроку предстоит взять на себя командование космическим кораблем, чтобы спасти галактику. Интерфейс приведен на Рисунок 5.

Рисунок 5 – Компьютерная игра FTL: Faster Than Light

Перед игроком окажется опасное путешествие, каждый шаг которого станет уникальной задачей с несколькими вариантами решения. Этот «двумерный симулятор космического корабля» отправит корабль и экипаж игрока в полное славы и горьких поражений путешествие через случайным образом генерируемую галактику.

Задачи анализа

1. Определить критерии сравнения выбранных компьютерных игр.
2. Провести оценку компьютерных игр по критериям сравнения.
3. Выявить сильные и слабые стороны реализации выбранных компьютерных игр по обозначенным направлениям.
4. Сделать выводы по результатам сравнения компьютерных игр.

Критерии сравнения основных характеристик компьютерных игр

1. Наличие генерации уровней случайным образом.
2. Наличие «перманентной смерти», не позволяющей игроку продолжить прохождение после гибели персонажа.
3. Различные способы и варианты прохождения.
4. Самостоятельное исследование игроком игровых предметов и их свойств.

5. Наличие интеграции сторонних пользователей через сторонние интернет-сервисы.

Условные обозначения:

- «+» – реализация решения удовлетворяет критерию;
- «+/–» – реализация решения не удовлетворяет критерию;
- «–» – отсутствие реализации решения.

Результаты исследования основных характеристик компьютерных игр представлены в **Ошибка! Источник ссылки не найден.**.

Таблица 1 – Основные характеристики компьютерных игр

Название игры	The Binding of Isaac: Rebirth	Hades	Enter the Gungeon	Noita	FTL: Faster Than Light
Наличие генерации уровней случайным образом	+	–	+	+	+
Наличие «перманентной смерти»	+	+	+	+	+
Различные способы и варианты прохождения	+	+	+	+	+
Самостоятельное исследование игроком игровых предметов и их свойств	+	+	+	+	+
Наличие интеграции сторонних пользователей через сторонние интернет-сервисы	–	–	–	–	–
Удобный и понятный интерфейс	+	+/–	+	+	–

Анализ проведенного сравнения компьютерных игр

Проведенное сравнение компьютерных игр в жанре Roguelite, позволяет сделать следующие выводы.

1. Формат реализации игр. Все предоставленные на анализ проекты имеют основополагающие ключевые факторы жанров Roguelite и Roguelike, такие как наличие «перманентной» смерти, различные способы прохождения и самостоятельное исследование игроком игровых предметов и их свойств.

2. Из рассмотренных игр четыре из пяти имеют полноценную случайную генерацию уровней.

3. У всех игр отсутствует какая-либо интеграция сторонних пользователей через сторонние интернет-ресурсы.

4. У трех игр из пяти присутствует удобный и очевидный для пользователя интерфейс. Это означает, что игрок сможет быстро адаптироваться и войти в игру, не запутавшись в UI.

Выбрав все сильные стороны и обозначив отрицательные моменты, во время разработки компьютерной игры стоит обратить внимание на:

- основополагающие факторы жанра Roguelite – случайную генерацию уровней, врагов, игровых предметов;
- удобный пользовательский интерфейс, не загроможденный различными цифрами статистики;
- реализацию интеграции сторонних пользователей интернет-ресурсов в игровой процесс. По компьютерным играм, которые были представлены в сравнении выше, можно понять, что данная задача ранее ещё не рассматривалась игровыми разработчиками как один из элементов геймдизайна для жанра Roguelite, что может означать актуальность этой проблемы в игровой сфере.

1.3 Инструменты и интернет-сервисы для создания программного продукта в предметной области

1.3.1 Выбор интернет-ресурса для создания интеграции с компьютерной игрой

Существует немало площадок, где пользователи могут проводить прямые трансляции (далее – стримы) для своих зрителей. Стримы проводятся в совершенно различных форматах, начиная от прогулки по улице и общения с людьми в чате до трансляции киберспортивных турниров и игры на музыкальных инструментах.

В качестве объектов сравнения выбраны следующие видеостриминговые сервисы.

Twitch

Видеостриминговый сервис, специализирующийся на тематике компьютерных игр, в том числе трансляциях геймплея и киберспортивных турниров. Владельцем является американская компания Amazon.

Facebook Gaming

Мобильное приложение, специализирующееся на предоставлении пользователям возможности проведения прямых трансляций с мобильного телефона. Владельцем является американская компания Facebook, Inc.

YouTube Gaming

Ответвление основного видеохостинга YouTube, предоставляющая пользователям услуги хранения, доставки и показа видео, а также проведения прямых трансляций. Владельцем является американская компания Google.

WASD.TV

Видеостриминговый сервис, специализирующийся на тематике компьютерных игр, киберспортивных турниров Владельцем является российская компания МТС.

GoodGame.ru

Видеостриминговый сервис, специализирующийся на тематике киберспорта, а также компьютерных игр. Владельцем является российская компания ООО «Мидиан».

Задачи анализа

1. Определить критерии сравнения интернет-ресурсов.
2. Провести оценку интернет-ресурсов по критериям.
3. Выявить сильные и слабые стороны реализации интернет-ресурсов по обозначенным направлениям.
4. Сделать выводы по результатам сравнения интернет-ресурсов для того, чтобы найти наилучший вариант для создания интеграции вместе с игрой.

Критерии сравнения основных характеристик интернет-ресурсов

1. Наличие API для отправки запросов и получения результатов и документации к ним.
2. Наличие балльной системы поощрения зрителя за просмотр.
3. Наличие интерфейса для создания и проведения общих голосований.
4. Наличие чата на канале пользователя во время проведения прямых трансляций.

Условные обозначения:

- «+» – реализация решения удовлетворяет критерию;
- «+/–» – реализация решения не удовлетворяет критерию;
- «–» – отсутствие предмета оценки.

Результаты исследования основных характеристик интернет-ресурсов представлены в Таблица 2.

Таблица 2 – Основные характеристики интернет-ресурсов

Название интернет-ресурса	Twitch	Facebook Gaming	YouTube Gaming	WASD.TV	Good-Game
Наличие API и документации	+	–	+	–	+
Наличие балльной системы поощрения зрителя за просмотр	+	–	–	–	–
Наличие интерфейса для создания и проведения общих голосований	+	–	–	–	–
Наличие чата на канале пользователя	+	+	+	+	+

Анализ проведенного сравнения интернет-ресурсов

Проведенное сравнение интернет-ресурсов, предоставляющих услуги проведения прямых трансляций, позволяет сделать следующие выводы

1. У всех интернет-ресурсов присутствует чат на каналах пользователей во время проведения трансляций.

2. У трех из пяти площадок для проведения стримов присутствует API и поясняющая документация.

3. Лишь у одного интернет-сервиса присутствуют балльная система поощрения зрителя и интерфейс для создания и проведения общих голосований.

Для интеграции сторонних пользователей выбран интернет-ресурс Twitch, так как он имеет больше всех плюсов и дополнительные способы взаимодействия с аудиторией, что упростит разработку в разы.

1.3.2 Выбор игрового движка для разработки

Игровой движок представляет собой базовое программное обеспечение компьютерной игры, предоставляя разработчику инструменты для редактирования сцен, расположения объектов, написания скриптов на поддерживаемых языках программирования.

В качестве объектов сравнения, выбраны следующие игровые движки.

Unity

Межплатформенная среда разработки компьютерных игр, разработанная американской компанией Unity Technologies. Unity позволяет создавать приложения, работающие на более чем 25 различных платформах, включающих персональные компьютеры, игровые консоли, мобильные устройства, интернет-приложения и другие.

Unreal Engine

Игровой движок, разрабатываемый и поддерживаемый компанией Epic Games. Первой игрой на этом движке был шутер от первого лица Unreal, выпущенный в 1998 году. Хотя движок первоначально был предназначен для разработки шутеров от первого лица, его последующие версии успешно применялись в играх самых различных жанров, в том числе стелс-играх, файтингах и массовых многопользовательских ролевых онлайн-играх. В прошлом движок распространялся на условиях оплаты ежемесячной подписки; с 2015 года Unreal Engine бесплатен, но разработчики использующих его приложений обязаны перечислять 5% роялти от общемирового дохода с некоторыми условиями.

CryEngine

Игровой движок, созданный немецкой частной компанией Crytek в 2002 году. CryEngine используется во всех играх, разработанных самой Crytek, начиная с Far Cry и по настоящее время; разработчики последовательно дорабатывали движок, выпуская новые версии. Crytek также предлагает движок для лицензирования другим компаниям.

Задачи анализа

1. Определить критерии сравнения выбранных игровых движков.
2. Провести оценку игровых движков по данным критериям.
3. Выявить сильные и слабые стороны реализации игровых движков по обозначенным направлениям.
4. Сделать выводы по результатам сравнения игровых движков для того, чтобы найти наилучший вариант для создания компьютерной игры.

Критерии сравнения основных характеристик игровых движков

1. Бесплатное использование движка.
2. Поддержка большого количества платформ.
3. Наличие удобного интерфейса для создания игр.
4. Поддержка языка программирования C#.

Условные обозначения:

- «+» – реализация решения удовлетворяет критерию;
- «+/–» – реализация решения не удовлетворяет критерию;
- «–» – отсутствие предмета оценки.

Результаты исследования основных характеристик игровых движков представлены в Таблица 3.

Таблица 3 – Основные характеристики игровых движков

Название игрового движка	Unity	Unreal Engine	CryEngine
Бесплатное использование движка	+	+	+/–
Поддержка большого количества платформ	+	+	–
Наличие удобного интерфейса для создания игр	+/–	+/–	–
Поддержка языка программирования C#.	+	+/–	–

Анализ проведенного сравнения игровых движков

Проведенное сравнение игровых движков, позволяет сделать следующие выводы.

1. Почти все движки распространяются бесплатно, однако в дальнейшем могут требовать процент от заработанных средств.
2. У двух из трех движков есть поддержка большого количества платформ, на которые можно выпускать свои игры.
3. Лишь у Unity присутствует удобный интерфейс для взаимодействия с будущей игрой и её объектами.
4. Почти все игровые движки так или иначе поддерживают язык программирования C#, необходимый для работы библиотеки TwitchLib.

Рассмотрев все плюсы и минусы игровых движков, объективно выбран Unity, так как он имеет больше всех плюсов.

Выводы по первому разделу

Проведено сравнение существующих компьютерных игр в жанре Roguelite, интернет-сервисов, предоставляющих пользователям проводить прямые трансляции, а также игровых движков, используемых для разработки игр.

Выбран игровой движок для разработки компьютерной игры.

Выбран интернет-сервис, специализирующийся на проведении прямых трансляций, для интеграции с компьютерной игрой.

2 РАЗРАБОТКА КОМПЬЮТЕРНОЙ ИГРЫ В ЖАНРЕ ROGUELITE

2.1 Описание используемых технологий

При создании Unity-проекта с компьютерной игрой выбран предустановленный шаблон «2D Game» для создания 2D игр. Он в себе содержит настроенные параметры проекта для работы в двухмерной плоскости[5].

Для написания игровых скриптов, классов и объектов используется среда разработки Microsoft Visual Studio 2019[4].

Для реализации обращения к серверу Twitch используется мощная библиотека TwitchLib на C#. Она обладает таким преимуществом, как упрощение подключения к серверам Twitch. При использовании программистом данной библиотеки от него требуется лишь ввод необходимых данных и определение спектра сервисов, с которым он собирается работать[2].

Создание текстур игровых объектов, персонажей и локаций осуществляется в графическом редакторе Adobe Photoshop[3].

2.2 Структура компьютерной игры

Для создания диаграмм по компьютерной игре использовалось специальное приложение Ramus [8].

Контекстная диаграмма представлена на Рисунок 6.

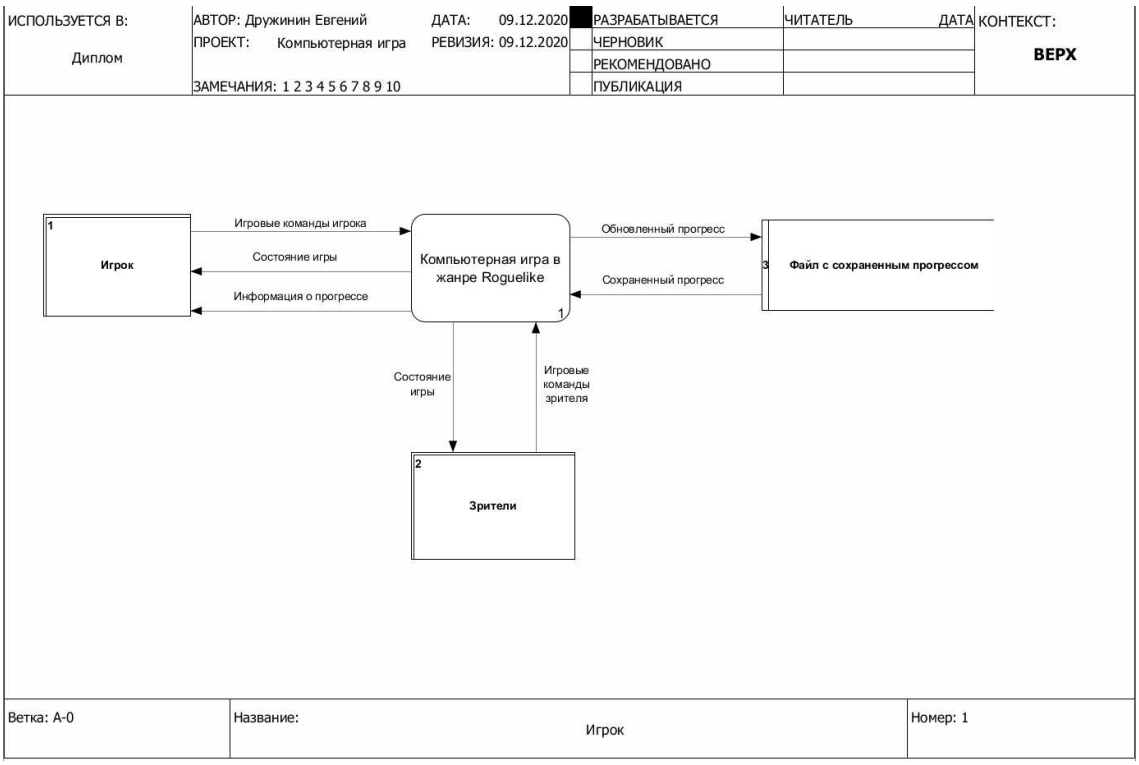

Рисунок 6 – Контекстная диаграмма компьютерной игры

Игрок взаимодействует с игрой посредством отправки игровых команд через устройства ввода: клавиатура и мышь.

Зрители влияют на ход игры с помощью игровых команд, которые посылаются с помощью интерфейса интернет-сервиса.

Файл с сохраненным прогрессом содержит в себе информацию о достижениях игрока по мере прохождения, а также информацию о расположении игрока относительно игрового мира в момент сохранения прогресса.

Стрелки контекстной диаграммы описаны в Таблица 4

Таблица 4 – Описание стрелок контекстной диаграммы

Название	Описание	Нотация
Игровые команды игрока	Передаются действия, выборы игрока	Нажатия клавиш управления
Состояние игры	Передается текущее положение игрока относительно игрового мира, его характеристики, количество очков здоровья	Координаты xy на локации, числа значений характеристик и очков здоровья

Окончание таблицы 2

Название	Описание	Нотация
Игровые команды зрителя	Передаются действия, выборы зрителя	Выбор команды на платформе Twitch
Обновленный прогресс	Сохранение прогресса в файле сохранения	Файл сохранения, хранящий статистику и текущий прогресс игрока
Сохраненный прогресс	Загрузка прогресса из файла сохранения	Файл сохранения, хранящий статистику и текущий прогресс игрока

Диаграмма декомпозиции компьютерной игры в жанре Roguelite представлена на Рисунок 7.

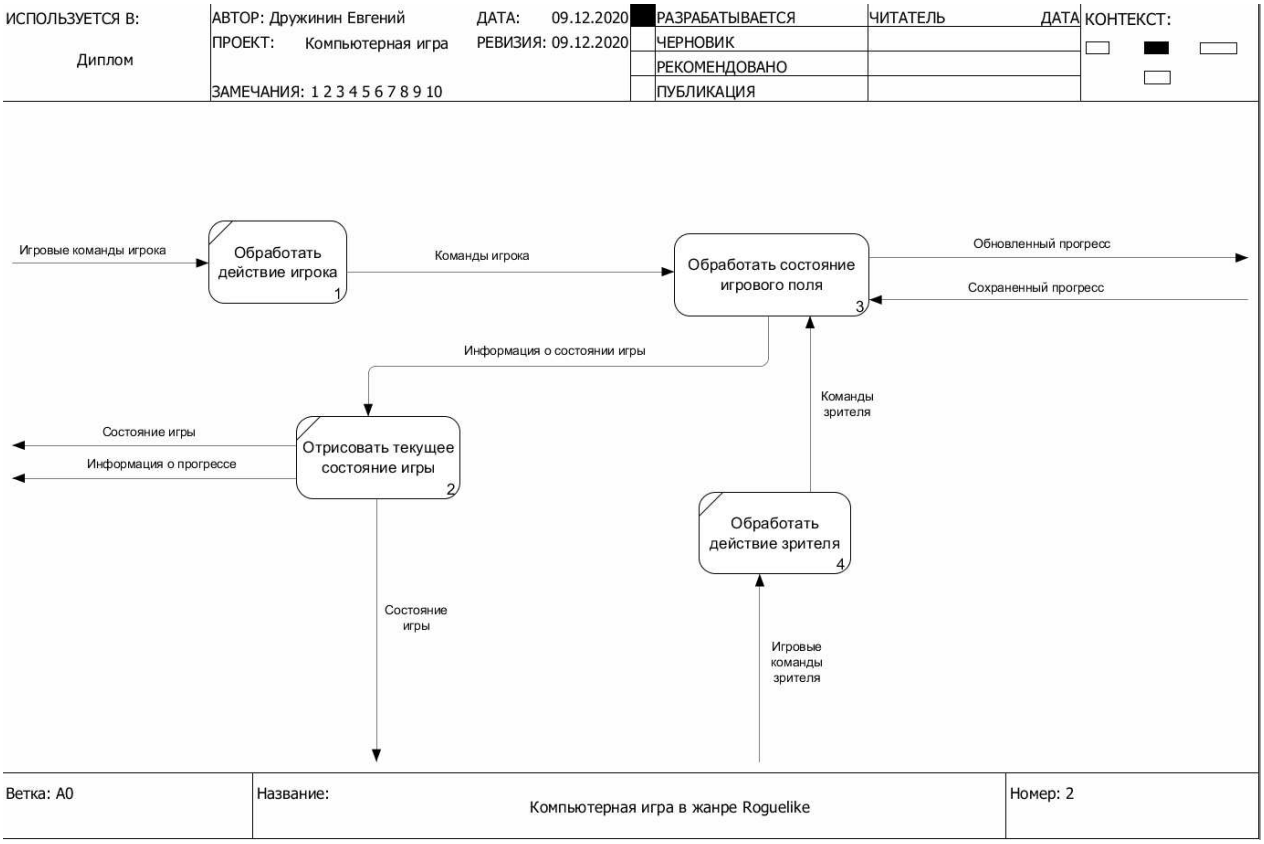

Рисунок 7 – Диаграмма декомпозиции компьютерной игры

Диаграмма предназначена для детализации функций и получилась при разбиении разбиении контекстной диаграммы на крупные подсистемы. Функциональные блоки диаграммы декомпозиции описаны в

Таблица 5.

Таблица 5 – Описание функциональных блоков диаграммы декомпозиции

Название	Описание	Нотация
Обработать действие игрока	Обработка нажатия клавиш, кликов мышью в игре и в меню	На входе: нажатие элемента управления На выходе: команда игрока внутри игры
Обработать состояние игрового поля	Применение команд игрока и команд зрителя на игровое поле	На входе: команды игрока, команды зрителя На выходе: состояние игры
Отрисовать текущее состояние игры	Отображение визуальной информации о текущем состоянии игры	На входе: информация о состоянии игры На выходе: визуальное состояние игры
Обработать действие зрителя	Обработка команд зрителя с платформы Twitch	На входе: нажатие элемента управления на платформе Twitch На выходе: команда зрителя внутри игры

2.3 Сюжет компьютерной игры

Классическая схема разработки игр в наши дни отталкивается от сюжета. Большинство разработчиков придумывают сюжет, по ходу которого игроку предстоит решить довольно несложные задания. Похоже на роман, в котором, чтобы перейти к новой главе, нужно решить кроссворд. Как результат, игрок зачастую испытывает мощные эмоции от сюжета, не слишком напрягаясь умственно и физически. Но это решение разработчиков не несёт в себе обучающего начала, которое придавало бы ценность игровой системе[11].

Главный герой безымянный получеловек полумашина в стимпанковском Лондоне 1800-ых годов (прим. Стимпáнк или паропáнк, – направление научной фантастики, включающее технологию и декоративно-прикладное искусство, вдохновлённое паровой энергией конца XIX века.).

Задача игрока выбраться из глубин завода, где его создали, и добраться до главы корпорации, которая его лишила персонажа памяти и преобразила его тело.

Зрители игры выступают в двух ролях.

Спонсоры корпорации

Их основная задача – помешать игроку выбраться с завода, где держат главного героя. Свой вклад спонсоры осуществляют посредством вложений виртуальной валюты (баллов канала на Twitch) в мешающие прохождению элементы игры такие, как появление дополнительных врагов в комнатах, уменьшению шанса появления полезных предметов.

Конкуренты корпорации

Их основная задача – помочь игроку выбраться с завода и поспособствовать ослаблению позиции корпорации на рынке. Свой вклад конкуреты осуществляют посредством вложений виртуальной валюты в помогающие прохождению элементы игры такие, как уменьшению количества врагов в комнатах, увеличению шанса появления полезных предметов.

2.4 Разработка компьютерной игры

Процесс разработки сопровождается созданием и определением очередности игровых сцен: MainMenu и Level.

2.4.1 Сцена «MainMenu»

Игровая сцена, которая отображается при запуске игры и предлагает пользователю выбор из трех пунктов: «Начать игру», «Настройки» и «Выход» Сцена «MainMenu» представлена на Рисунок 8.

Рисунок 8 – Главное меню

При нажатии на кнопку «Начать игру» происходит плавный переход с затемнением на сцену «Level».

При нажатии на кнопку «Настройки» происходит переход в меню настроек игры. Меню настроек игры представлено на Рисунок 9.

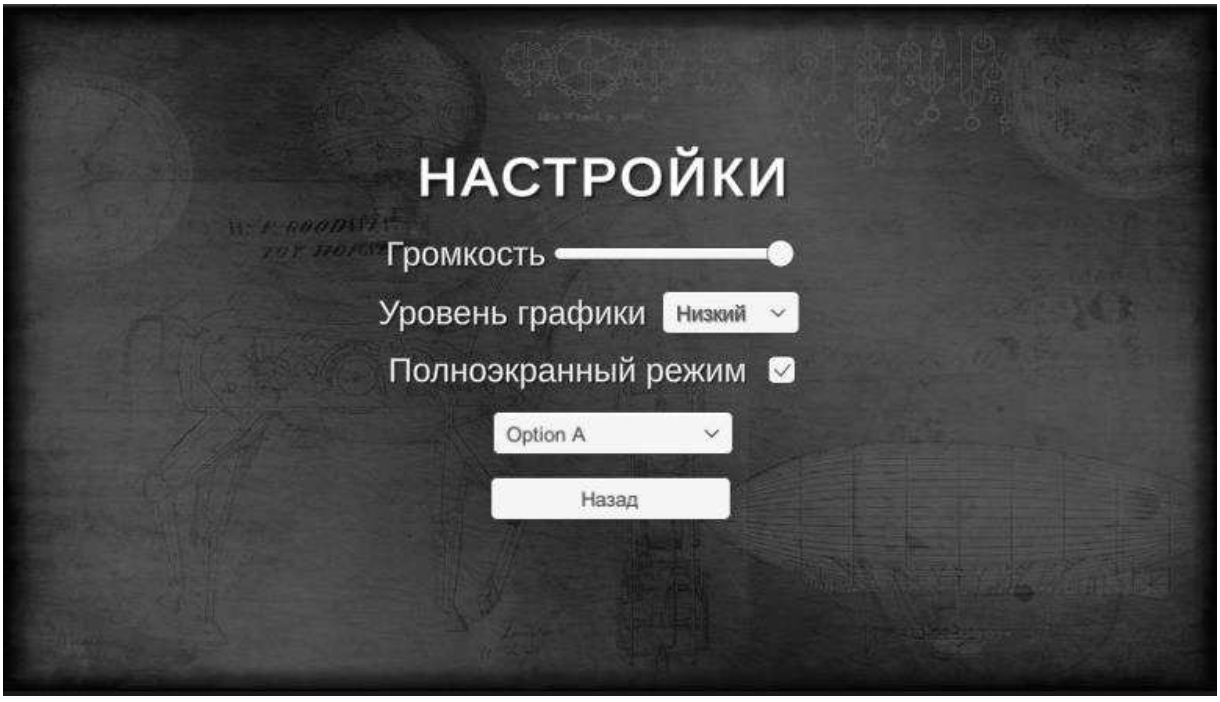

Рисунок 9 – Меню настроек игры

Пользователю даётся возможность выбрать предпочитаемый уровень, графики, громкости звуков, разрешения экрана и переключение режима экрана.

2.4.2 Сцена «Level»

Игровая сцена, в которой происходит основной процесс игры. В ней определены следующие объекты и компоненты:

- MainCamera – основная камера игры, которая осуществляет захват и отображение игрового мира игроку;
- MapGenerator – объект, который хранит в себе скрипты генерации локаций;
- HUD – часть визуального интерфейса, отображающаяся поверх основного игрового пространства;
- Player – объект, который контролирует сам игрок;
- Twitch – объект, который хранит в себе скрипты для подключения к серверам Twitch;
- GameState – объект, хранящий в себе скрипты основных состояний игры, такие как пауза, конец игры.

Для создания и конструирования уровней созданы «префабы» – особые типы ресурсов, позволяющие хранить в себе весь игровой объект со всеми компонентами и значениями свойств. Префаб выступает в роли шаблона для создания экземпляров хранимого объекта в сцене. В список префабов входят все возможные варианты комнат, отличающихся друг от друга расположением выходов из них, а также объекты компьютерных противников. Пример префаба комнаты с 4-мя выходами приведен на Рисунок 10.

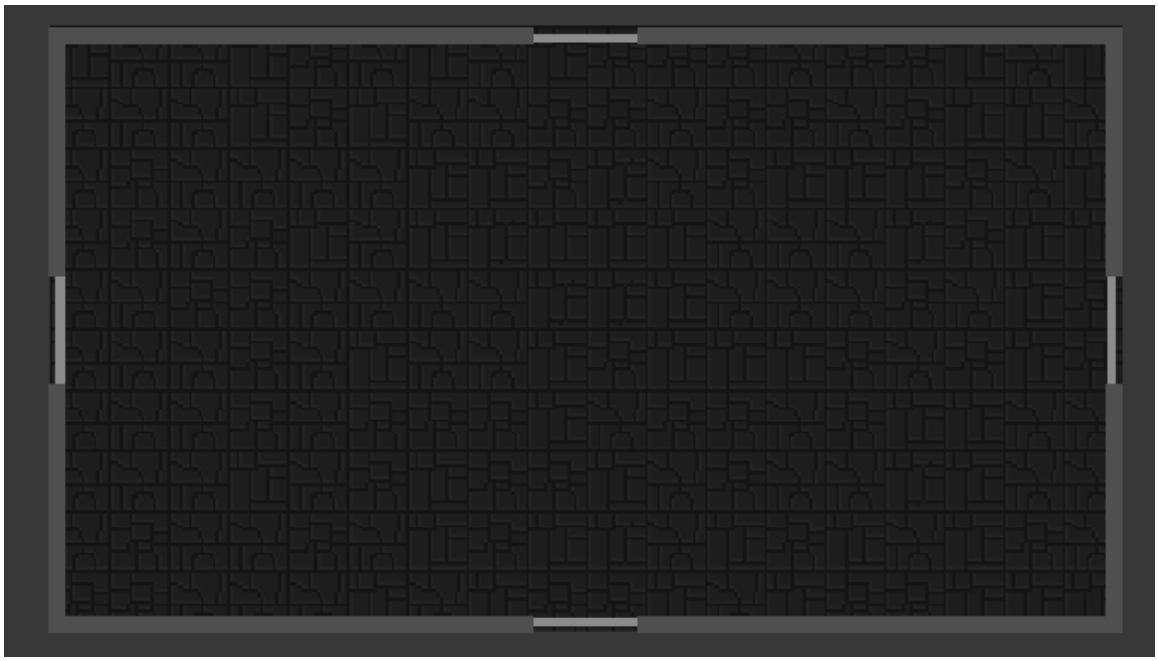

Рисунок 10 – Префаб комнаты

2.4.3 Разработка генератора уровней

При переходе на сцену «Level», прежде чем её отобразить, объект «MapGenerator» осуществляет генерацию всего уровня. Пример случайно сгенерированной локации представлен на Рисунок 11.

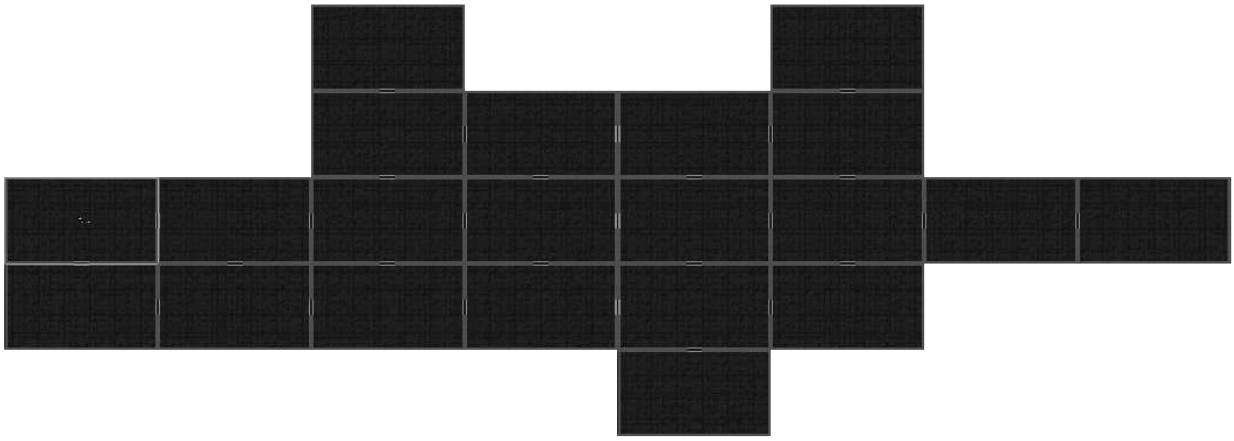

Рисунок 11 – Обзор всей локации

Генератор уровней построен так, что он сначала случайным образом генерирует само число комнат, далее создается первоначальная локация, откуда игрок начинает своё похождение. Стартовая локация представлена на Рисунок 12.

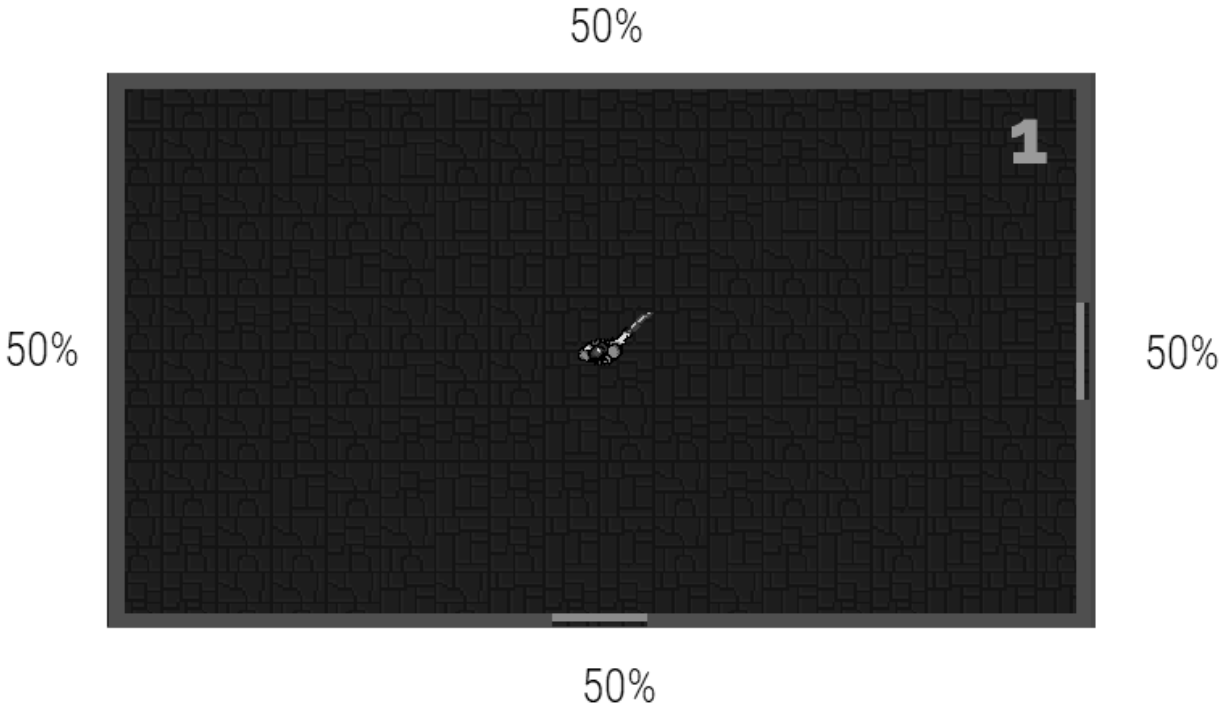

Рисунок 12 – Генерация уровней

Для каждого направления верх, низ, право, лево с шансом 50% создается ещё одна соседняя комната. И так для каждого помещения происходит генерация до тех пор, пока не подойдет к концу число необходимых локаций. Генерация игровых локаций представлена в Листинг А.1.

Генерация соседних комнат представлена на Рисунок 13.

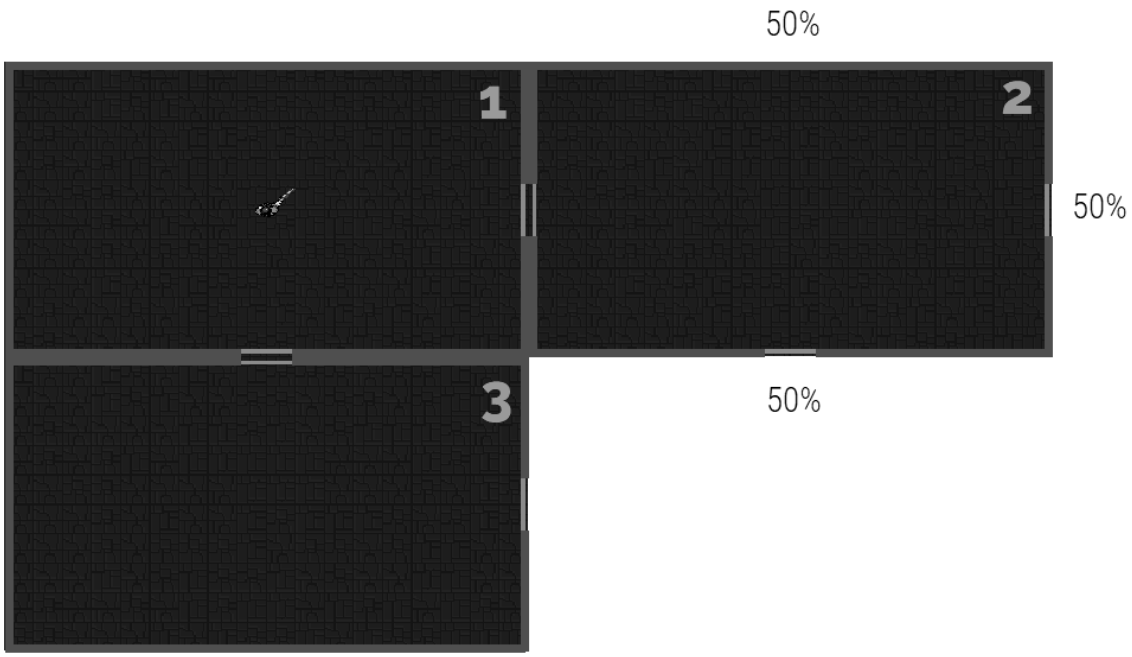

Рисунок 13 – Генерация соседних комнат

Когда схема карты создана, происходит подбор подходящих макетов комнат с дверьми в нужных направлениях. Создается герметичная локация, дабы игрок не смог выйти за её пределы. Генерация объектов комнат представлена в Листинг А.2.

Далее начинается процесс генерации объектов, препятствий в комнатах Пример возможных объектов в комнате представлен на Рисунок 14.

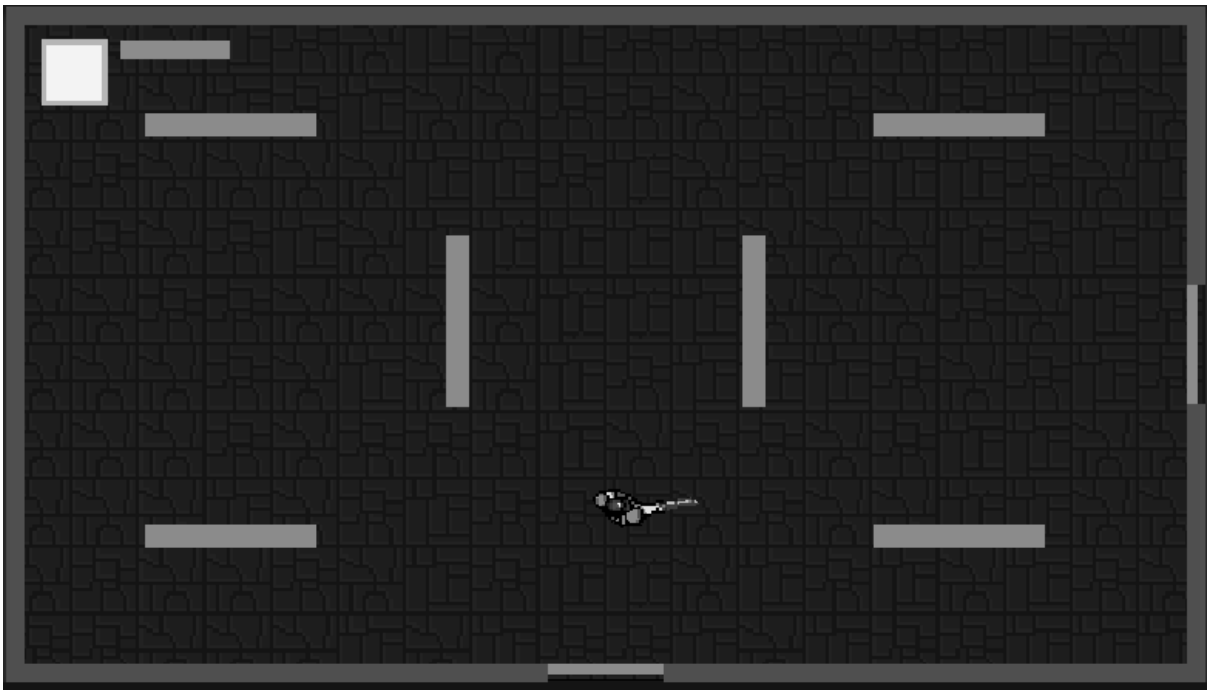

Рисунок 14 – Препятствия в комнате

Препятствия и объекты в комнатах создают более разнообразные игровые локации и усиливают динамику прохождения игры.

2.4.4 Разработка компьютерных противников

Почти что в каждой комнате игрока будут ждать управляемые искусственным интеллектом враги, задача которых убить игрока.

Для каждой комнаты на уровне есть свой параметр сложности, который тоже генерируется случайным образом. От этого параметра зависят количество врагов, их сила атаки, количество здоровья, скорость перемещения и поворота головы в сторону игрока. Пример расположения врагов можно увидеть на Рисунок 15.

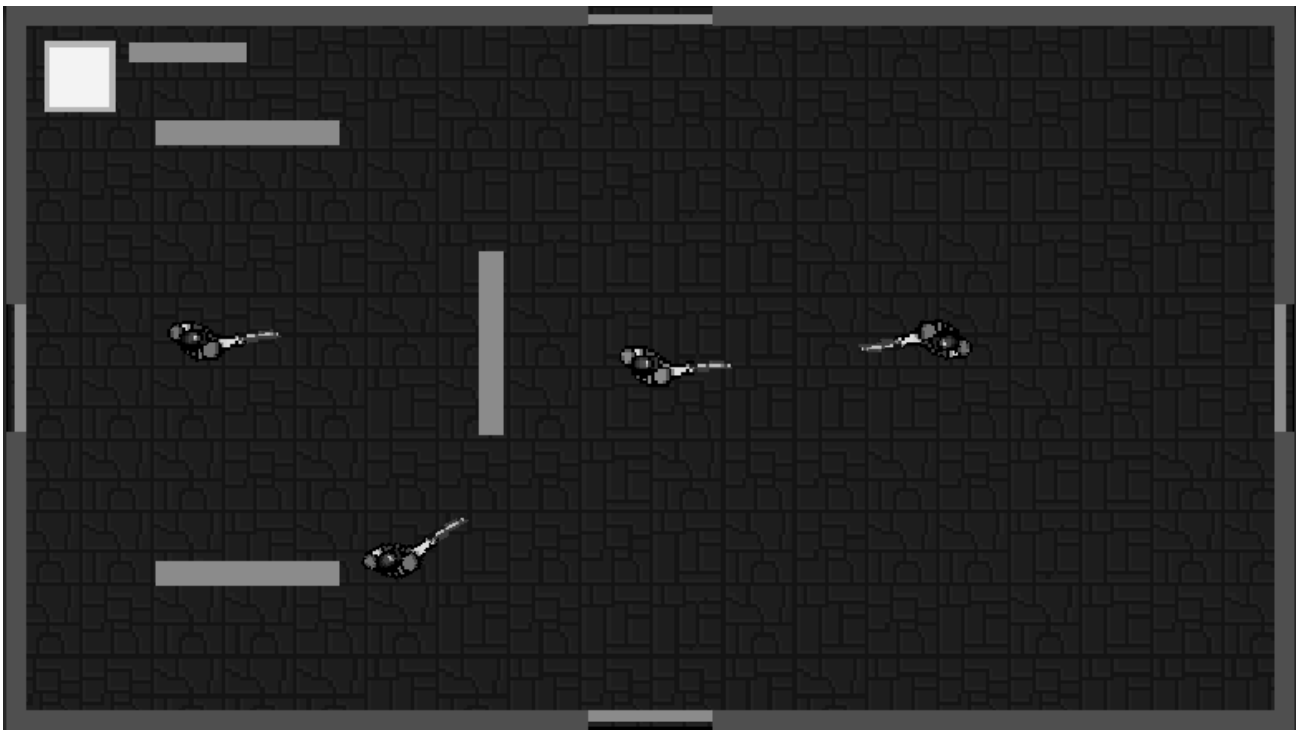

Рисунок 15 – Враги и игрок в одной комнате

У каждого врага есть свои индивидуальные параметры, им присуще иметь у себя в инвентаре определенное оружие, со своим боезапасом, который может иссякнуть, поэтому противник будет вынужден либо найти оружие на уровне, либо атаковать игрока в рукопашном бою.

Ориентация компьютерного врага на местности осуществляется с помощью компонента «A* Pathfinder». Он разбивает всю игровую локацию на равные по размерам клетки, в которых проверяется, есть ли объект, через который нельзя пройти. Все эти клетки участвуют в процессе построения маршрута компьютерного врага, формируя кратчайший до игрока путь, минуя все препятствия. По мере движения игрока путь до него обновляется, тем самым обеспечивая постоянное движение противника в сторону пользователя.

Как только в зоне видимости компьютерного врага появляется игрок, искусственный интеллект начинает в него стрелять. Пули создаются в месте, где у персонажа находится конец оружия, и летят строго в направлении вектора, куда смотрел враг. При соприкосновении с препятствиями или игроком пуля исчезает.

У каждого оружия есть свои характеристики, такие как урон от пули, количество патронов в магазине, общий боезапас, время перезарядки и процент точности. При генерации уровней, есть шанс, что в одной из комнат будет лежать оружие, которое игрок сможет подобрать и использовать в дальнейшем своём прохождении.

2.4.5 Разработка игровых предметов

Игровые предметы представляют собой объекты, которые с небольшим шансом появляются в комнатах вместе с врагами. Игровые предметы можно разделить на два вида.

Усилители

Усилители представляют собой предметы, которые влияют на характеристики игрока: максимальный уровень здоровья, скорость передвижения, наносимый урон. Пример усилителя максимального уровня здоровья представлен на Рисунок 16.

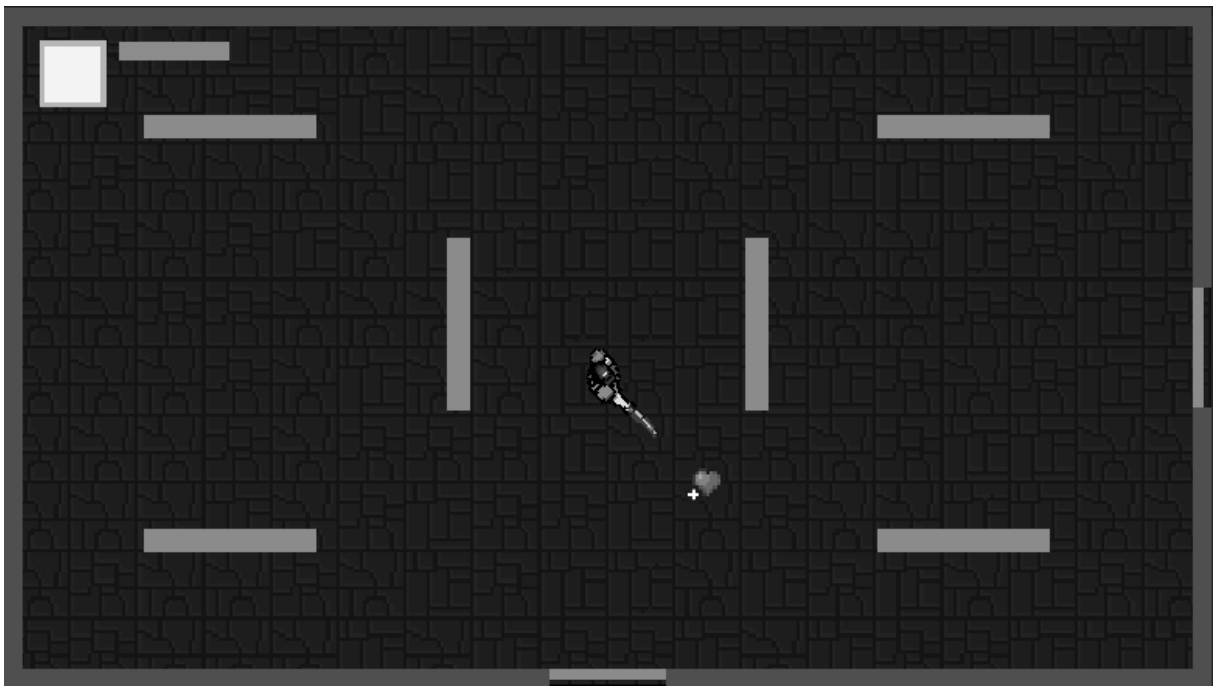

Рисунок 16 – Усилитель здоровья на локации

При приближении игрока к усилителю, он его подберет, а сам игровой объект исчезнет.

Оружия

Помимо усилителей игрока, на игровых уровнях с некоторым шансом появляются оружия. Пример оружия на локации представлен на Рисунок 17.

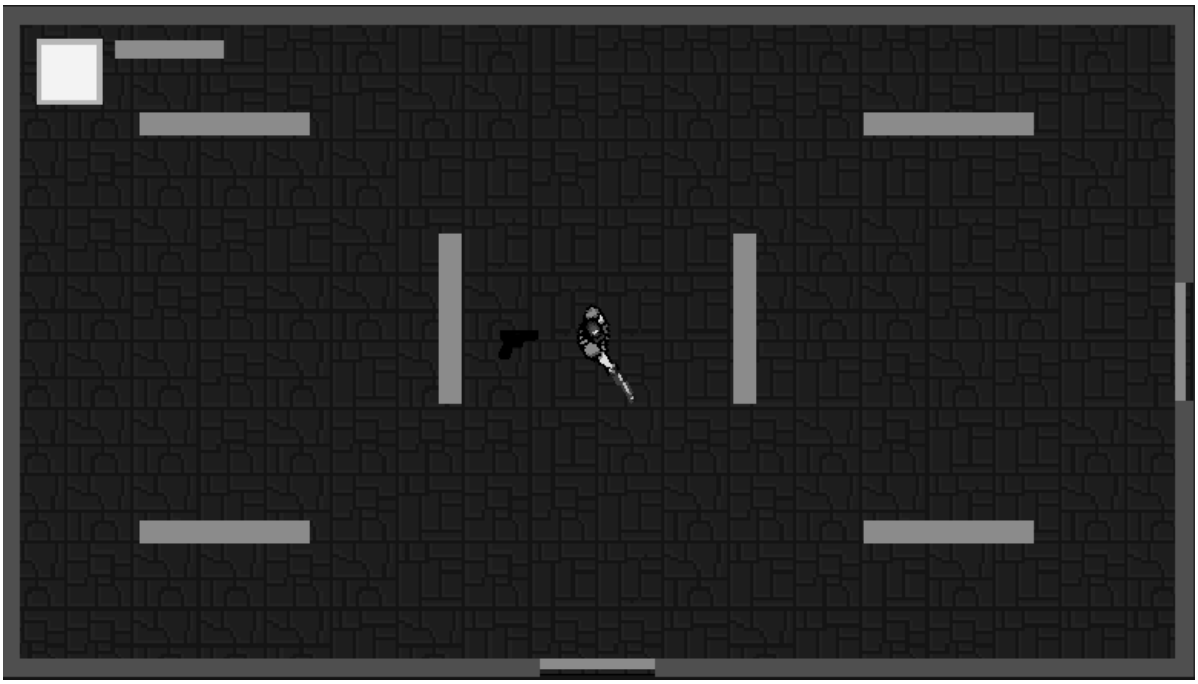

Рисунок 17 – Пистолет на локации

Каждое оружие имеет свои индивидуальные характеристики: урон наносимый одной пулей, время перезарядки, точность стрельбы, скорострельность, размер магазина и общий боезапас.

2.4.6 Интеграция сторонних пользователей в игровой процесс

Перед загрузкой игровой сцены «Level» и генерацией уровня происходит создание WebSocket соединения с сервером Twitch. Осуществляется это при помощи методов C# библиотеки TwitchLib. Чтобы осуществить подключение, игроку нужно предоставить свой логин от аккаунта Twitch и OAUTH-ключ, который нужно сгенерировать заранее на специальном сайте.

Как только процесс подключения к серверу и генерация уровня закончатся, у сторонних пользователей, которые находятся на стриминговом канале игрока, появится специальный интерфейс для взаимодействия с игрой, пример которого представлен на Рисунок 18.

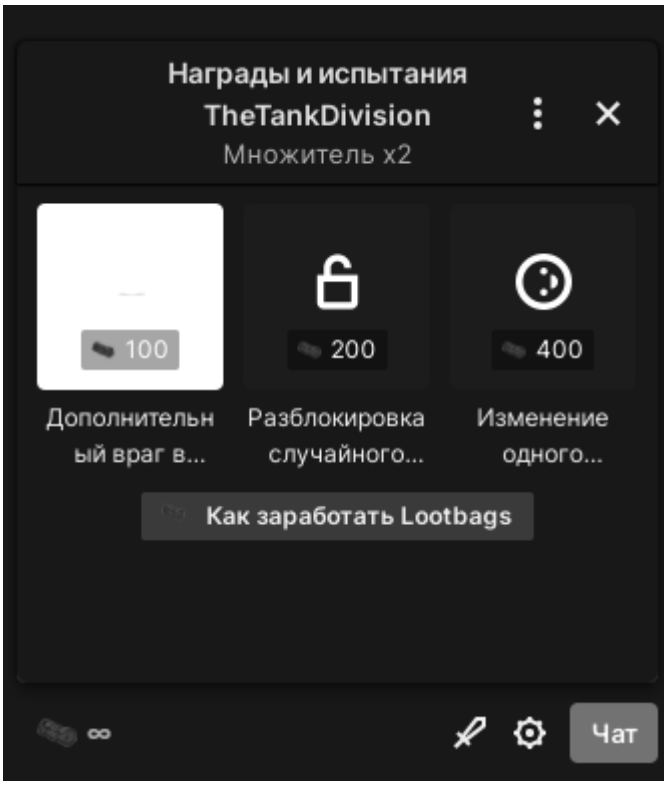

Рисунок 18 – Интерфейс взаимодействия с игрой

Взаимодействие происходит с помощью системы наград за «баллы канала». Зрители за определенное время просмотра прямой трансляции игрока могут заработать баллы, которые могут потратить на «награды», которые осуществляют взаимодействие с игрой пользователя. Пример награды представлен на Рисунок 19.

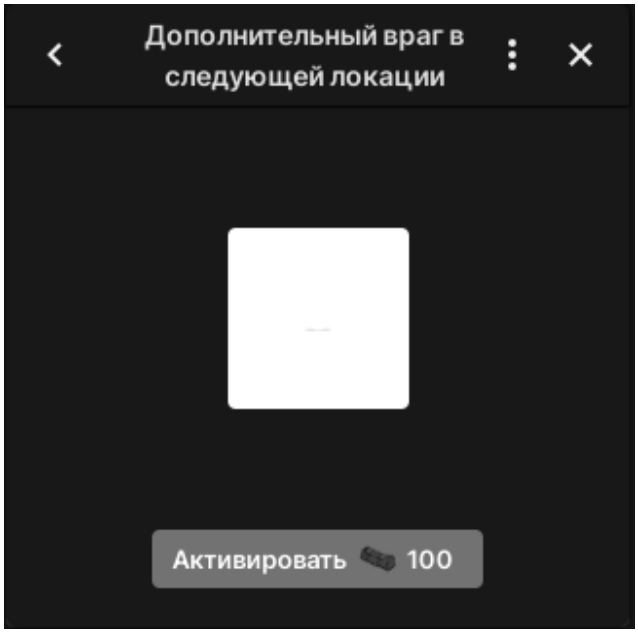

Рисунок 19 – Пример награды для взаимодействия с игрой

После выхода пользователя из игры, награды, взаимодействующие с игрой, исчезают из списка доступных для активации.

Выводы по второму разделу

Созданы игровые сцены, отображающие состояние игры.

Созданы игровые объекты, шаблоны локаций и врагов.

Реализован алгоритм генерации игровых уровней.

Разработан искусственный интеллект врагов.

Реализован процесс взаимодействия сторонних пользователей с игрой.

Разработана компьютерная игра в жанре Roguelite

3 РАСЧЕТ ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ИГРЫ

Разработка компьютерной игры сопряжена с капитальным вложением, как на приобретение техники и программного обеспечения, так и на разработку проектов, выполнение подготовительных работ и т.д. Поэтому необходимо экономическое обоснование целесообразности ведения разработок.

Хорошо составленный бизнес-план является частью успеха хорошего распространения и продаж разрабатываемой компьютерной игры. Он содержит данные необходимые для эффективного внедрения на рынок и успешного варьирования в зависимости от состояния рынка, а также содержит ряд показателей дающих представления об экономической и коммерческой эффективности данной работы.

Определение затрат на разработку продукта

В данном разделе определяются необходимые инвестиции для разработки и реализации программного продукта.

Расчет единовременных затрат приведён в Таблица 6.

Таблица 6 – Расчет единовременных затрат

	Наименование технического средства и ПО	Тип или модель	Стоимость
	Операционная система	Microsoft Windows 10 Home Edition	5020
	Системный блок	IRU Home 315	51990
	Монитор	Asus VA24EHE	9320
	Клавиатура и мышь	Logitech Wireless Combo MK270	2130
	Стол компьютерный		5000
	Стул		4000
Итого			77460

Подсчет единовременные затраты:

$$Зез = 5020 + 51990 + 9320 + 2130 + 5000 + 4000 = 77460 \text{ руб.}$$

Затраты на создание программного продукта складываются из расходов по оплате труда разработчика и расходов по оплате машинного времени при отладке кода. Расчет текущих затрат приведён в Таблица 7.

Таблица 7 – Календарный план-график основных этапов выполнения дипломной работы

Наименование этапа	Дата начала	Длительность, недель
1. Разработка концепции игры: Определение главных героев написание истории, создание концепции персонажей, создание концепции окружения и локаций, создание концепции интерфейса	12.01.2021	3
2. Разработка первоначального прототипа игры: создание первых локаций, создание модели игрового персонажа, реализация основных игровых механик, создание первых игровых предметов	02.02.2021	3
3. Доработка прототипа, отрисовка дизайна и текстур локаций, доработка модели персонажей, создание текстур, дополнение игровых механик	23.02.2021	4
4. Консультации с руководителем проекта, поиск и устранение визуальных и технических багов	23.03.2021	1
5. Отчёт в электронном виде	30.03.2021-13.04.2021	2

Затраты на разработку концепции игры рассчитаны по формуле (1).

$$Зк = Ср * Тк, \quad (1)$$

где Ср = 300 руб./час – тарифная ставка программиста уровня Junior;

Тк – время, затраченное на работу с литературой и теоретический анализ;

Зк = 300 * 120 = 36000 руб.

Затраты на разработку первоначального прототипа игры рассчитаны по формуле (2).

$$Зп = Ср * Тп, \quad (2)$$

где Тп – время, затраченное на разработку первоначального прототипа игры.

Зп= 300 * 120 = 36000 руб.

Затраты на доработку прототипа рассчитаны по формуле (3).

$$Здп = Ср* Тдп, \quad (3)$$

где Тдп – время, затраченное на доработку прототипа.

Здп = 300 * 160 = 48000 руб.

Затраты на консультации с руководителем проекта рассчитаны по формуле (4).

$$Зкр = Срп * Ткр, \quad (4)$$

где Ткр – время, затраченное на консультации с руководителем проекта;

Срп =400 руб./час – тарифная ставка руководителя проекта;

Зкр = 400 Ч 45 = 18000 руб.

Полные затраты рассчитаны по формуле (5).

$$Зсум = Зк + Зп + Здп + Зкр, \quad (5)$$

Зсум = 36000 + 36000+ 48000 + 18000 = 138000 руб.

Результаты расчетов представлены в Таблица 8.

Таблица 8 – Составляющие затрат при проектировании

Наименование затрат	Обозначение	Сумма	
		в рублях	в %
1. Разработка концепции игры	Зк	36000	26
2. Разработка первоначального прототипа игры	Зп	36000	26
3. Доработка прототипа	Здп	48000	35
4. Консультация с руководителем проетка	Зкр	18000	13
ИТОГО	Зсум	138000	100

На основании значений, полученных по данным формулам, составляется календарный план-график работы над проектом.

Оплата за пользование электричеством составляет

Зэ=0,3кВатт * 585 ч * 2,7 руб= 473,85 руб.

Текущие затраты (себестоимость) (С) включают затраты на постановку задачи, разработку алгоритмов и программ, а также затраты, связанные с содержанием и эксплуатацией ВТ. Расчет производится по формуле (6).

$$С = Зпр + Нз + Зэ, \quad (6)$$

где Зпр – затраты на заработную плату программиста;

Зпр = 25019,2 руб.

Текущие затраты представлены в Таблица 9.

Таблица 9 – Текущие затраты

Наименование статей затрат	Сумма
1. Затраты на заработную плату (Зпр)	25019,2
2. Отчисления в фонды с заработной платы труда (ФФОМС, ПФР, ФСС) (Зф)	7505,76
3. Расходы на электричество при пользовании вычислительной техники	473,85
Итого затрат (З)	32998,81

Сведем расчет затрат в общую Таблица 10.

Таблица 10 – Расчет затрат на разработку программного средства

Затраты и ожидаемые доходы от внедрения ИС	Период			
Затраты	1 квартал	2 квартал	3 квартал	4 квартал
1. Капитальные затраты	77460			
2. Текущие затраты:	32998,81	32998,81	32998,81	32998,81
Заработная плата	25019,2	25019,2	25019,2	25019,2
Отчисления в фонды (ФФОМС, ПФР, ФСС)	7505,76	7505,76	7505,76	75567505,76
Расходы на электричество при пользовании вычислительной техникой	534,24	534,24	534,24	534,24
Итого затрат:	110458,81	32998,81	32998,81	32998,81

Отчисления от фонда оплаты труда (Зф) (30 % от Зпр);

Зф = 30%*Зпр = 7505,76 руб.

Определение доходов от внедрения в работу программного обеспечения

Таблица 11 – Доходы

Доходы/Месяцы	1 квартал	2 квартал	3 квартал	4 квартал
1. Привлечение новых игроков.	30000	50000	70000	90000
Итого доходов тыс. руб.:	30000	50000	70000	90000

Показатели эффективности

Расчет экономической эффективности проводится по следующим показателям:

а) чистый дисконтированный доход (ЧДД) или интегральный эффект;

б) индекс доходности (ИД);

в) внутренняя ставка доходности (ВСД);

г) срок окупаемости;

Чистый дисконтированный доход – превышение интегральных результатов над интегральными затратами. Определяется как сумма текущих эффектов за весь расчетный период, приведенная к начальному шагу.

Метод дисконтированных денежных потоков предполагает анализ потоков инвестиционного капитала, причем как затратных, так и прибыльных, с учетом их временно-стоимостной оценки.

Выбор данного метода обусловлен следующими причинами:

- необходимость инфляционной корректировки показателей;
- обеспечение наглядности и простоты расчета срока окупаемости;
- стабильность и предсказуемость денежных потоков;
- равномерность денежных потоков.

Этот метод нашел широкое применение в зарубежной практике в ходе оценки инвестиций в материальные и нематериальные активы, особенно в случаях нестабильной макроэкономической ситуации (инфляция, возможность кризисных ситуаций и т. д.).

Суть этого метода заключается в отнесении предполагаемых денежных потоков (и положительных, и отрицательных) к временным интервалам, начиная с

момента начала инвестирования. Как правило, при малых показателях рентабельности продукта сроком анализа этого метода выбирают нормативный «срок жизни» продукта, или срок окупаемости продукта-субститута. В данном случае для установления срока окупаемости достаточно ограничиться сроком в 3 года.

К недостаткам данного метода можно отнести то, что он не позволяет учесть случайные риски (как внешние, так и внутренние), а также известную условность расчета срока окупаемости, связанную с выбором ставки дисконтирования.

Чистый дисконтированный доход определяется как сумма текущих эффектов за весь расчетный период. Он рассчитывается по формуле (7).

$$ЧДД = \sum_{t=1}^{5} (Д_t - P_t) \cdot \frac{1}{(1+\alpha)^t} \quad (7)$$

где Дt – результаты, достигаемые на t-ом шаге расчета;

Pt – затраты, осуществляемые на том же шаге;

Дt - Pt – эффект достигаемый на t-ом шаге.

$1/(1+ a)^t$ – коэффициент приведения по времени результатов и затрат;

а – норма дисконта, равная приемлемой для инвестора норме дохода на капитал -0,14.

Предполагаемые доходы от продаж (Д) определяются по формуле (8).

$$Д = Цпр*N, \quad (8)$$

где Цпр – цена продажи ПП, руб;

N – объем продаж по периодам в соответствии с исследованиями рынка, шт.

Расходы (Р) включают, кроме текущих затрат, расходы на тиражирование и рекламу и рассчитываются по формуле (9).

$$Р = Зтир*N + Зр*n \text{ мес}, \quad (9)$$

где n – количество месяцев в рассматриваемом периоде.

Результаты расчетов по технико-экономическим показателям работы приведены в Таблица 12.

Таблица 12 – Технико-экономические показатели работы

Периоды (месяц)	Показатели		$1/(1+a)^t$	Дисконтированные	Годовая экономическая эффективность	ЧДД с нарастающим итогом	
	Доходы	Расходы					
	1	2	3	4=1*3	5=2*3	6=4-5	7
1 квартал 2021	30000	110458,8	0.88	26315,8	96893,7	-70578	-70578
2 квартал 2021	50000	32998,8	0.77	38473,4	25391,5	13081,9	-57496
3 квартал 2021	70000	32998,8	0.67	47248	22273,3	24974,7	-32521
4 квартал 2021	90000	32998,8	0.59	53287,2	19537,9	33749,3	1227,99
1 квартал 2022	110000	32998,8	0.46	50114,5	15033,8	35080,7	36308,7
2 квартал 2022	130000	32998,8	0.52	67517,9	17138,5	50379,4	86688,1
Итого:	480000	275452,9	3,89	282956,8	196268,8		

Индекс доходности – представляет собой отношение суммы приведенных эффектов к величине капитальных вложений и рассчитывается по формуле (10).

$$ИД = \frac{\sum_{t=1}^{T} Д_t * \frac{1}{(1+\alpha)^t}}{\sum_{t=1}^{T} P_t * \frac{1}{(1+\alpha)^t}} \quad (10)$$

Индекс доходности строится из тех же элементов, что и ЧДД. Если ЧДД положителен, то ИД > 1 и наоборот. Индекс доходности: 1,44

Расчет ЧДД инвестиционного проекта показывает, является ли он эффективным при некоторой заданной норме дисконта.

Срок окупаемости – минимальный временной интервал (от начала осуществления проекта), за пределами которого интегральный эффект становится положительным и в дальнейшем остается неотрицательным. Это период (измеряемый в месяцах, кварталах или годах), начиная с которого первоначальные вложения и другие затраты, связанные с проектом, покрывается суммарными результатами его осуществления.

Определим срок окупаемости графически. Для этого необходимо построить график срока окупаемости проекта. График окупаемости представлен на Рисунок 20.

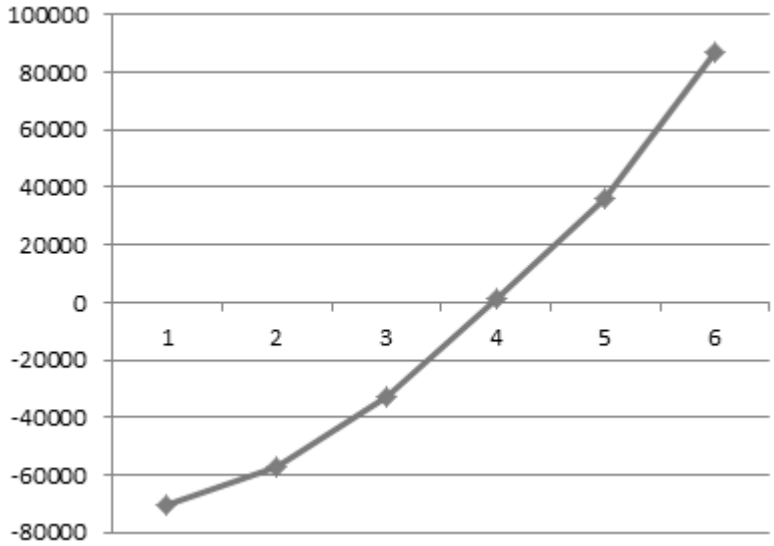

Рисунок 20 – График окупаемости

Выводы по третьему разделу

Проведён анализ основных разделов бизнес-плана.

Осуществлена калькуляция темы с оценкой экономической эффективности реализации программного продукта.

ЗАКЛЮЧЕНИЕ

Выполнен сравнительный анализ игр-аналогов. С учетом полученной в ходе анализа информации, принято решение в пользу разработки компьютерной игры, сочетающей в себе все ключевые факторы жанра Roguelite .

Рассмотрены существующие интернет-сервисы, предоставляющие возможность проведения прямых трансляций и специализирующиеся на тематике компьютерных игр.

Рассмотрены игровые движки, предоставляющие нужный инструментарий для разработки компьютерной игры.

Разработана компьютерная игра в жанре Roguelite с элементами интеграции сторонних пользователей интернет сервиса Twitch.

С помощью библиотеки TwitchLib на языке программирования C# реализовано клиент-серверное взаимодействие компьютерной игры и интернет-сервиса Twitch, созданы обработчики событий активации команд пользователями.

С помощью игрового движка Unity и языка программирования C# реализована компьютерная игра

Таким образом, цель дипломной работы достигнута. Результат работы может быть применен на практике в любом направлении, где потребуется реализация интеграции сторонних пользователей интернет-сервисов.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1 C# Programming Guide. [Электронный ресурс] URL: https://msdn.microsoft.com/en-us/library/67ef8sbd.aspx

2 TwitchLib Documnetation [Электронный ресурс] URL: https://github.com/TwitchLib/TwitchLib

3 Adobe Photoshop User Guide [Электронный ресурс] URL: https://helpx.adobe.com/ru/photoshop/user-guide.html

4 Официальный сайт Visual Studio. [Электронный ресурс] URL: https://www.visualstudio.com/ru/vs/

5 Unity3D Manual. [Электронный ресурс] URL: http://docs.unity3d.com/Manual/index.html

6 Вигенс К., Битти Д. Разработка требований к программному обеспечению. – М.: Русская Редакция, 2014. – 736 с.

7 Гамма Э., Хелм Р., Джонсон Р., Влиссидс Д. Приемы объектноориентированного программирования. Паттерны проектирования. – СПб.: Питер, 1994. – 395 с.

8 Грекул В. И. , Денищенко Г. Н., Коровкина Н. Л. Проектирование информационных систем: курс лекций – Москва: Интернет-Университет Информационных Технологий (ИНТУИТ), 2005 – 304 с.

9 Майерс Г., Баджет Т. Искусство тестирования программ. – М.: Вильямс, 2012. – 272 с.

10 Официальный сайт Unity3D. [Электронный ресурс] URL: https://unity3d.com/ru

11 Рэф Костер: Разработка игр и теория развлечений – ДМК, 2018 – 288 с.

12 Рамбо Дж. UML 2.0. Объектно-ориентированное моделирование разработка. – СПб.: Питер, 2007. – 544 с.

ПРИЛОЖЕНИЕ А

Исходные коды компьютерной игры

Листинг А.1 – Генерация уровня.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Map
{
    private List<Room> rooms;
    int roomsCount;
    System.Random rnd;
    int minCount;
    int maxCount;
    int chance;
    public Map(System.Random _rnd, int _roomsCount, int _chance)
    {
        rnd = _rnd;
        roomsCount = _roomsCount;
        rooms = new List<Room>();
        chance = _chance;
    }

    public Hashtable GenerateRooms()
    {
        rooms.Clear();
        Hashtable hashtable = new Hashtable();
        rooms.Add(new Room(0, 0));
        hashtable.Add("0,0", new Room(0, 0));
        Room[] directions = new Room[4];
        directions[0] = new Room(1, 0);
        directions[1] = new Room(0, 1);
        directions[2] = new Room(-1, 0);
        directions[3] = new Room(0, -1);
        int count = roomsCount-1;
        int i = 1;
        while (count>0)
        {

            bool isNotCreated = true;

            for (int j = 0; j < 4; j++)
            {
                int percent = rnd.Next(0, 100);

                int x = rooms[i - 1].X + directions[j].X;
                int y = rooms[i - 1].Y + directions[j].Y;
```

```
                string key = $"{x},{y}";
                if (percent >= chance && !hashta-
ble.ContainsKey(key))
                {
                    Room tempRoom = new Room(x, y);
                    hashtable.Add(key, tempRoom);
                    rooms.Add(tempRoom);
                    count--;
                    isNotCreated = false;
                }

                if (count <= 0)
                    break;
            }
            if (isNotCreated)
            {
                for (int j = 0; j < 4; j++)
                {
                    int x = rooms[i - 1].X + directions[j].X;
                    int y = rooms[i - 1].Y + directions[j].Y;
                    string key = $"{x},{y}";
                    if (!hashtable.ContainsKey(key))
                    {
                        Room tempRoom = new Room(x, y);
                        hashtable.Add(key, tempRoom);
                        rooms.Add(tempRoom);
                        count--;
                        break;
                    }

                    if (count <= 0)
                        break;
                }
            }
            i++;
        }
        return hashtable;
    }
}
```

Листинг А.2 – Генерация объектов комнат

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using Pathfinding;
public class MapObjectGenerator : MonoBehaviour
{
    Map map;
    System.Random rnd;
    [SerializeField] int minCount;
```

```
    [SerializeField] int maxCount;
    [SerializeField] int chance;
    [SerializeField] float width = 17.776f;
    [SerializeField] float heigh = 10;
    [SerializeField] GameObject[] allRooms;
    [SerializeField] GameObject[] allRoomStuffs;
    List<List<GameObject>> sortedRooms;
    // Start is called before the first frame update

    void Start()
    {
        sortedRooms = new List<List<GameObject>>();
        for (int i = 0; i < 4; i++)
        {
            sortedRooms.Add(new List<GameObject>());
        }
        foreach (var room in allRooms)
        {
            var roomScript = room.GetComponent<RoomObject>();
            int directCount = roomScript.DirectionsCount() - 1;
            sortedRooms[directCount].Add(room);
        }
        int seed = gameObject.GetComponent<Seed>().seedNumber;
        rnd = gameObject.GetComponent<Seed>().rnd;
        rnd = new System.Random(seed);
        int roomsCount = rnd.Next(minCount, maxCount);
        map = new Map(rnd, roomsCount, chance);
        var hashtable = map.GenerateRooms();
        CreateRooms(hashtable);
    }

    void CreateRooms(Hashtable _hashtable)
    {
        Room[] directions = new Room[4];
        directions[0] = new Room(1, 0);  //East
        directions[1] = new Room(0, 1);  //North
        directions[2] = new Room(-1, 0); //West
        directions[3] = new Room(0, -1); //South
        foreach (var hashKey in _hashtable.Keys)
        {
            Room tempRoom = _hashtable[hashKey] as Room;
            bool[] directionsBool = new bool[4];
            int directionsCount = -1;
            for (int i = 0; i < 4; i++)
            {
                string key = $"{directions[i].X +
tempRoom.X},{directions[i].Y + tempRoom.Y}";
                if (_hashtable.Contains(key))
                {
                    directionsBool[i]=true;
```

```
                    directionsCount++;
                }
            }
            List<GameObject> possibleRooms = new List<GameObject>();
            foreach (var item in sortedRooms[directionsCount])
            {
                if
(item.GetComponent<RoomObject>().DirectionsEqual(directionsBool[0],
directionsBool[1], directionsBool[2], directionsBool[3]))
                {
                    possibleRooms.Add(item);
                }
            }
            int index = rnd.Next(possibleRooms.Count);
            var finalRoomObject = possibleRooms[index];
            var obj = Instantiate(finalRoomObject, new Vec-
tor2(tempRoom.X * width, tempRoom.Y * heigh), final-
RoomObject.transform.rotation);
            index = rnd.Next(allRoomStuffs.Length);
            Instantiate(allRoomStuffs[index], obj.transform);
        }
        var path = gameObject.GetComponent<AstarPath>();

        path.Scan();

    }
}
```