**Техническое задание для проекта**

**«Telegram-бот "Memetique" для автоматизированной выдачи мемов по категориям с системой предотвращения повторений.»**

**1. Введение**

**1.1 Описание системы:**

Телеграм-бот предоставляет пользователям доступ к коллекции мемов, распределенных по тематическим категориям. Система позволяет выбирать категории через интерактивное меню, после чего выдает случайный мем с учетом истории просмотров для минимизации повторов. Администратор может расширять библиотеку контента путем добавления изображений в соответствующие папки без необходимости изменения кода. Бот реализует базовый механизм ротации контента и ведения истории показов для каждого пользователя.

**1.2 Цель проекта:** разработать функционального Telegram-бота для автоматизированной выдачи пользователям тематических мемов из предустановленных категорий.

**2. Требования к системе**

**2.1. Функциональные требования**

1. Выбор категорий: Бот должен предоставлять интерактивную клавиатуру с категориями: "Грустные", "Про работу", "Про учебу", "Глупые", "Котики", "Любые"
2. Отправка мемов: По запросу пользователя бот отправляет случайный мем из выбранной категории в формате JPEG/PNG/JPG
3. Система истории просмотров: Бот не должен повторять мемы в течение 5 последних запросов (глобально) и учитывать персональную историю пользователя
4. Обработка ошибок: При отсутствии мемов в категории бот выдает сообщение: "В этой категории пока нет мемов"

**2.2. Нефункциональные требования**

1. Время отклика ≤ 2 сек
2. Поддержка 100+ одновременных пользователей
3. Хранение истории просмотров в оперативной памяти
4. Кросс-платформенность (Windows/Linux/macOS)

**3. Описание системы**

**3.1. Основные компоненты**

|  |  |  |
| --- | --- | --- |
| **Компонент** | **Технология** | **Назначение** |
| Ядро бота | python-telegram-bot v20.0+ | Обработка команд и сообщений |
| Генератор случайных чисел | LCG-алгоритм | Рандомизация выдачи |
| Система истории | collections.deque | Учет просмотренных мемов |

**3.2. Схема работы**

1. Пользователь запускает бота командой /start
2. Бот отображает клавиатуру с категориями
3. При выборе категории:
   1. Система проверяет историю просмотров
   2. Выбирает подходящий мем
   3. Отправляет изображение с подписью
   4. Обновляет историю

**4. Требования к контенту**

**4.1. Структура папок**

Memetique – главная папка

Memes - папка с категориями мемов

Sad – грустные мемы

Work – про работу

Study – пор учебу

Stupid – глупые

Cats – котики

**4.2. Форматы файлов**

* Изображения: .jpg, .jpeg, .png

**5. Этапы разработки**

|  |  |  |
| --- | --- | --- |
| **Этап** | **Срок** | **Результат** |
| Анализ | 5 дн. | ТЗ, UML-диаграммы |
| Проектирование | 7 дн. | Архитектурная схема, ER-диаграмма |
| Реализация | 15 дн. | Исходный код (Python) |
| Тестирование | 5 дн. | Отчет с тест-кейсами |
| Документирование | 3 дн. | Составление README.md |

**6. Тестовые сценарии**

|  |  |
| --- | --- |
| Дейстивие | Ожидаемый результат |
| Отправка /start | Появление клавиатуры с категориями |
| Выбор категории "Котики" | Получение случайного мема из папки /cats |
| 5 запросов подряд | Отсутствие повторяющихся мемов |
| Запрос к пустой категории | Сообщение об отсутствии контента |
| Запросы от 5 разных пользователей одновременно | Отправка мемов без появления ошибок |

**7. Требования к документации**

1. **Техническая документация:**

* Описание архитектуры
* Схемы взаимодействия компонентов

1. **Пользовательская документация:**

* Инструкция по установке
* Руководство по добавлению мемов