Пояснительная записка к проекту "Приватные чаты в Telegram (без звонков) на AvaloniaUI + ASP.NET"

Введение

В современном мире мессенджеры играют ключевую роль в коммуникации. Telegram, благодаря своей безопасности и гибкости API, является одной из популярных платформ. Однако стандартный клиент не всегда удовлетворяет специфическим требованиям приватности и кастомизации.

Цель данного проекта – разработать упрощённый клиент для Telegram, поддерживающий только приватные чаты (без звонков, каналов и ботов) с использованием AvaloniaUI (кроссплатформенный UI-фреймворк) и ASP.NET (для backend-логики).

Глава 1. Элементы анализа литературы по теме

1.1 С чем работали

В ходе исследования были изучены:

- AvaloniaUI – кроссплатформенный фреймворк для создания GUI на .NET.

- ReactiveUI - это кроссплатформенный MVVM-фреймворк для .NET, использующий парадигму реактивного программирования.

- ASP.NET Core – для возможного backend-взаимодействия (если требуется серверная часть).

- SignalR - это абстракция некоторых видов транспорта, необходимых для работы в режиме реального времени между клиентом и сервером.

1.2 Кто ещё с этим работал

Аналогичные проекты:

- Telegram Desktop (официальный клиент) – использует Qt и C++.

- Nekogram – модифицированный клиент Telegram с открытым исходным кодом.

- Kotatogram – ещё один форк Telegram с дополнительными функциями.

Проблемы, которые решает данный проект:

- Упрощение интерфейса (только чаты).

- Улучшенная приватность (отсутствие лишних функций).

- Кроссплатформенность (благодаря AvaloniaUI).

1.3 Актуальность нашей работы

- Потребность в минималистичных мессенджерах растёт.

- Использование .NET и AvaloniaUI позволяет создать стабильное кроссплатформенное решение.

- Отсутствие звонков и каналов снижает нагрузку на клиент.

Глава 2. Методика создания проекта

Этапы разработки:

1. Проектирование архитектуры:

- Клиентская часть: AvaloniaUI (C#).

- Серверная часть ASP.NET Core

2. Реализация базового функционала:

- Авторизация

- Загрузка списка чатов.

- Отправка и получение сообщений.

3. Разработка UI:

- Список чатов.

- Окно переписки.

- Настройки.

4. Тестирование:

- Проверка работы на Windows.

- Тестирование стабильности соединения.

6. Оптимизация и доработка.

Инструменты:

- Язык: C# (.NET 9).

- UI: AvaloniaUI + ReactiveUI.

- Сервер: ASP.NET Core.

- База данных: Postgresql

Глава 3. Экспериментальная проверка

После реализации MVP проведено тестирование:

- Авторизация: успешный вход

- Загрузка чатов: отображение диалогов.

- Обмен сообщениями: отправка и получение текста.

- \*\*Производительность\*\*: низкая нагрузка на систему.

Заключени

Разработан упрощённый клиент Telegram с фокусом на приватные чаты. AvaloniaUI обеспечил кроссплатформенность, а .NET – стабильность работы. Проект может быть расширен дополнительными функциями (шифрование, темы).

Список литературы

1. Официальная документация Telegram API.

2. Документация AvaloniaUI.

3. GitHub: `WTelegramClient`.

4. Статьи по MTProto и клиентским реализациям.