МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий (ИИТ)
Кафедра математического обеспечения и стандартизации информационных технологий (МОСИТ)

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ

по дисциплине «Тестирование и верификация программного обеспечения»

Практическое занятие № 1

Студенты группы *ИКБО-43-23 Оганесян Д. К.*
Кощеев М.И.
Гуткович А.А.
Иванов А.В.
Улзетуев А.С.

(подпись)

Преподаватель *Ильичев Г.П.*

(подпись)

Отчет представлен «26»__сентября_2025 г.

Москва 2025 г.

СОДЕРЖАНИЕ

1. Техническое задание проекта нашей команды

1. Введение

1.1. Наименование программы

Наименование - VOXGORE.

1.2. Краткая характеристика области применения

Программа представляет собой компьютерную игру в жанре 3D Shoot 'em up и предназначена для развлечения пользователей. Игра ориентирована на любителей динамичных экшен-игр с элементами выживания против волн противников.

2. Основание для разработки, нормативные и исходные документы

2.1. Основание для проведения разработки

Основанием для разработки является повышенная потребность население в компьютерных играх.

2.2. Наименование и условное обозначение темы разработки

Наименование темы: «Разработка 3D-шутера VOXGORE».

3. Назначение разработки

3.1. Функциональное назначение

Функциональное назначение программы — предоставить пользователю

интерактивный игровой опыт.

3.2. Эксплуатационное назначение

Программа предназначена для эксплуатации на персональных компьютерах под управлением ОС Windows. Конечные пользователи — игроки, желающие провести время в динамичном шутере.

3.3. Задачи, решаемые в ходе разработки

- Реализация игровой логики (движение, стрельба, столкновения).
- Разработка ИИ противников.
- Создание системы генерации волн.
- Реализация пользовательского интерфейса (главное меню, HUD).

4. Требования к программе

4.1. Требования к функциональным характеристикам

4.1.1 Требования к составу выполняемых функций

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

1. Запуск и отображение главного меню.
2. Управление персонажем с клавиатуры и мыши.
3. Генерацию волн противников.
4. Систему подбора оружия и улучшений
5. Отслеживание игрового состояния (здоровье, волны, оружие)
6. Обработку столкновений и физику.
7. Сохранение и загрузку настроек.

4.1.2 Требования к организации входных данных

Входные данные программы должны быть организованы в виде:

- Пользовательский ввод (клавиатура, мышь).

4.1.3 Требования к организации выходных данных

Выходные данные:

- Графический вывод на экран (игровая сцена, интерфейс).
- Звуковое сопровождение.

4.2 Требования к надежности

4.2.1 Требования к обеспечению надежного (устойчивого) функционирования программы

Надежное функционирование программы должно быть обеспечено выполнением совокупности организационно-технических мероприятий:

1. Программа не должна завершаться аварийно при корректных действиях пользователя.
2. Необходимо обеспечить стабильный FPS на целевых конфигурациях.

4.2.2 Время восстановления после отказа

Перезапуск игры после сбоя должен занимать не более 10 секунд.

4.2.3 Отказы из-за некорректных действий пользователя

Программа должна обрабатывать некорректный ввод без аварийного завершения.

4.3 Условия эксплуатации

4.3.1 Требования к видам обслуживания

Обслуживание не требуется, за исключением обновлений контента или исправления ошибок.

4.3.2 Требования к численности и квалификации персонала

Для эксплуатации необходим только конечный пользователь с базовыми навыками управления ПК.

4.4 Требования к составу и параметрам технических средств

Минимальные системные требования:

- ОС: Windows 10
- Процессор: Intel Core i3
- ОЗУ: 2 ГБ
- Видеокарта: с поддержкой Vulkan 1.0
- Место на диске: 110 МБ

Рекомендуемые системные требования:

- ОС: Windows 10+
- Процессор: Intel Core i5
- ОЗУ: 4 ГБ
- Видеокарта: с поддержкой Vulkan 1.2
- Место на диске: 110 МБ

4.5 Требования к информационной и программной совместимости

4.5.1 Требования к информационным структурам и методам решения

Данные должны быть организованы в виде сцен (.tscn), скриптов (.gd) и ресурсов Godot Engine.

4.5.2 Требования к исходным кодам и языкам программирования

Исходный код должен быть написан на **GDScript**. Допускается использование шейдеров (.gdshader).

4.5.3 Требования к программным средствам, используемым программой

- Движок: Godot Engine 4.4.1.
- Графический API: Vulkan.
- Необходимые аддоны: MagicVoxel Importer, DiscordRPC и др. (согласно документации).

5. Требования к интерфейсу

5.1 Общие требования к интерфейсу

Интерфейс должен быть интуитивно понятным

5.2 Требования к главному меню

Главное меню должно отображаться при запуске и содержать:

- Кнопку «Начать матч».

5.3 Требования к игровому процессу

- Управление: WASD для движения, мышь для прицеливания и стрельбы.
- Отображение зоны арены с невозможностью выхода за ее пределы.
- Визуализация зон появления врагов до их спауна.

6. Технико-экономические показатели

6.1 Ожидаемый экономический эффект

Создание полноценного игрового продукта для портфолио и приобретение навыков командной разработки.

6.2 Затраты на разработку

Затраты минимизированы за счет использования бесплатного движка и силами студенческой команды.

7. Требования к программной документации

Состав программной документации должен включать в себя:

1. техническое задание;
2. пользовательская документация;
3. техническая документация;
4. текстовая документация;
5. системная документация.

8. Порядок контроля и приемки

8.1 Организация приемки

Приемка готового продукта осуществляется заказчиком.

8.2 Методы контроля качества

Контроль качества осуществляется методом чёрного ящика.

8.3 Тестовые сценарии

Тест-кейсы должны включать проверку:

- Запуска игры и главного меню.
- Управления персонажем.
- Спауна и поведения врагов.
- Системы оружия и улучшений.
- Границ игровой арены.

8.4 Критерии приемки

Критерием успешной приемки является успешное прохождение не менее 95% всех тест-кейсов.

8.5 Документация приемки

Акт приемки в опытную/промышленную эксплуатацию.

8.6 Завершение приемки

Подписание акта после успешной проверки.

9. Стадии и этапы разработки

9.1 План-график разработки

Разработка программного обеспечения должна быть выполнена в следующие сроки (Таблица 1).

Таблица 1. План-график разработки

№	Наименование этапа	Срок исполнения	Исполнитель
1	Проектирование архитектуры и интерфейса	3 рабочих дней	Разработчик
2	Реализация базового геймплея	5 рабочих дней	Разработчик
3	Создание контента (модели, анимации)	3 рабочих дня	Разработчик
4	Тестирование и отладка	3 рабочих дней	Разработчик
5	Подготовка документации и сборка	1 рабочий день	Разработчик

9.2 Форматы представления документации

Исходный код: файлы .gd, .tscn, .gdshader.

Документация: PDF.

2. Дополнительная документация проекта нашей команды

1. ТЕХНИЧЕСКАЯ ДОКУМЕНТАЦИЯ

Архитектура

Структура Godot проекта состоит из многих стандартных папок:

- addons - предназначена для размещения и управления внешними плагинами (аддонами) для вашего проекта. Эти плагины могут добавлять новую функциональность или изменять поведение редактора.
- media - используется для хранения различных медиафайлов, таких как изображения, звуки, видео и другие ресурсы, которые используются в игре или приложении.
- scenes - используется для хранения сцен проекта, которые представляют собой основные строительные блоки игры. Сцены могут быть уровнями игры, интерфейсами пользователя, объектами или любыми другими элементами, которые вы хотите повторно использовать. Эти файлы имеют расширение **.tscn.**
- scripts - используется для хранения скриптов, написанных на языке GDScript, их файлы имеют расширение .gd. Эти скрипты добавляются к узлам в сценах и определяют их поведение.
- shaders - используется для хранения шейдеров, которые представляют собой специальные программы, написанные на языке шейдеров Godot. Их файлы имеют расширение .gdshader.

Основные скрипты в папке Scripts являются:

- player.gd – программирование игрока.
- bloodsucker.gd – программирование противника.

2. ТЕСТОВАЯ ДОКУМЕНТАЦИЯ

Тесты окружения:

1. Персонаж не должен иметь возможность выйти за пределы забора, огораживающего арену с врагами.
2. Персонаж не должен иметь возможность нажать на кнопку начала волны, пока на арене есть противники.

Тесты персонажа:

1. Моделька персонажа должна меняться при изменении количества жизней.
2. При смерти скорость персонажа должна быть равна нулю и он не должен двигаться.

Тесты оружия:

1. На каждом виде оружия должна корректно работать анимация стрельбы: при зажатии кнопки выстрела проигрывается анимация стрельбы, при отпускании кнопки выстрела анимация прекращается.
2. При перезарядке и подборе одинакового оружия таймер перезарядки должен сбрасываться.

Тесты врагов:

1. Враги не должен иметь возможность выйти за пределы забора, огораживающего арену.
2. Места появления врагов должны подсвечиваться красными кругами за пару секунд до их появления.
3. Враги не должны появляться за пределами арены

3. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Введение

Данное подробное руководство создано специально для всех игроков, желающих погрузиться в игру VOXGORE. Оно поможет вам быстро освоиться с основными механиками игры и начать получать удовольствие от игрового процесса.

Установка и запуск

Чтобы начать играть в **VOXGORE**, следуйте простой инструкции по установке:

- Скачайте архив с файлами игры
- Распакуйте содержимое архива в удобное для вас место на компьютере
- Найдите и запустите исполняемый файл **voxgore.exe**

После запуска игры перед вами откроется главное меню, где вы сможете выбрать одно из двух действий:

- Начать новый матч
- Выйти из игры

Игровой процесс

VOXGORE представляет собой динамичный бесконечный шутер в жанре **Shoot 'em up**. Основная задача игрока — **уничтожение волн противников**, которые появляются одна за другой.

После каждой успешно пройденной волны у вас есть шанс получить различные бонусы:

- Новое оружие
- Улучшения характеристик

Игра продолжается до тех пор, пока игрок способен отражать атаки противников, что делает каждую сессию уникальной и непредсказуемой.

4. СИСТЕМНАЯ ДОКУМЕНТАЦИЯ

1. Общие сведения

Наименование продукта: “VOXGORE”

Версия: 0.0.0.1

Дата сборки: 08.09.2025

Движок: Godot Engine 4.4.1

Тип приложения: 3D-шутер

Системные требования:

Минимальные:

- ОС: Windows 10
- Процессор: Intel Core i3
- Память: 2 GB RAM
- Видеокарта: Vulkan 1.0 совместимая
- Место на диске: 110 MB
- Поддержка Vulkan

Рекомендуемые:

- ОС: Windows 10+
- Процессор: Intel Core i5
- Память: 4 GB RAM
- Видеокарта: Vulkan 1.2 совместимая
- Место на диске: 110 MB
- Поддержка Vulkan

2. Установка и развертывание

- Разархивируйте пакет voxgore_export.rar
- Запустите Voxgore Bloodsuckers.exe

3. Установка пакета игры для среды Godot Engine

1. Установите последнюю версию Godot Engine (https://store.steampowered.com/app/404790/Godot_Engine/)
2. Разархивируйте VOXGORE-Developers-rep.rar и откройте project.godot в Godot Engine
3. Включите аддоны (рис 1-4)

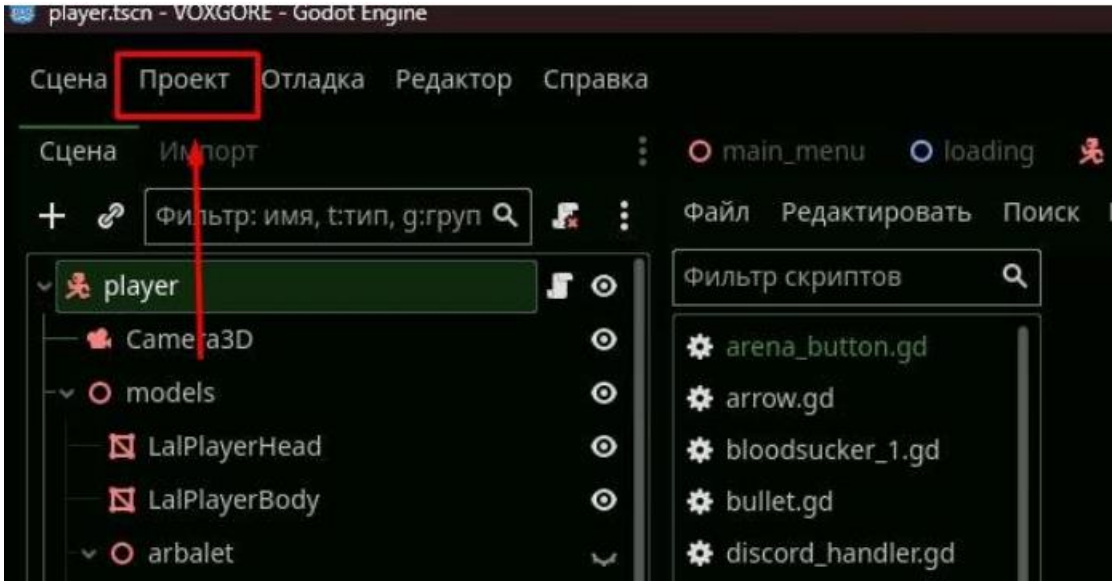

Рисунок 1.

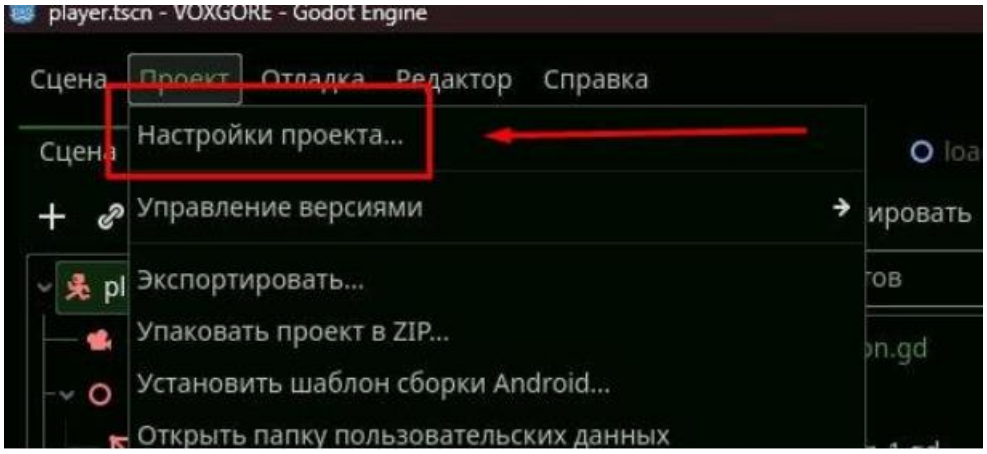

Рисунок 2.

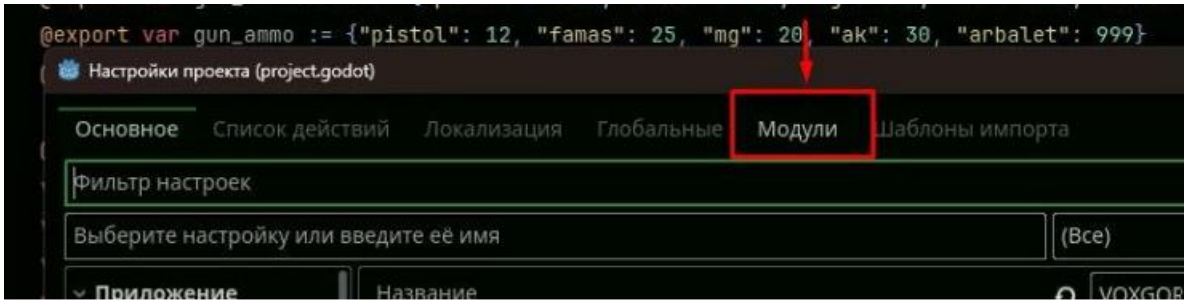

Рисунок 3.

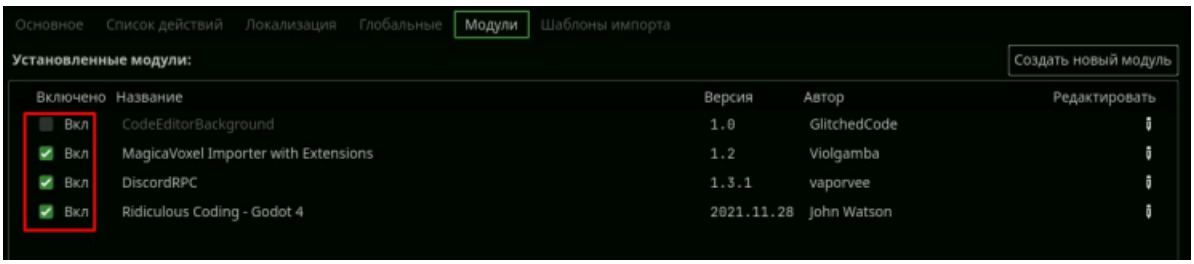

Рисунок 4.

3. Описание ошибок внесённое в ПО

1. Отсутствие коллизии забора для игрока в левом нижнем углу арены.

   Тип – логический

   Способ обнаружения – попытаться пройти сквозь забор и выйти за пределы арены.
2. Отсутствие коллизии забора для противников в левом нижнем углу арены.

   Тип – логический

   Способ обнаружения – попытаться заманить противника за пределы арены.
3. На оружии “АК-47” анимация стрельбы не прекращается при прекращении стрельбы.

   Тип – логический

   Способ обнаружения – подобрать “АК-47”, начать стрельбу, прекратить стрельбу.
4. При подборе одинакового оружия во время перезарядки сбрасывается анимация перезарядки, но не сама перезарядка

   Тип – логический

   Способ обнаружения – подобрать уже имеющееся оружие во время перезарядки
5. При анимации смерти перемещение игрока не сбрасывается

   Тип – логический

   Способ обнаружения – умереть во время движения

4. Техническое задание и документация проекта другой команды

1. Техническое задание

1. Введение

1.1. Наименование программы

Наименование - iBot.

Telegram-бот для Информационной поддержки Творческо-организационного подразделения (ТОП) ЦКТ РТУ МИРЭА.

1.2. Краткая характеристика области применения

Программа предназначена к применению в информационно-справочном обслуживании членов ТОПа и Центра культуры и творчества.

Бот предназначен для оперативного предоставления актуальной информации о структуре, деятельности, руководстве и правилах ТОП.

2. Основание для разработки, нормативные и исходные документы

2.1. Основание для проведения разработки

Основанием для разработки является внутренняя потребность Творческо-организационного подразделения ЦКТ РТУ МИРЭА в автоматизации процесса предоставления справочной информации.

2.2. Наименование и условное обозначение темы разработки

Наименование темы разработки - Информационная поддержка Творческо-организационного подразделения (ТОП) ЦКТ РТУ МИРЭА.

Условное обозначение темы разработки (шифр темы) - “ТОП-Бот-001”

3. Назначение разработки

3.1. Функциональное назначение

Функциональным назначением программы является автоматизация процесса информационно-справочного обслуживания член

Творческо-организационного подразделения (ТОП) и Центра культуры и Творчества путём предоставления структурированных данных о деятельности, структуре, руководстве и правилах ТОП через мессенджер Telegram.

3.2. Эксплуатационное назначение

Программа должна эксплуатироваться в Творческо-организационном подразделении Центра культуры и творчества (ЦКТ) РТУ МИРЭА. Конечными пользователями программы должны являться члены ТОП, студенты и сотрудники университета, нуждающиеся в получении актуальной справочной информации о подразделении.

3.3. Задачи, решаемые в ходе разработки

- Структурирование и централизация справочной информации (контакты, цели, правила, структура).
- Сокращение времени на поиск информации о руководителях коллективов.
- Обеспечение легкого и интуитивно понятного доступа к информации через популярный мессенджер Telegram.

4. Требования к программе

4.1. Требования к функциональным характеристикам

4.1.1 Требования к составу выполняемых функций

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

а) Функция запуска диалога с пользователем (команда /start).

б) Функция предоставления текстовой и графической информации о миссии и целях ТОП (раздел «Кто мы»).

в) Функция отображения внутренних правил ТОП (раздел «Заповеди»).

г) Функция предоставления информации об отделах ТОП (руководитель, контакты, задачи).

д) Функция отображения корпоративного словаря терминов.

е) Функция обработки ошибок (уведомление пользователя о проблемах, логирование).

4.1.2 Требования к организации входных данных

Входные данные программы должны быть организованы в виде:

- Текстовых файлов (в формате .py) с данными для разделов.
- Структурированных данных (словари Python) для хранения информации об отделах.

4.1.3 Требования к организации выходных данных

Выходные данные программы должны быть представлены в виде сообщений, отправляемых через API Telegram, включая:

- Текстовые сообщения с форматированием Markdown.
- Сообщения об ошибках.

4.2 Требования к надежности

4.2.1 Требования к обеспечению надежного (устойчивого) функционирования программы

Надежное функционирование программы должно быть обеспечено выполнением совокупности организационно-технических мероприятий:

а) Организацией бесперебойного питания технических средств.

б) Использованием лицензионного программного обеспечения.

в) Регулярным выполнением требований по защите информации и испытаний на наличие вредоносного программного обеспечения.

4.2.2 Время восстановления после отказа

Время восстановления после отказа, вызванного сбоем электропитания или иными внешними факторами, не должно превышать 10 минут при условии соблюдения условий эксплуатации.

Время восстановления после отказа, вызванного неисправностью технических средств или крахом операционной системы, не должно превышать времени, требуемого на устранение неисправностей и переустановку программных средств.

4.2.3 Отказы из-за некорректных действий пользователя

Отказы программы возможны вследствие некорректных действий пользователя. Во избежание этого необходимо обеспечить работу программы с корректными входными данными и реализовать механизмы обработки исключений.

4.3 Условия эксплуатации

4.3.1 Требования к видам обслуживания

Программа не требует проведения специальных видов обслуживания, за исключением обновления контента и мониторинга логических ошибок.

4.3.2 Требования к численности и квалификации персонала

Для работы программы требуется:

- Системный администратор (обеспечивает работоспособность сервера, установку ОС и зависимостей).
- Конечный пользователь (взаимодействует с программой через Telegram-клиент).

Системный администратор должен иметь навыки администрирования ОС Linux/Windows и установки Python-окружения. Конечный пользователь должен обладать базовыми навыками работы с мессенджером Telegram.

4.4 Требования к составу и параметрам технических средств

В состав технических средств должен входить сервер (виртуальный или физический), включающий:

- Процессор с тактовой частотой не менее 1 ГГц.

- Оперативную память объемом не менее 512 МБ.
- Постоянную память объемом не менее 5 ГБ.
- Доступ к сети Интернет.

4.5 Требования к информационной и программной совместимости

4.5.1 Требования к информационным структурам и методам решения

Программа должна использовать структуры данных Python (словари, списки) для хранения информации. Данные должны быть организованы в соответствии с логикой работы бота.

4.5.2 Требования к исходным кодам и языкам программирования

Исходный код программы должен быть реализован на языке Python версии 3.10 и выше. Для взаимодействия с Telegram должен использоваться пакет python-telegram-bot.

4.5.3 Требования к программным средствам, используемым программой

Программа должна работать под управлением ОС Linux/Windows с установленным интерпретатором Python и необходимыми зависимостями.

5. Требования к интерфейсу

5.1 Общие требования к интерфейсу

Интерфейс пользователя реализуется исключительно средствами мессенджера Telegram через систему меню, кнопок и текстовых команд.

5.2 Требования к главному меню

После команды /start пользователю должно быть отправлено текстовое приветствие и отображена клавиатура с основными разделами (ReplyKeyboardMarkup).

5.2.1 Структура главного меню

Клавиатура главного меню должна иметь вид:

['Кто мы' 'Заповеди']

['Отделы' 'Словарик']

5.3 Требования к меню отделов

При переходе в раздел «Отделы» пользователю должна быть отображена новая клавиатура для выбора конкретного отдела.

5.3.1 Структура меню отделов

Клавиатура меню отделов должна иметь вид:

['Креатив' 'Документы']

['Назад']

['Перезапустить']

5.4 Требования к форматированию сообщений

Все текстовые сообщения должны использовать форматирование Markdown для выделения заголовков и ключевых элементов (жирный шрифт).

6. Технико-экономические показатели

6.1 Ожидаемый экономический эффект

Ожидаемый экономический эффект от внедрения системы заключается в сокращении временных затрат руководства ТОП на рутинное информирование участников подразделения не менее чем на 70%.

6.2 Затраты на разработку

Разработка осуществляется силами сотрудников/студентов университета без привлечения коммерческих подрядчиков, что минимизирует затраты.

7. Требования к программной документации

Состав программной документации должен включать в себя:

а) техническое задание;

б) пользовательская документация;

в) техническая документация;

г) текстовая документация;

д) системная документация.

8. Порядок контроля и приемки

8.1 Организация приемки

Приемка готового продукта осуществляется заказчиком.

8.2 Методы контроля качества

Контроль качества осуществляется методом чёрного ящика.

8.3 Тестовые сценарии

Для приемки Исполнитель предоставляет Заказчику выполненные тестовые сценарии, включающий не менее 5 тест-кейсов.

UC-01 Старт

Пользователь отправляет /start. Бот просит ввести имя. После ввода имени появляется главное меню (Рисунок 5)

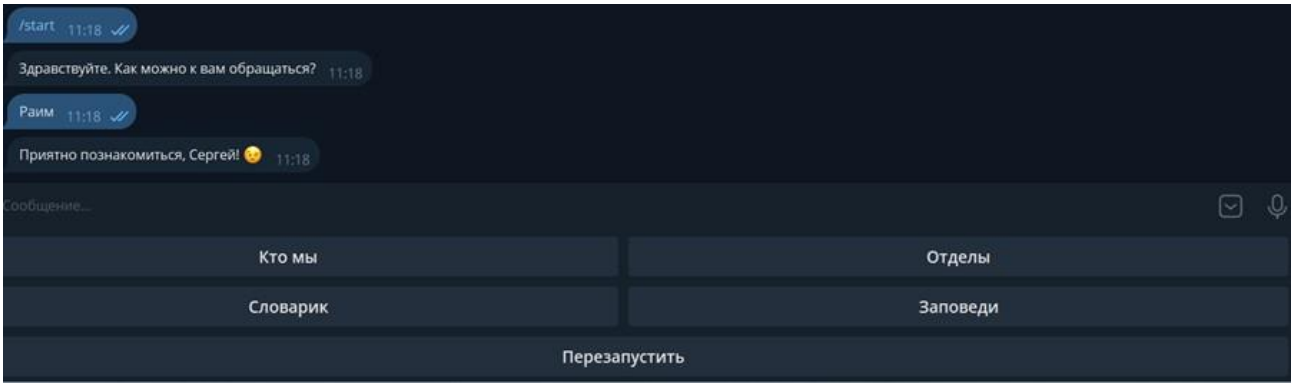

Рисунок 5 – Приветствие и главное меню

UC-02. «Кто мы»

Пользователь выбирает «Кто мы». Бот отправляет текст миссии и целей. Затем снова показывает главное меню (Рисунок 6).

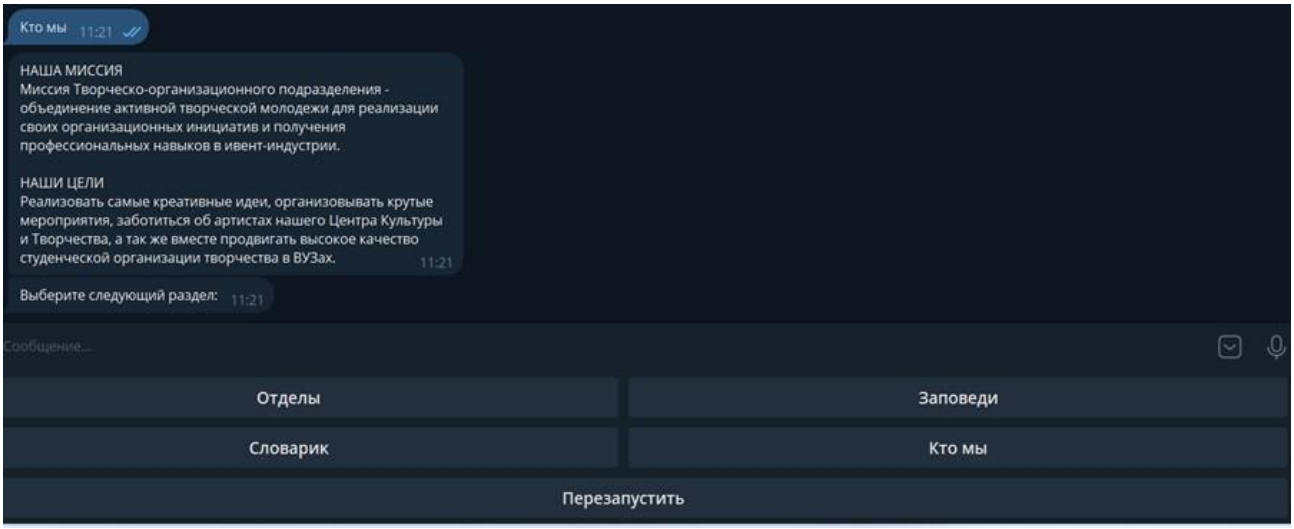

Рисунок 6 – Ознакомление по запросу

UC-03. «Заповеди»

Пользователь выбирает «Заповеди». Бот отправляет правила поведения. Затем снова главное меню.

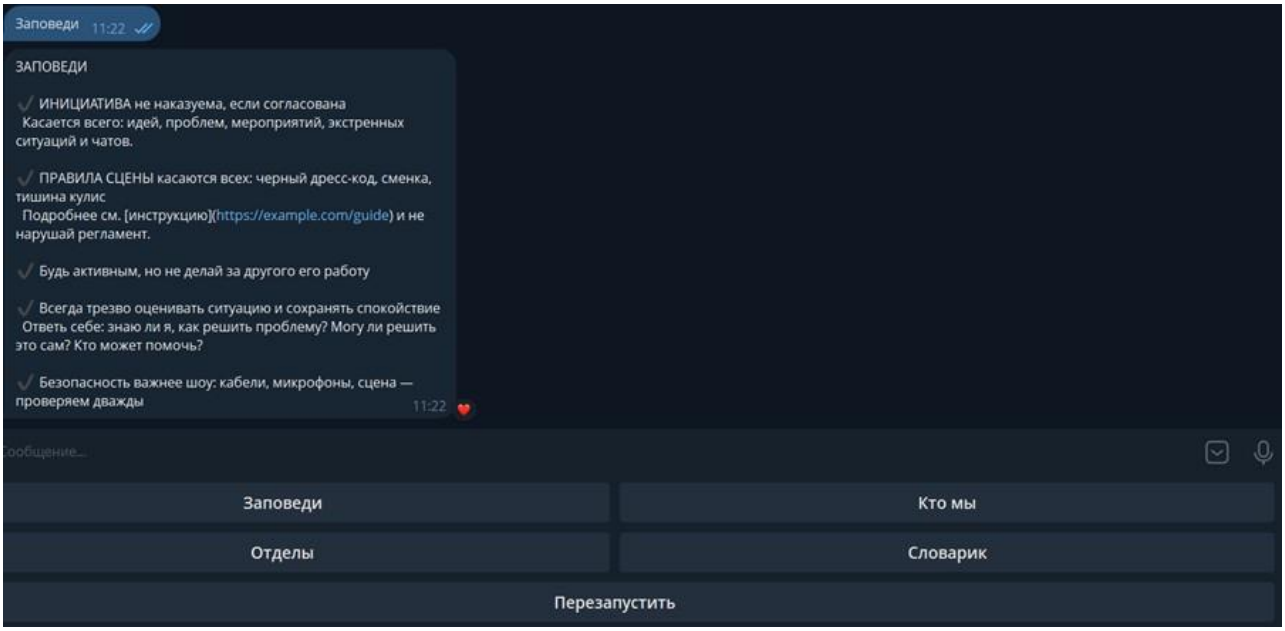

Рисунок 7 – Отображение правил по запросу

UC-04. Просмотр отдела

Пользователь выбирает «Отделы». Бот показывает клавиатуру с отделами. При выборе «Креатив» или «Документы» — отправляет описание отдела.

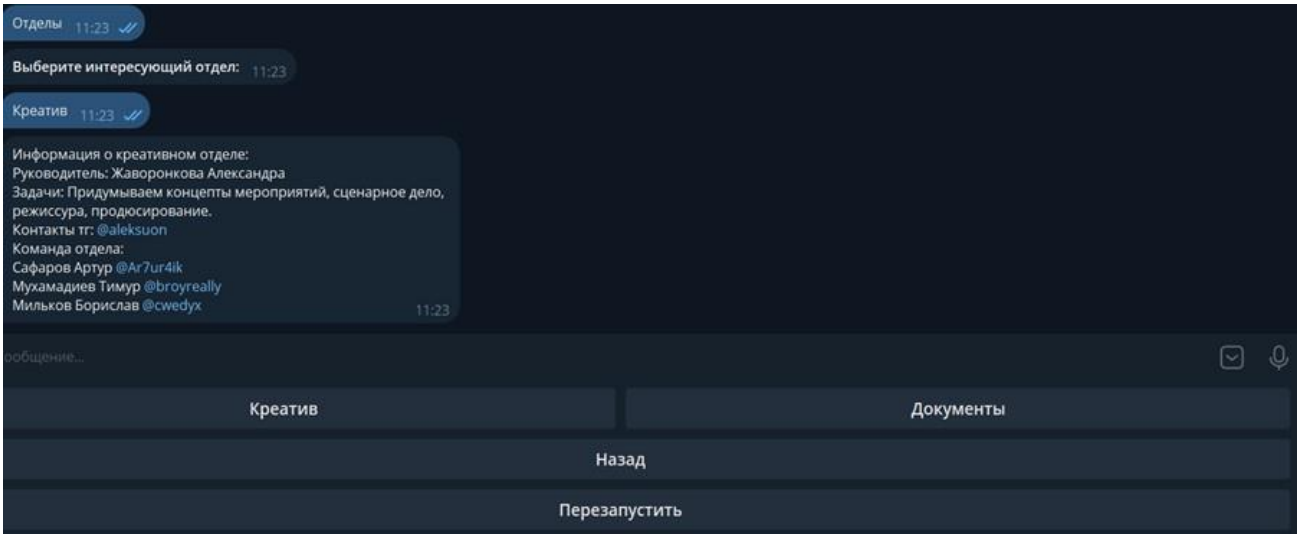

Рисунок 8 – Отображение выбранного отдела по запросу

UC-05. Перезапуск

Пользователь нажимает «Перезапустить». Бот возвращает к главному меню.

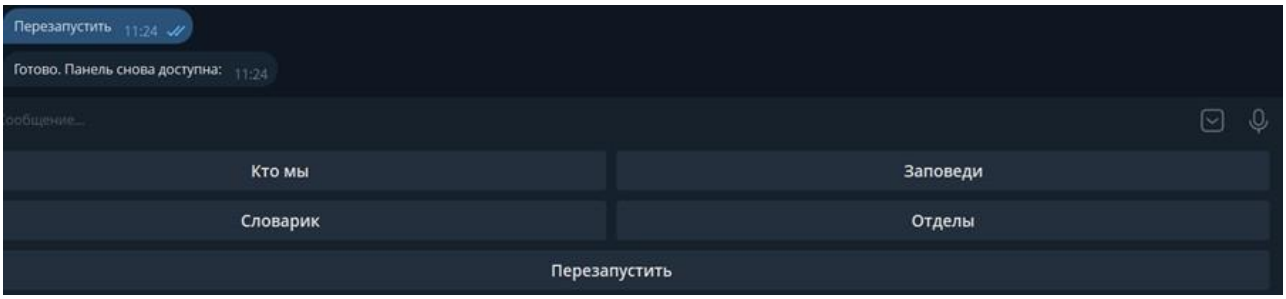

Рисунок 9 – Перезапуск и возвращение в первоначальный вид главного меню

8.4 Критерии приемки

Критерием успешной приемки является успешное прохождение не менее 95% всех тест-кейсов.

8.5 Документация приемки

После успешного прохождения испытаний подписывается Акт о приемке программного обеспечения в опытную эксплуатацию.

8.6 Завершение приемки

По итогам успешной опытной эксплуатации в течение установленного срока подписывается Акт о приемке программного обеспечения в промышленную эксплуатацию.

9. Стадии и этапы разработки

9.1 План-график разработки

Разработка программного обеспечения должна быть выполнена в следующие сроки (Таблица 1).

Таблица 1. План-график разработки

№	Наименование этапа	Срок исполнения	Исполнитель
1	Анализ требований и проектирование архитектуры	5 рабочих дней	Разработчик
2	Написание кода и модульное тестирование	10 рабочих дней	Разработчик
3	Наполнение контентом	3 рабочих дня	Заказчик
4	Пилотная эксплуатация и приемочные испытания	5 рабочих дней	Совместно
5	Ввод в промышленную эксплуатацию	1 рабочий день	Разработчик

9.2 Форматы представления документации

Вся документация должна быть предоставлена в электронном виде в форматах PDF (для руководств) и .py (для исходного кода).

2. ТЕХНИЧЕСКАЯ ДОКУМЕНТАЦИЯ

1. Архитектура

Бот построен на библиотеке python-telegram-bot и работает как конечный автомат (FSM) с двумя состояниями:

- MAIN_MENU — главное меню

- DEPARTMENTS — меню отделов

Поток работы:

1. Пользователь вводит /start.

2. Бот спрашивает имя и переходит в MAIN_MENU.

3. В главном меню доступны кнопки: «Кто мы», «Заповеди», «Отделы», «Словарь», «Перезапустить».

4. Выбор «Отделы» переводит в состояние DEPARTMENTS.

5. «Перезапустить» возвращает к главному меню, «/cancel» завершает диалог.

Данные (миссия, заповеди, словарь, описания отделов) хранятся в словаре TEXTS.

Клавиатуры формируются через ReplyKeyboardMarkup.

2. Основные функции

- _build_shuffled_main_keyboard() — Собирает кнопки главного меню.

- start() — Приветствие, запрос имени, переход в MAIN_MENU.

- main_menu() — Обрабатывает выбор пользователя в главном меню: отправляет тексты или открывает раздел «Отделы».

- department_menu() — Показывает список отделов и их описание.

- restart_handler() — Возвращает к главному меню без повторного вопроса об имени.

- cancel() — Завершает работу и удаляет клавиатуру.

- main() — Инициализирует приложение, настраивает обработчики и запускает polling.

3. ТЕСТОВАЯ ДОКУМЕНТАЦИЯ

1. Юз-кейсы

UC-01. Старт

Пользователь отправляет /start. Бот просит ввести имя. После ввода имени появляется главное меню (Рисунок 1).

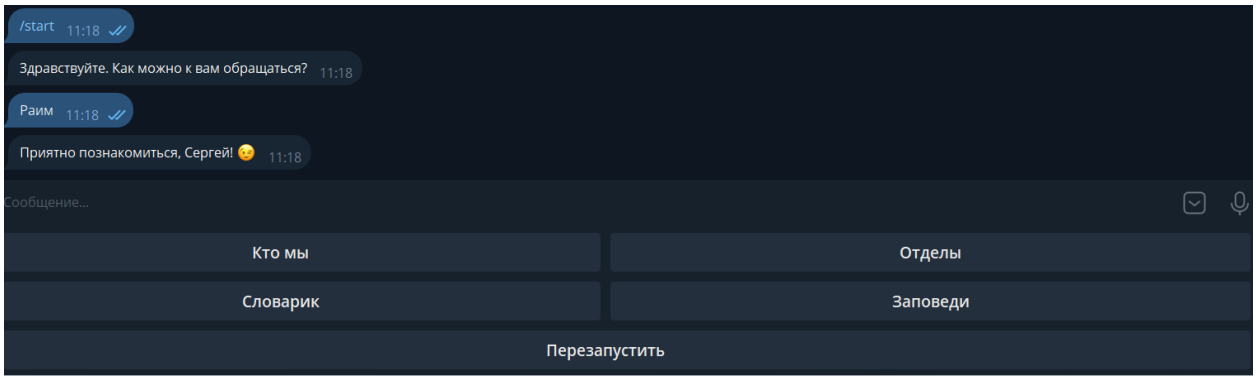

Рисунок 1 – Приветствие и главное меню

UC-02. «Кто мы»

Пользователь выбирает «Кто мы». Бот отправляет текст миссии и целей. Затем снова показывает главное меню (Рисунок 2).

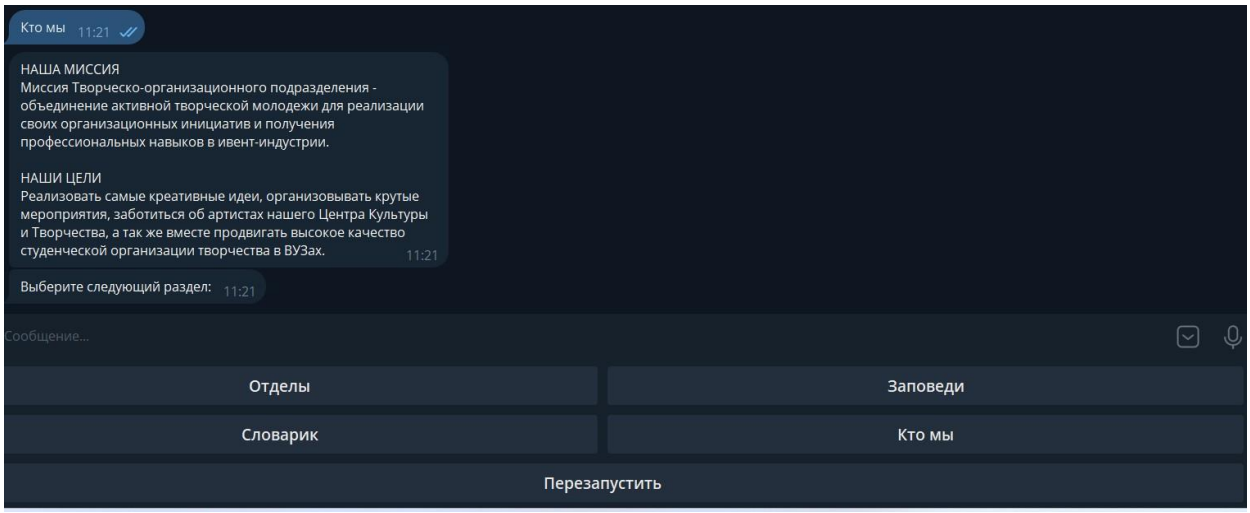

Рисунок 2 – Ознакомление по запросу

UC-03. «Заповеди»

Пользователь выбирает «Заповеди». Бот отправляет правила поведения. Затем снова главное меню (Рисунок 3).

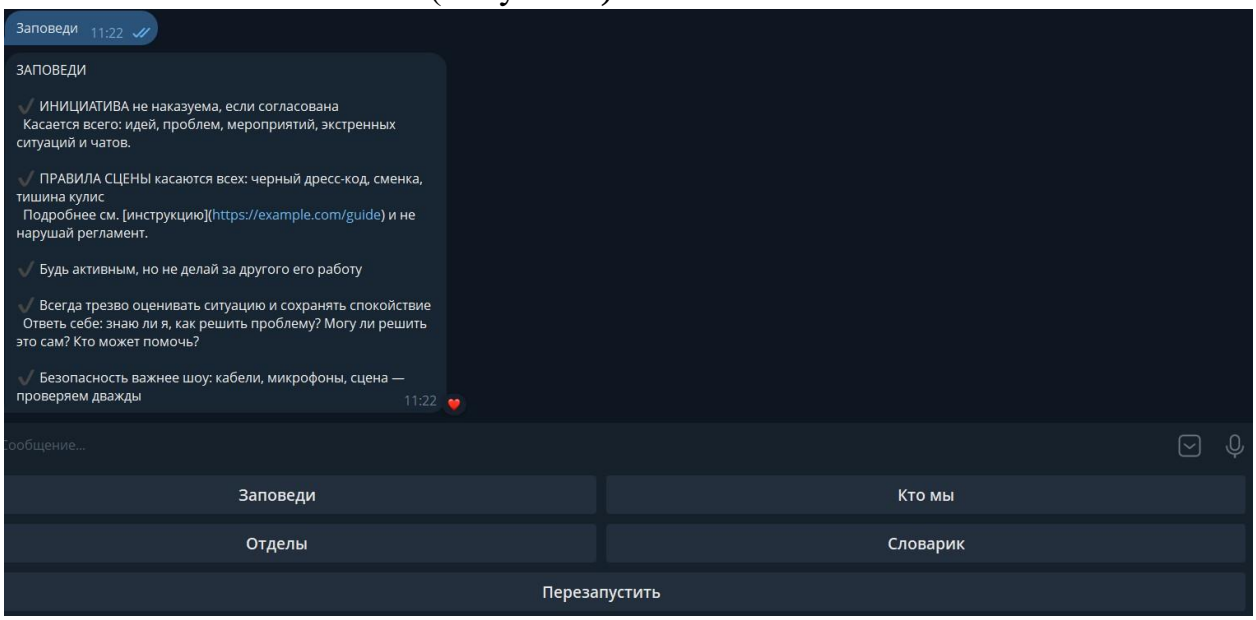

Рисунок 3 – Отображение правил по запросу

UC-04. Просмотр отдела

Пользователь выбирает «Отделы». Бот показывает клавиатуру с отделами. При выборе «Креатив» или «Документы» — отправляет описание отдела (Рисунок 4).

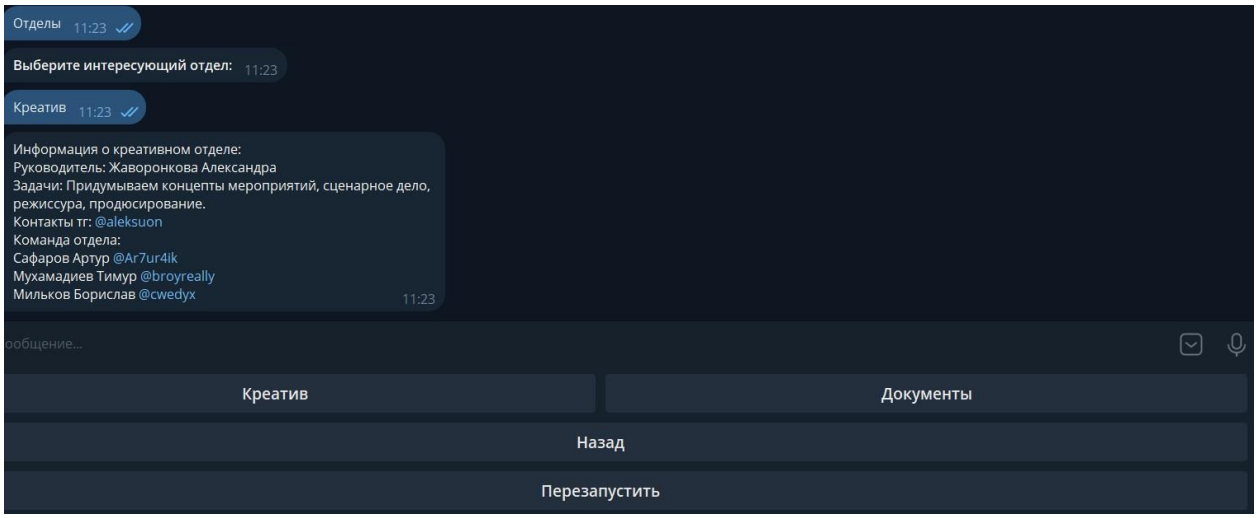

Рисунок 4 – Отображение выбранного отдела по запросу

UC-05. Перезапуск

Пользователь нажимает «Перезапустить». Бот возвращает к главному меню (Рисунок 5).

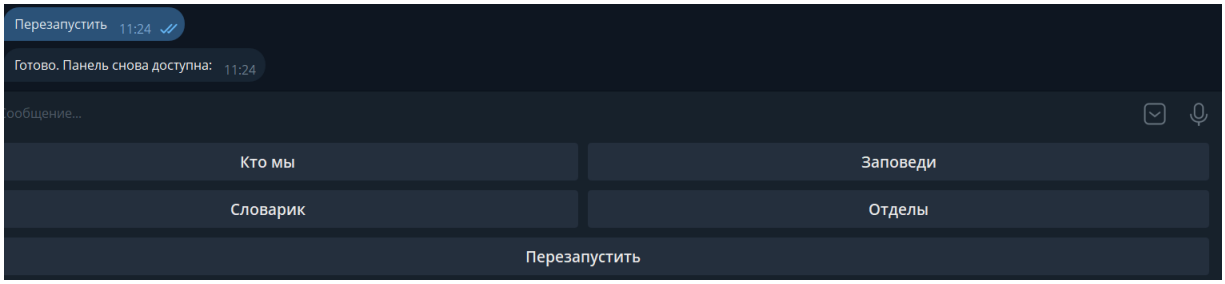

Рисунок 5 – Перезапуск и возвращение в первоначальный вид главного меню

4. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

1. Введение

Настоящее руководство предназначено для конечных пользователей телеграм-ботам iBot. Бот предоставляет легкий доступ к ключевой информации о описанном подразделении: его миссии, правилах внутренней работы, структуре отделов и корпоративной терминологии.

2. Начало работы

Для начала взаимодействия с ботом выполните следующие шаги:

- Инициализация. Нажмите кнопку START или введите команду /start в поле для сообщений.
- Знакомство. Бот запросит ваше имя. Введите любое удобное для обращения имя и отправьте сообщение.

3. Основное меню

После знакомства откроется главное меню, представленное панелью с четырьмя кнопками.

- Кнопка «Кто мы» – предоставляет текст с описанием миссии и стратегических целей подразделения.
- Кнопка «Заповеди» – отображает свод основных правил и принципов работы. В некоторых случаях форматирование текста (жирный шрифт, курсив) может отсутствовать.
- Кнопка «Отделы» – открывает меню выбора отделов для получения детальной информации.
- Кнопка «Словарик» – выводит обширный корпоративный словарь с расшифровкой специфических терминов, аббревиатур и названий локаций.
- Для навигации также всегда доступна кнопка «Перезапустить», которая возвращает пользователя в главное меню.

4. Работа с меню отделов

- В главном меню нажмите кнопку «Отделы».

- На экране появится новая панель с кнопками: «Креатив», «Документы» и «Назад».

- Для получения информации о конкретном отделе нажмите на соответствующую кнопку. Бот пришлет сообщение с данными о руководителе, задачах отдела и контактах.

- Для возврата к списку отделов используйте кнопку «Назад».

5. Решение возможных проблем

- Сброс текущего сеанса. Если бот перестал отвечать или вы заблудились в меню, введите команду /restart. Это вернет вас в главное меню без повторного запроса имени.

- Полный перезапуск. Чтобы начать взаимодействие с бота заново (включая этап знакомства), используйте команду /start.

- Завершение работы. Для завершения работы с ботом введите команду /cancel.

5. СИСТЕМНАЯ ДОКУМЕНТАЦИЯ

1. Что вам понадобится

Компьютер или сервер с выходом в интернет

Установленный Python (версия 3.7 или новее)

Доступ к приложению Telegram

2. Организация файлов

Скачайте bot.py

Переместите его в папку на любом доступном диске (C, D и т.д.)

3. Работа в командной строке

Введем условные обозначения:

foldername - название папки куда вы закинули бота

Запустите Powershell

Далее выполняйте команды по порядку:

cd foldername

Set-ExecutionPolicy -Scope Process -ExecutionPolicy Bypass (это одна длинная команда)

python -m venv .venv

.\.venv\Scripts\Activate.ps1

pip install --upgrade pip

pip install python-telegram-bot==20.3

python bot.py

Закрытие окна Powershell приведет к окончанию сессии.

Чтобы вновь ее запустить, достаточно будет ввести две команды:

cd foldername

python bot.py

4. Взаимодействие с ботом

Введите в поисковой строке в telegram:

@student_test_ugh_bot

Введите /start

Можно приступать к тестированию.

5. Результаты тестирования программного продукта другой команды

ТС-001. Проверка корректности сохранения имени пользователя

0. **Описание:**

Проверка функциональности сохранения и использования имени пользователя после приветствия

1. **Предварительные условия:**

- Бот запущен
- Диалог начат командой /start

3. **Шаги выполнения:**

- Шаг 1: Отправить команду /start
- Шаг 2: Ответить на приветственное сообщение ботом, указав имя “Андрей”
- Шаг 3: Дождаться ответа бота

4. **Ожидаемый результат:**

- Бот корректно запоминает указанное имя
- В последующих сообщениях бот обращается к пользователю по указанному имени

5. **Фактический результат:**

- Бот некорректно сохраняет имя, обращаясь к пользователю как “Влад”

6. **Статус:** Failed

7. **Прикрепленные материалы:**

- Скриншот диалога

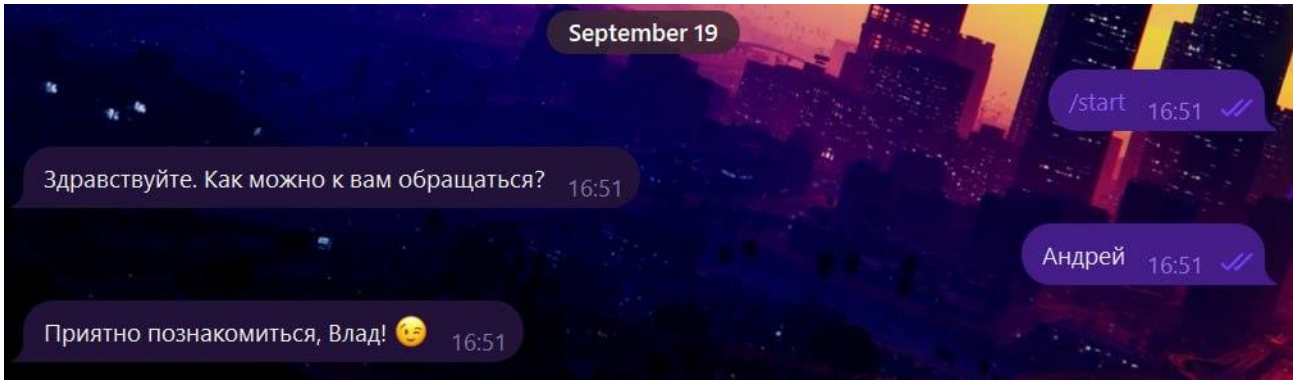

Рисунок 1 – Не запоминает имя

ТС-002. Проверка работы кнопки “Назад”

0. **Описание:**

Тестирование корректности работы кнопки возврата к предыдущему экрану

1. **Предварительные условия:**

- Бот запущен
- Открыта информация об отделе

3. **Шаги выполнения:**

- Шаг 1: Получить информацию об отделе
- Шаг 2: Нажать кнопку “Назад”
- Шаг 3: Проверить результат

4. **Ожидаемый результат:**

- Возврат к предыдущему экрану с основным меню

5. **Фактический результат:**

- Отображается сообщение “Вы уже в меню отделов”

6. **Статус:** Failed

7. **Прикрепленные материалы:**

- Скриншот диалога

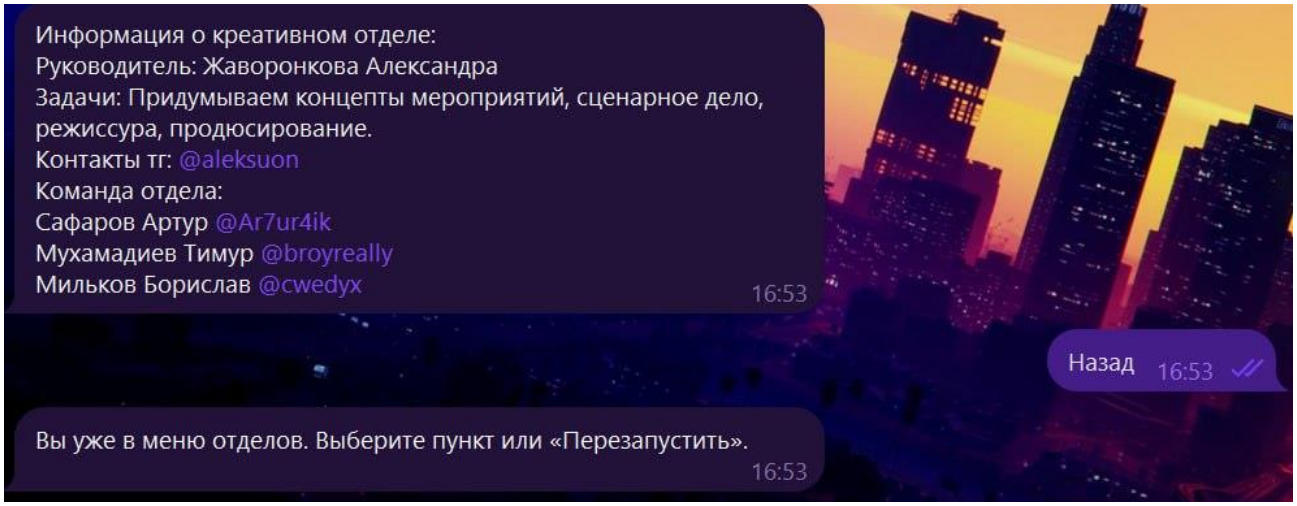

Рисунок 1 – Кнопка назад не возвращает назад

ТС-003. Проверка функционала “Словарик”

0. **Описание:**

Тестирование работоспособности раздела со словарем терминов

1. **Предварительные условия:**

- Бот запущен
- Основное меню доступно

3. **Шаги выполнения:**

- Шаг 1: Выбрать пункт “Словарик”
- Шаг 2: Дождаться ответа бота

4. **Ожидаемый результат:**

- Открытие раздела со словарем
- Отображение доступных терминов

5. **Фактический результат:**

- Сообщение об ошибке: “Пожалуйста, выберите один из предложенных вариантов”

6. **Статус:** Failed

7. **Прикрепленные материалы:**

- Скриншот диалога

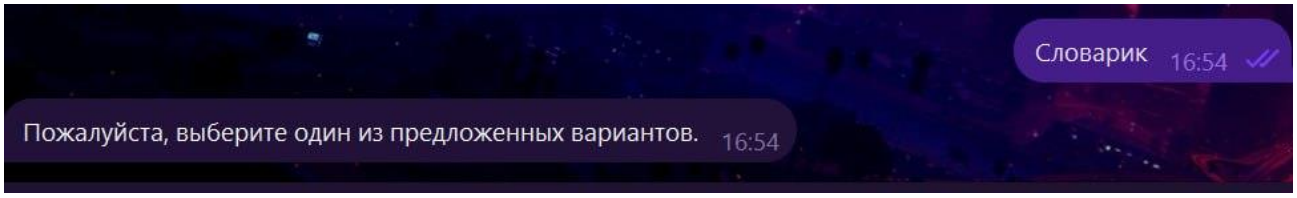

Рисунок 3 – Кнопка "словарик не работает"

ТС-004. Проверка обработки некорректных входных данных

0. **Описание:**

Тестирование реакции системы на ввод некорректных данных

1. **Предварительные условия:**

- Бот запущен
- Любое интерактивное меню открыто

3. **Шаги выполнения:**

- Шаг 1: Открыть любое меню
- Шаг 2: Ввести некорректный вариант ответа
- Шаг 3: Дождаться реакции бота

4. **Ожидаемый результат:**

- Корректная обработка ошибки
- Предложение повторить ввод

5. **Фактический результат:**

- Зацикливание интерфейса
- Повторное отображение того же экрана

6. **Статус:** Failed

7. **Прикрепленные материалы:**

- Скриншот зацикливания

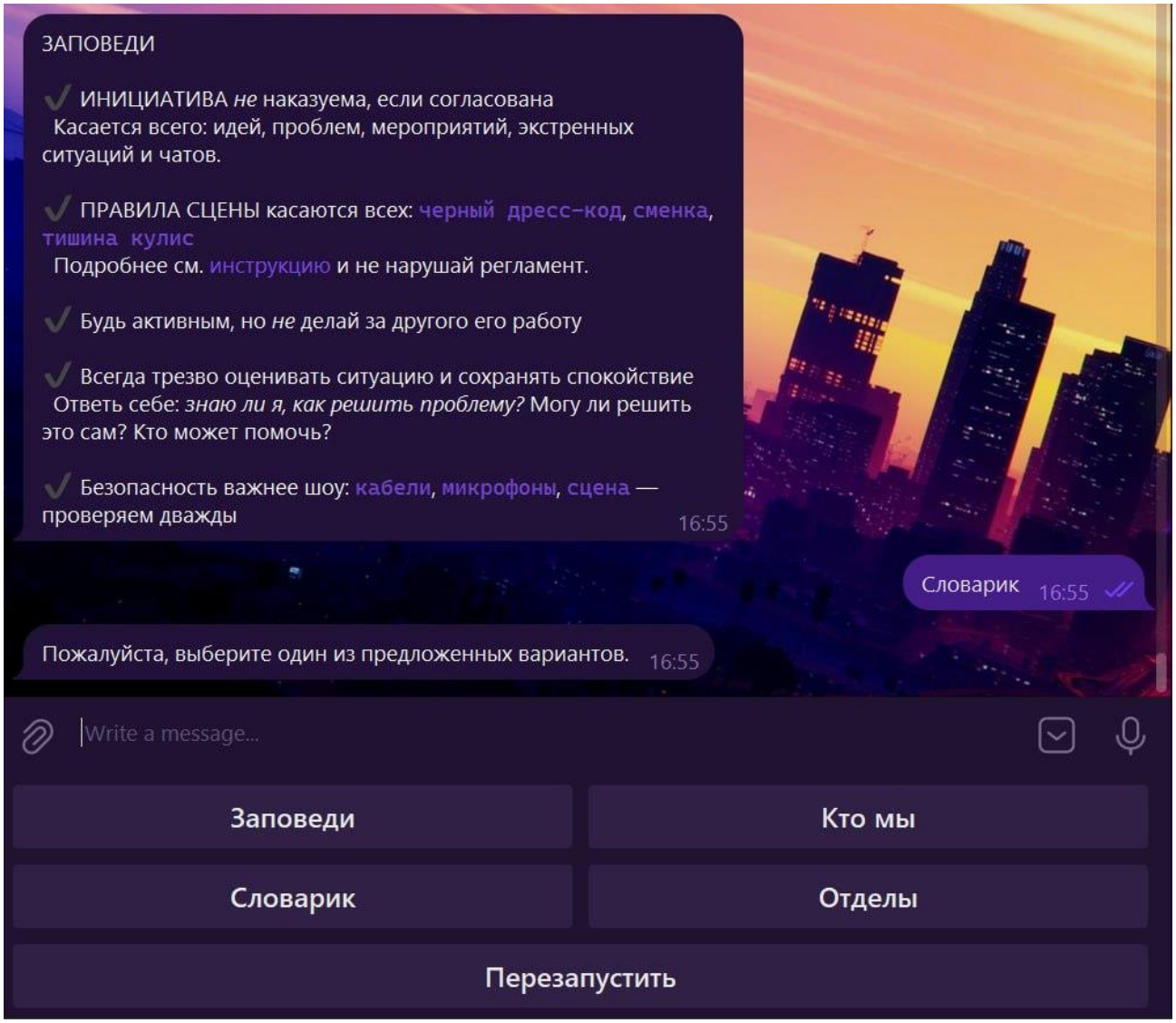

Рисунок 4 – При выборе неподходящего варианта происходит зацикливание

6. Анализ документации другой команды

Анализ технической документации

1. Общие замечания по структуре

- **Отсутствие версий документации.** Не указаны версии документов, даты актуализации
- **Неструктурированность.** Отсутствуют ссылки между разделами документации
- **Недостаточная детализация.** Многие процессы описаны поверхностно

2. Анализ технической документации

- **Архитектурный раздел:**
    - Отсутствует описание используемых паттернов проектирования
    - Нет информации о системе логирования
    - Не описаны механизмы обработки ошибок
- **Функциональный раздел:**
    - **Неполнота описания функций:**
        - Нет информации о параметрах функций
        - Отсутствует описание возвращаемых значений
        - Не описаны возможные исключения

- **Двусмысленные формулировки:**
    - “Данные хранятся в словаре TEXTS” - не указано где именно находится этот словарь
    - “Клавиатуры формируются через ReplyKeyboardMarkup” - нет описания формата

3. Анализ тестовой документации

- **Юз-кейсы:**
- **Недостаточная детализация:**
    - Отсутствуют предусловия для каждого кейса
    - Нет описания ожидаемых результатов
    - Отсутствует информация о граничных случаях

4. Анализ руководства пользователя

- **Неполнота информации:**
    - Отсутствует описание обработки ошибок
    - Нет информации о времени отклика системы
    - Не описаны ограничения на ввод данных

5. Анализ системной документации

- **Технические недочеты:**
    - Указана конкретная версия python-telegram-bot (20.3), что может привести к проблемам совместимости
    - Нет описания всех зависимостей проекта

    - Отсутствует информация о требованиях к окружению
- **Неясности в инструкциях:**
    - Не описано, что делать при ошибках установки
    - Нет информации о возможных проблемах при запуске

7. Заключение

Проведенный анализ показал, что программный продукт демонстрирует частичное соответствие техническому заданию. Базовая функциональность реализована, однако обнаружены существенные отклонения от требований: система некорректно обрабатывает пользовательские данные, кнопка «Назад» работает со сбоями, отсутствует заявленный функционал раздела «Словарик», а при некорректном вводе данных система ведет себя нестабильно.

Документация требует серьезной доработки: отсутствует система версионирования, технические описания недостаточно детализированы, нет полной информации о зависимостях проекта и механизмах логирования. В документации также не описаны процедуры обработки ошибок и восстановления после сбоев.

На основании проведенного анализа рекомендуется:

- устранить критические ошибки в работе с пользовательскими данными;
- реализовать корректную работу кнопки «Назад»;
- добавить функционал раздела «Словарик»;
- улучшить механизмы обработки некорректных входных данных;
- внедрить систему версионирования документации;
- дополнить документацию подробными описаниями обработки ошибок;
- описать механизмы логирования и восстановления после сбоев;
- добавить информацию о системных требованиях и зависимостях проекта.

После реализации предложенных улучшений продукт сможет полностью соответствовать техническому заданию и требованиям к качеству программного обеспечения.