

Федеральное государственное автономное образовательное учреждение
высшего образования
«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»
Факультет информационных технологий
Направление подготовки «Системная и программная инженерия»

ОТЧЁТ

по проектной практике

Студент: Трошкин Дмитрий Александрович, группа 241-327

Место прохождения практики: Московский политех

Отчет принят с оценкой _____ Дата _____

Руководитель практики: Баринова Наталья Владимировна

Москва 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	3
Общая информация о проекте	3
Описание задания по проектной практике	4
ПРАКТИЧЕСКАЯ ЧАСТЬ	5
Базовая часть.....	5
Написание документов в Markdown.....	6
Создание статического веб-сайта	6
Взаимодействие с организацией-партнёром	11
ВАРИАТИВНАЯ ЧАСТЬ	13
Серверная часть	14
Клиентская часть	14
Функциональные возможности	15
ЗАКЛЮЧЕНИЕ	19
СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ	21

ВВЕДЕНИЕ

Общая информация о проекте

Название проекта: «EasyAccess. Браузерное расширение для повышения веб-доступности.»

Актуальность: В современном мире веб-доступность играет критически важную роль в обеспечении равного доступа к информации для всех пользователей, включая людей с ограниченными возможностями здоровья. Однако многие веб-сайты не полностью соответствуют стандартам доступности, что создает барьеры для значительной части пользователей. Проект EasyAccess нацелен на решение этой проблемы путем создания браузерного расширения, позволяющего адаптировать содержимое веб-страниц под индивидуальные потребности пользователей.

Веб-доступность сегодня — это не только социальная ответственность, но и юридическое требование во многих странах. Согласно исследованиям WebAIM, более 96% из миллиона самых популярных веб-страниц имеют ошибки доступности, нарушающие руководящие принципы WCAG (Web Content Accessibility Guidelines). Это делает разработку инструментов, улучшающих доступность, крайне актуальной задачей.

Цели и задачи проекта:

1. Разработка браузерного расширения для адаптации веб-контента под различные потребности пользователей.
2. Создание системы пакетов модификаций с возможностью их обмена между пользователями.
3. Реализация функционала для настройки визуального отображения, изменения контрастности, размера шрифта и других параметров.
4. Интеграция возможностей голосового управления и экранного чтения.

5. Создание маркетплейса для обмена пакетами настроек между пользователями.

Описание задания по проектной практике

Задание на проектную (учебную) практику разработано для студентов первого курса, обучающихся по направлениям подготовки, связанным с информационными технологиями и информационной безопасностью. Трудоёмкость практики составляет 72 академических часа. Задание может выполняться индивидуально или в составе группы до 3 человек.

Задание состоит из двух частей:

- базовая часть (обязательная для всех студентов):
 - настройка Git и репозитория,
 - написание документов в Markdown,
 - создание статического веб-сайта,
 - взаимодействие с организацией-партнёром,
 - отчёт по практике;
- вариативная часть.

В моем случае для вариативной части была выбрана тема «Практическая реализация технологии» из списка, представленного в репозитории [codecrafters-io/build-your-own-x](https://github.com/codecrafters-io/build-your-own-x). Конкретно я реализовал чат-приложение с клиент-серверной архитектурой на языке C++ с использованием фреймворка TSP.

ПРАКТИЧЕСКАЯ ЧАСТЬ

Базовая часть

В рамках практики был создан репозиторий на GitHub на основе предоставленного шаблона. Для работы с Git использовались как консольный интерфейс GitBash, так и графический интерфейс GitHub Desktop.

Выполненные действия:

1. Создание форка репозитория mospol/practice-2025-1.
2. Клонирование репозитория на локальную машину командой `git clone`.
3. Создание структуры директорий согласно требованиям.
4. Работа с ветками для разработки вариативной части с помощью команд `git checkout`, `git merge`.
5. Регулярная фиксация изменений с осмысленными комментариями с помощью команд `git add`, `git commit`, `git push`.

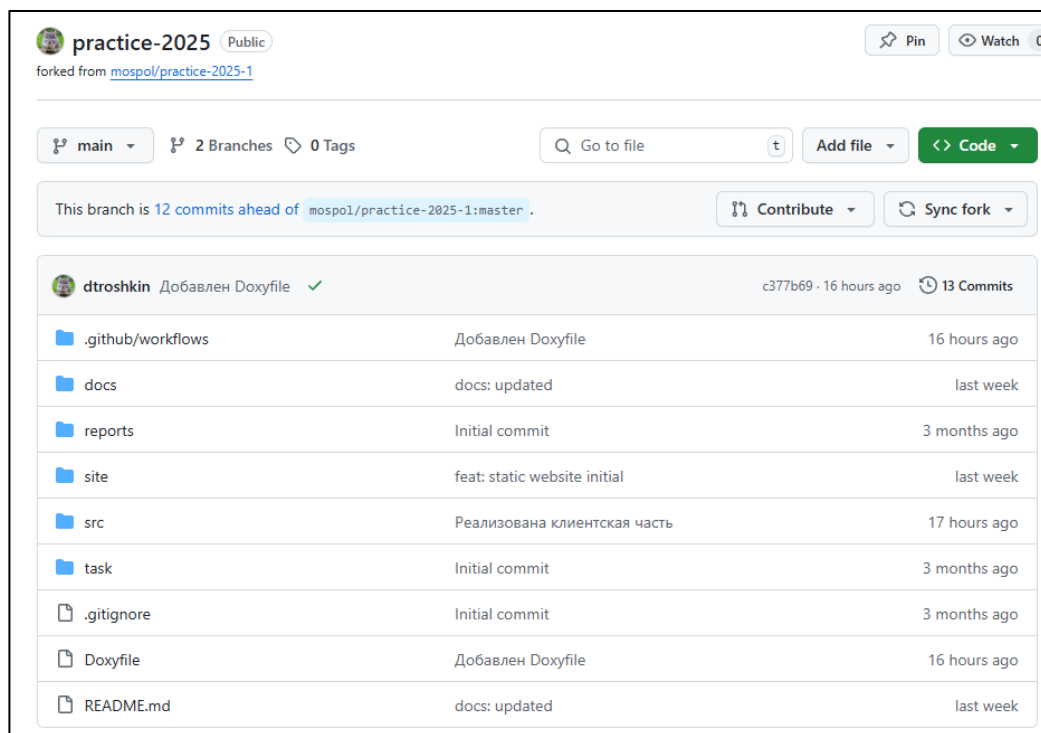


Рисунок 1. Репозиторий проектной практики

В процессе работы с репозиторием были освоены ключевые функции Git:

- Создание и переключение между ветками;
- Фиксация изменений с понятными описаниями;
- Разрешение конфликтов слияния;
- Работа с удаленным репозиторием;
- Использование .gitignore для исключения временных файлов.

Работа с Git позволила эффективно организовать командную работу и обеспечить контроль версий на протяжении всего проекта.

Написание документов в Markdown

Markdown был использован для создания документации проекта, включая:

- README.md в корне репозитория с основной информацией о проекте;
- документацию к модулям и компонентам в папке docs/;
- описание вариативной части в docs/individual_task.md;
- инструкции по установке и использованию.

Использование Markdown значительно упростило процесс документирования проекта, обеспечивая хорошую читаемость как в текстовом формате, так и в отрендеренном виде на GitHub.

Создание статического веб-сайта

В рамках базовой части задания был разработан статический веб-сайт, посвященный проекту "EasyAccess". Сайт создан с использованием HTML и CSS.

```
site/
├─ index.html      # Главная страница
├─ about.html      # О проекте
├─ team.html       # Участники
├─ journal.html    # Журнал
├─ resources.html  # Ресурсы
├─ assets/
│   └─ logo.svg    # Логотип
├─ styles/
│   ├── base.css   # Основные стили
│   ├── main.css   # Стили главной страницы
│   ├── about.css  # Стили страницы о проекте
│   ├── team.css   # Стили страницы участников
│   ├── journal.css # Стили журнала
│   ├── resources.css # Стили страницы ресурсов
│   └─ mockup.css  # Стили для демонстрации
└─ images/
    └─ demo.png    # Изображения
```

Рисунок 2. Структура сайта

Использованные технологии:

- HTML5 для структуры страниц;
- CSS3 для стилизации (включая Flexbox и Grid);
- CSS Variables для унификации стилей;
- шрифт Inter из Google Fonts;
- Phosphor Icons для иконок.

Особенности реализации:

- адаптивный дизайн для корректного отображения на мобильных устройствах;
- единая система дизайна (цвета, отступы, типографика);
- компонентный подход к CSS (переиспользуемые классы);
- интерактивные элементы с анимациями;
- семантическая верстка для лучшей доступности.

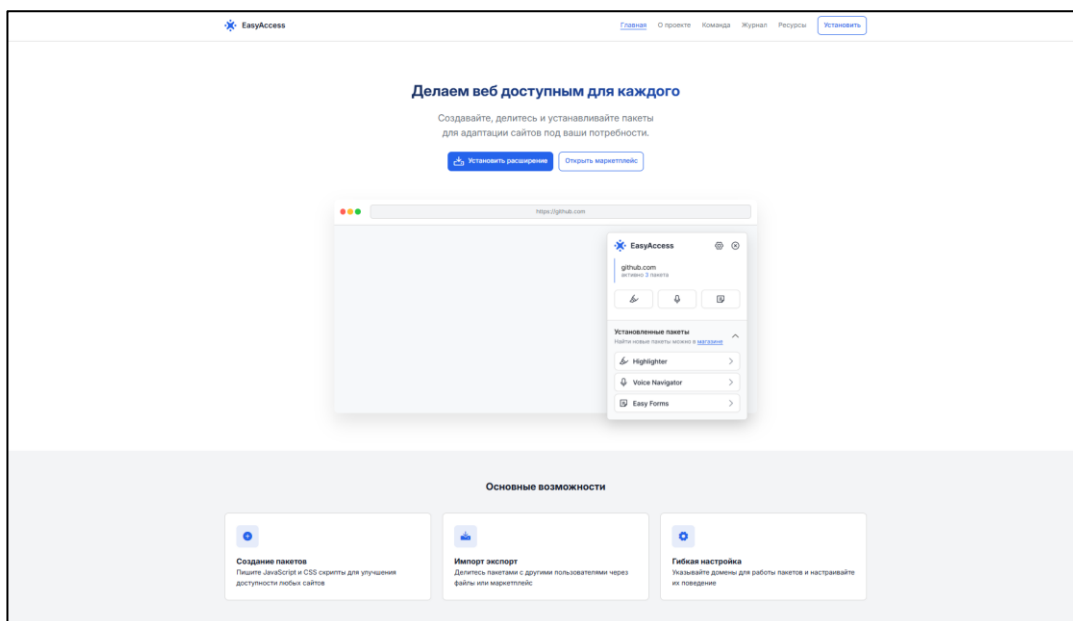


Рисунок 3. Главная страница

На главной странице представлена ключевая информация о проекте EasyAccess, описаны его цели и основной функционал. Дизайн выполнен с акцентом на основном сообщении проекта — «Делаем веб доступным для каждого».

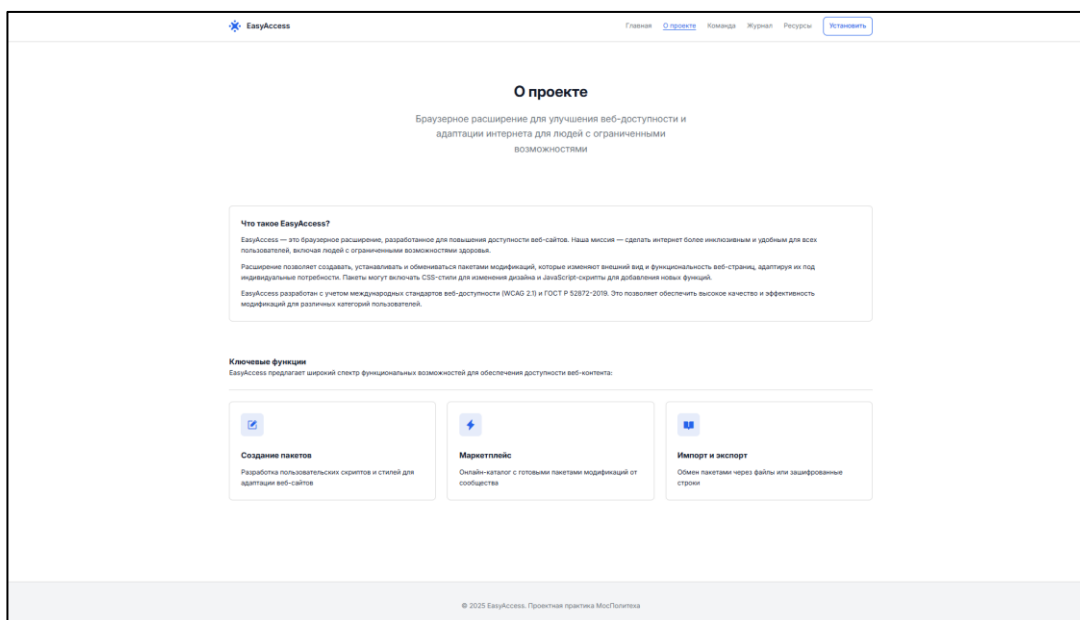


Рисунок 4. Страница «О проекте»

Страница «О проекте» содержит подробное описание проекта, его цели, задачи и технические аспекты реализации. Информация структурирована в виде блоков, что улучшает читаемость и восприятие.

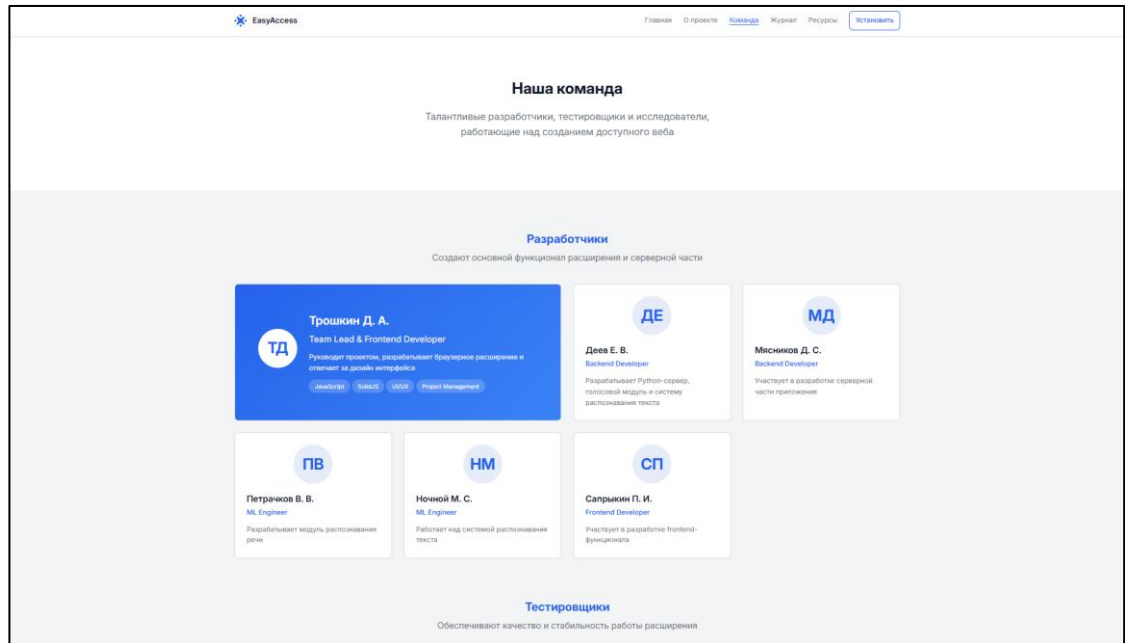


Рисунок 5. Страница «Наша команда»

На странице представлена информация о команде проекта, включая мою роль как тимлида и frontend-разработчика, а также информацию о других участниках.

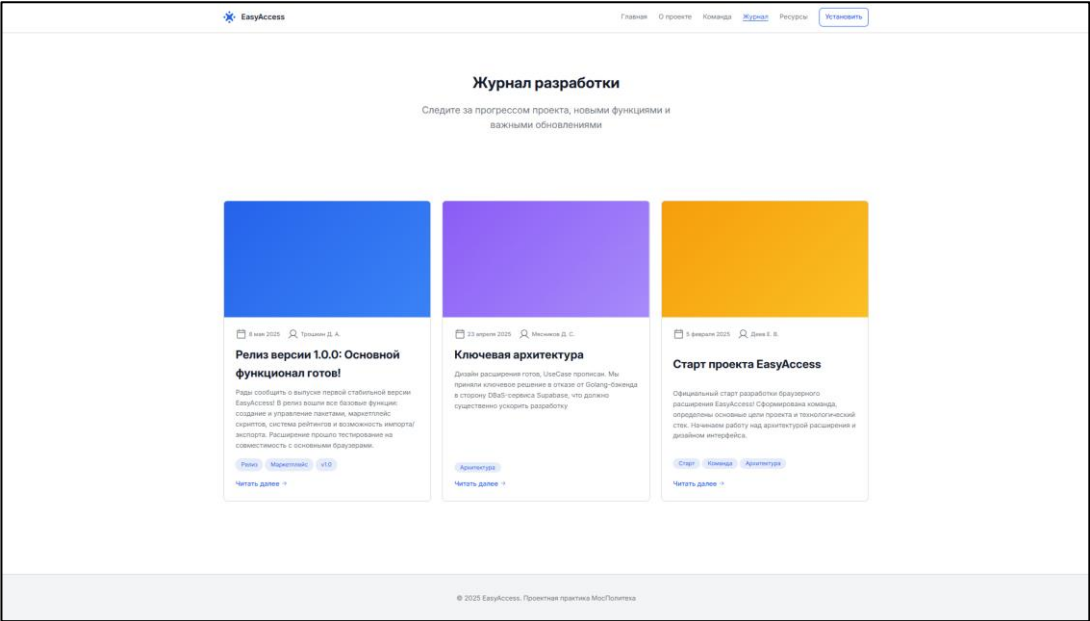


Рисунок 6. Страница «Журнал»

Журнал содержит новости и обновления о ходе разработки проекта, включая три ключевых записи:

- релиз версии 1.0.0 с основным функционалом;
- разработка ключевой архитектуры;
- официальный старт проекта.

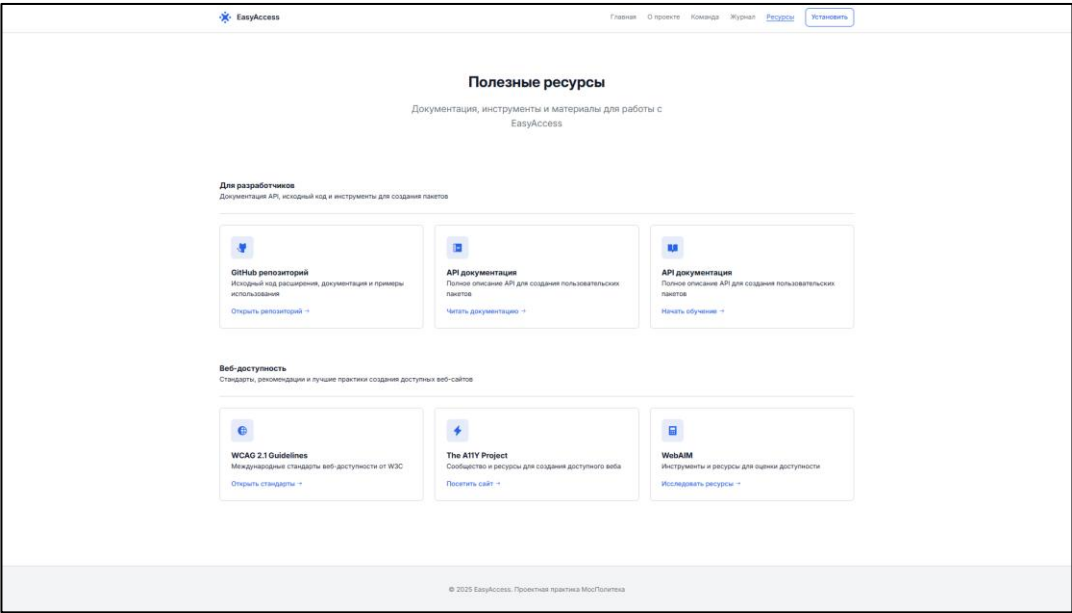


Рисунок 7. Страница «Ресурсы»

На странице представлены полезные ссылки на документацию, инструменты разработки и ресурсы по веб-доступности.

Все страницы имеют единый дизайн шапки и футера, что обеспечивает консистентность пользовательского опыта. Навигация интуитивно понятна и доступна с любой страницы сайта.

Взаимодействие с организацией-партнёром

В рамках взаимодействия с организацией-партнером я принял участие в мероприятии Y&&Y Lab, организованном компанией Яндекс.

Дата: 20 марта 2025 г.

Место проведения: БЦ Морозов, Москва

Организатор: Яндекс (Young&&Yandex)

Формат участия: Очное присутствие.

Описание мероприятия: Young&&Yandex Lab представляет собой масштабный карьерный фестиваль Яндекса, направленный на развитие молодых специалистов в IT-сфере.

Мероприятие включало различные треки по направлениям:

- Бэкенд;
- Фронтенд;
- Мобильная разработка;
- Data Science;
- DevOps;
- Нетехнические специальности.

Ключевые активности фестиваля:

1. Презентация проектов для молодёжи

Были представлены новые возможности для студентов и начинающих специалистов, включая:

- летние школы 2025,
- IT-стажировки в Яндексе,
- Business Camp для нетехнических направлений,

- тренажёр CodeRun для прокачивания навыков,
- Баттл вузов 2025;

2. Митап по фронтенд-разработке.

В рамках митапа выступил Алексей Ершков, руководитель фронтенда i18n Плюса Фантеха, с докладом «Что может пойти не так с простым компонентом в больших проектах?».

Основные темы доклада:

- Разработка универсального переключателя языков для нескольких сервисов Яндекса;
- Проблемы адаптации интерфейсов под различные языки;
- Аспекты доступности мультиязычных компонентов;
- Технические решения для унифицированного интерфейса.

Участие в этом мероприятии напрямую повлияло на подход к разработке нашего расширения EasyAccess:

Получение экспертизы в области доступности интерфейсов:

- Рассмотренный на митапе подход к созданию универсальных компонентов с учетом доступности дал ценные идеи для нашего проекта.
- Применение практик Яндекса в области мультиязычности.
- Расширение сетевых контактов – во время фестиваля вакансий и нетворкинга удалось установить контакты с разработчиками, имеющими опыт в создании доступных интерфейсов.

Полученные знания и навыки:

- Понимание подходов крупной IT-компании к проблемам доступности;
- Технические приемы для создания адаптивных и доступных компонентов интерфейса;
- Инсайты по потенциальным проблемам и их решениям при масштабировании простых компонентов.

ВАРИАТИВНАЯ ЧАСТЬ

В рамках вариативной части задания была выбрана тема "Build your own Chat App" из репозитория `codecrafters-io/build-your-own-x`. Реализовано полноценное чат-приложение с клиент-серверной архитектурой.

Технологический стек:

- Язык программирования: C++;
- GUI фреймворк: Qt 6.8;
- Сетевое взаимодействие: Qt Network (QTcpSocket, QTcpServer);
- База данных: SQLite (через интерфейс Qt SQL);
- Сборка проекта: CMake;
- Тестирование: Qt Test Framework;
- Документация: Doxygen.

```
src/
├── client/
│   ├── apiservice.cpp
│   ├── apiservice.h
│   ├── chatcontroller.cpp
│   ├── chatcontroller.h
│   ├── dialog.cpp
│   ├── dialog.h
│   ├── dialog.ui
│   ├── emojiconverter.cpp
│   ├── emojiconverter.h
│   ├── loginwindow.cpp
│   ├── loginwindow.h
│   ├── loginwindow.ui
│   ├── main.cpp
│   ├── messageformatter.cpp
│   ├── messageformatter.h
│   ├── testcontroller.cpp
│   └── testcontroller.h
├── server/
│   ├── CMakeLists.txt
│   ├── src/
│   │   ├── database/
│   │   │   ├── database.cpp
│   │   │   └── database.h
│   │   ├── main.cpp
│   │   └── server/
│   │       ├── command_handler.cpp
│   │       ├── command_handler.h
│   │       ├── server.cpp
│   │       └── server.h
│   ├── .gitignore
│   └── CMakeLists.txt
```

Рисунок 8. Структура приложения

Серверная часть

Архитектура серверной части представлена несколькими ключевыми компонентами. В основе лежит класс `Server`, который отвечает за прослушивание TCP-соединений, управление клиентскими сокетами и обработку базовых сетевых событий, таких как подключение и отключение клиентов.

Для работы с командами используется `CommandHandler`. Этот класс обеспечивает регистрацию и аутентификацию пользователей, управляет отправкой и получением сообщений, обрабатывает запросы истории общения и предоставляет информацию о пользователях, находящихся в сети.

Взаимодействие с хранилищем данных осуществляется через класс `Database`. Он представляет собой интерфейс для работы с базой данных, где хранятся учетные записи пользователей, сохраняется история всех сообщений и собираются данные для последующей аналитики.

Клиентская часть

Клиентская часть начинается с сетевого класса `ApiService`, обеспечивающего подключение к серверу, отправку и получение JSON-сообщений, а также обработку серверных ответов.

Бизнес-логика клиента реализована в классе `ChatController`. Он управляет соединением с сервером, отвечает за авторизацию пользователя и служит связующим звеном между пользовательским интерфейсом и сетевым уровнем.

Пользовательский интерфейс представлен классами `LoginWindow` и `Dialog`. Они отвечают за графическое взаимодействие с пользователем, отображение сообщений и списка активных участников, а также предоставляют формы для ввода и отправки сообщений.

Дополнительно в системе присутствуют вспомогательные классы: `MessageFormatter` для обработки сообщений с поддержкой Markdown-

подобного синтаксиса и EmojiConverter для преобразования текстовых эмодзи в соответствующие Unicode-символы.

Функциональные возможности

Сервер обеспечивает регистрацию новых пользователей, аутентификацию существующих, пересылку сообщений между клиентами, хранение истории переписки и отслеживание онлайн-статуса каждого пользователя.

Клиентская часть позволяет подключаться к серверу по IP-адресу и порту, проходить регистрацию и авторизацию, обмениваться сообщениями, просматривать историю переписки и список пользователей онлайн. Также клиент поддерживает форматирование текста с использованием Markdown-подобного синтаксиса и работу с эмодзи.



The image displays two JSON objects representing commands sent from a client to a server. Each object is enclosed in a light gray box with a 'json' label at the top left. The first object is a login command with fields for 'command', 'username', and 'password'. The second object is a send message command with fields for 'command' and 'message'.

```
json
{
  "command": "login",
  "username": "user123",
  "password": "password123"
}

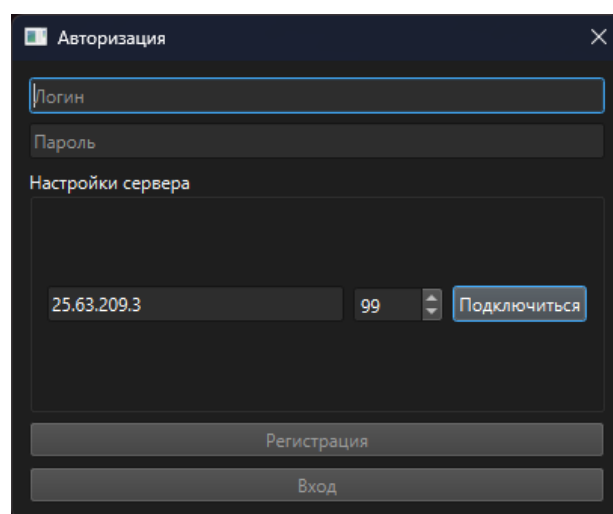
json
{
  "command": "send_message",
  "message": "Привет, мир!"
}
```

Рисунок 9. Команды от клиента к серверу

```
json
{
  "status": "ok",
  "message": "success login"
}

json
{
  "type": "message",
  "sender": "user123",
  "content": "Привет, мир!",
  "timestamp": "2025-05-10T15:30:00Z"
}
```

Рисунок 10. Ответы от сервера клиенту



Авторизация

Логин

Пароль

Настройки сервера

25.63.209.3 99 Подключиться

Регистрация

Вход

Рисунок 11. Окно авторизации в чат-приложении

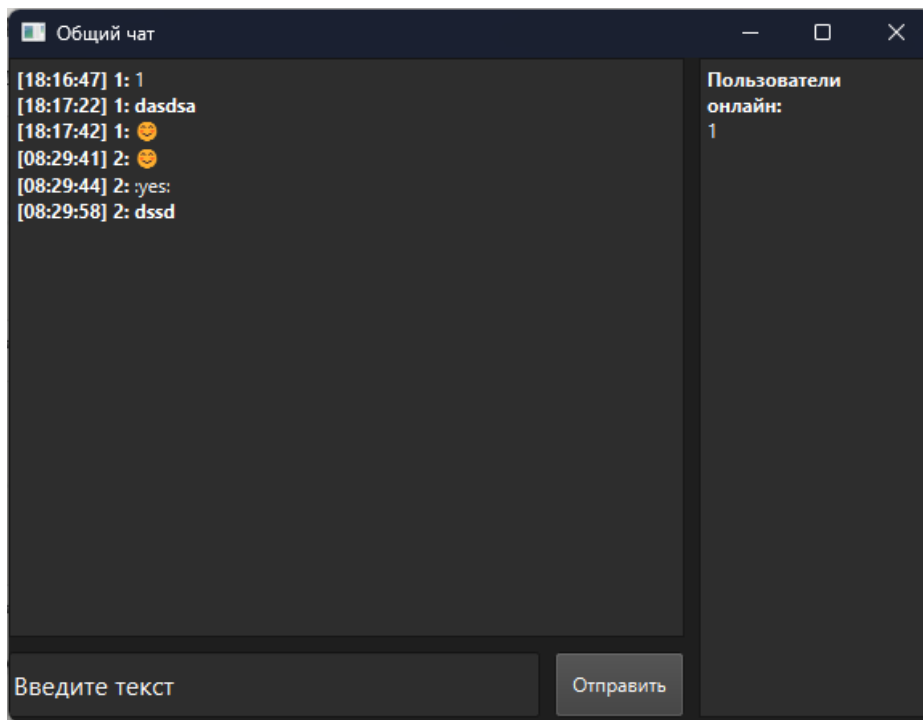


Рисунок 12. Основное окно чата

Реализованные улучшения:

1. Формат сообщений с Markdown-подобным синтаксисом:

- поддержка жирного текста с помощью двойных звездочек;
- поддержка курсивного текста с помощью одинарных звездочек;
- автоматическая замена текстовых эмодзи на Unicode-символы.

2. Архитектурные улучшения:

- внедрение паттерна «Model-View-Controller» для разделения логики;
- использование паттерна «Singleton» для сервисов соединения;
- использование паттерна «Command» для обработки сообщений на сервере.

3. Безопасность:

- проверка наличия повторных запусков клиента;
- защита от SQL-инъекций через параметризованные запросы.

ЗАКЛЮЧЕНИЕ

В рамках проектной практики были успешно выполнены все поставленные задачи как в базовой, так и в вариативной частях.

В базовой части:

- изучены и применены на практике технологии Git для контроля версий;
- освоен язык разметки Markdown для создания документации;
- разработан полноценный статический веб-сайт с использованием HTML и CSS;
- налажено взаимодействие с организацией-партнером через участие в мероприятии Y&&Y Lab от Яндекса.

В вариативной части:

- реализовано клиент-серверное чат-приложение на C++ с использованием Qt Framework;
- создана архитектура, основанная на современных паттернах проектирования;
- разработан функционал для регистрации, аутентификации и обмена сообщениями;
- реализованы дополнительные улучшения, такие как форматирование сообщений и поддержка эмодзи.

В проекте EasyAccess я выполнял роль тимлида и frontend-разработчика, отвечая за руководство командой, проектирование интерфейса и разработку основного функционала расширения.

Основные навыки, полученные в ходе практики:

- применение инструментов разработки и контроля версий;
- разработка пользовательских интерфейсов;
- проектирование архитектуры программных систем;
- командная работа и распределение задач;
- документирование проектов.

Эта практика стала ценным опытом, позволившим применить теоретические знания в реальных проектах и подготовиться к дальнейшей профессиональной деятельности в сфере разработки программного обеспечения.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Начало работы с GitHub Desktop [Электронный ресурс] // GitHub Docs. - URL: <https://docs.github.com/ru/desktop/overview/getting-started-with-github-desktop> (дата обращения: 10.04.2025).
2. CSS [Электронный ресурс] // Дока. - URL: <https://doka.guide/css/> (дата обращения: 20.03.2025).
3. HTML [Электронный ресурс] // Дока. - URL: <https://doka.guide/html/> (дата обращения: 20.03.2025).
4. Flexbox [Электронный ресурс] // Дока. - URL: <https://doka.guide/css/flexbox-guide/> (дата обращения: 25.03.2025).
5. Markdown [Электронный ресурс] // Дока. - URL: <https://doka.guide/tools/markdown/> (дата обращения: 20.03.2025).
6. Qt Documentation [Электронный ресурс] // Qt. - URL: <https://doc.qt.io/> (дата обращения: 15.04.2025).
7. Web Content Accessibility Guidelines (WCAG) 2.1 [Электронный ресурс] // W3C. - URL: <https://www.w3.org/TR/WCAG21/> (дата обращения: 05.04.2025).
8. Build Your Own X [Электронный ресурс] // GitHub. - URL: <https://github.com/codecrafters-io/build-your-own-x> (дата обращения: 12.04.2025).
9. Young&&Yandex LAB [Электронный ресурс] // Официальный сайт. - URL: <https://yandex.ru/yaintern/lab> (дата обращения: 20.03.2025).