Государственное бюджетное общеобразовательное учреждение
города Москва школа №924

Ομιλία

Выполнили ученики 10 «Б» класса,
Петров Н.М. и Гулиев Э.Ш. под
руководством Дарсавелидзе А. А.

Москва
2025

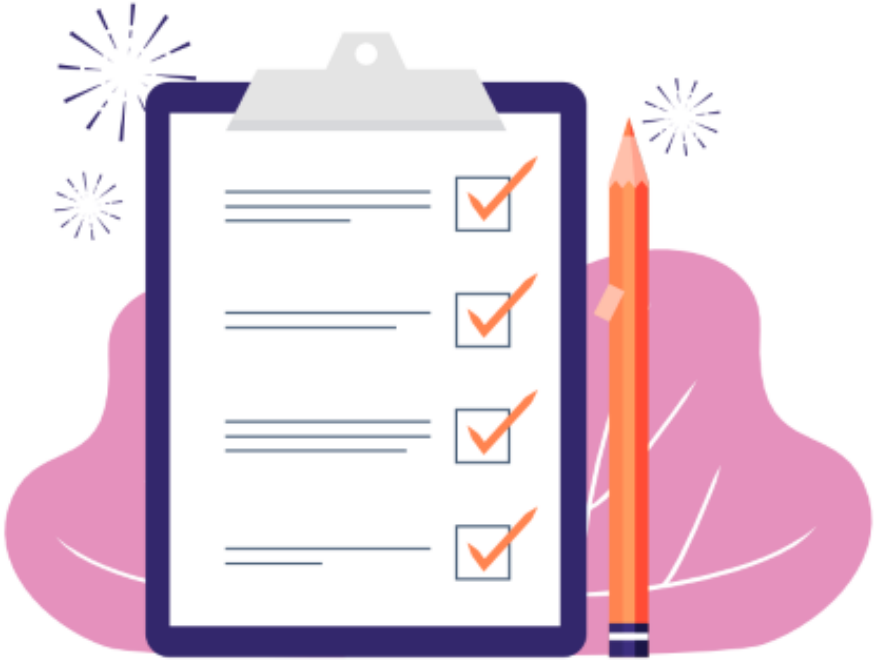
Постановка проблемы

Большинство мессенджеров требуют либо персональных данных, либо сложной модерации.

Нет единого решения с удобным открытым API и AI-модерацией.

Ομιλία (Omilia) нацелена на решение этих вопросов, предлагая анонимный чат-сервис с Flask-API и интеграцией в различные платформы (Go-сайт, Telegram-бот и т.д.).

Определение критериев результативности



Чтобы объективно оценить успех проекта, определены следующие критерии:

Безопасность: никакой утечки личных данных, корректная AI-модерация.

Масштабируемость: стабильная работа при увеличении нагрузки.

Удобство интеграции: доступное открытое API, понятная документация (Swagger).

Кроссплатформенность: синхронизация веб-приложения и Telegram-бота.

Удовлетворённость: положительные отзывы пользователей.

Создание концепции. Анализ ситуации. Прогнозирование последствий

Концепция: единая платформа Ομιλία (Omilia), где основа — Flask-API, а также Telegram-бот (PyTelegramBotAPI) и веб-сайт (Go).

Анализ ситуации показал, что разрозненные решения не дают продвинутой модерации и гибкого API одновременно.

Возможные последствия:

- *Плюсы*: безопасное общение, привлечение сторонних разработчиков.
- *Риски*: «ложные» блокировки AI, потребность в ресурсах при быстром росте.

Определение доступных ресурсов

Чтобы успешно реализовать проект, необходимо иметь следующие ресурсы:

Технические ресурсы:

сервер с установленным Python, Go, а также база данных (SQLite) и средства для AI-модерации.

Люди:

2 разработчика.

Время:

несколько месяцев.

Финансы:

аренда хостинга (VPS).

План выполнения проекта

1. Анализ требований.

2. Разработка Flask-API (основа проекта).

3. Разработка примеров клиентов:
   - веб-сайт на Go.
   - Telegram-бот.

3. Тестирование и оптимизация.

4. Запуск и поддержка.

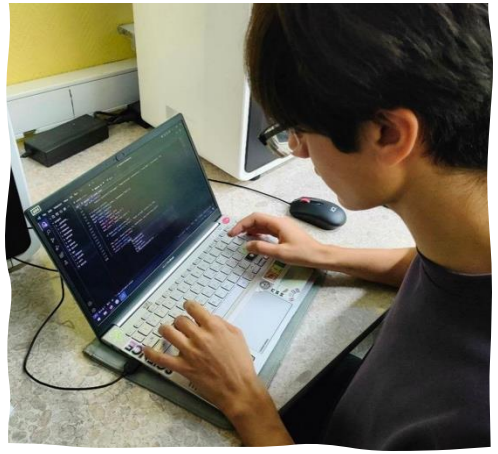
Реализация плана. Корректировка

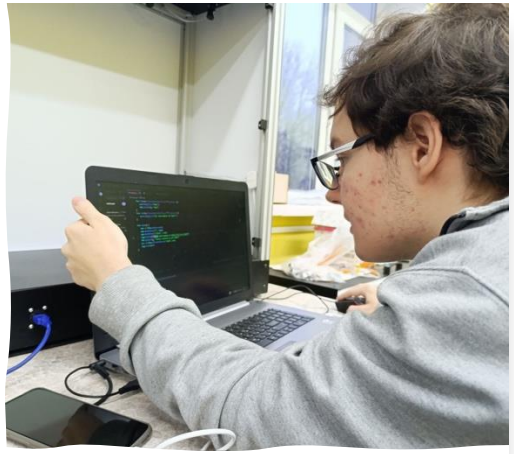
Перегрузка базы данных: внедрили Redis для уменьшения запросов к SQLite.

Валидация: использовали Marshmallow, чтобы снизить риск ошибок при приёме данных.

Модерация: интегрировали YOLOv5 (для изображений) и Hugging Face (для текстов).

В процессе: корректировали структуру API, чтобы упростить подключение новых платформ (в планах мобильная версия).

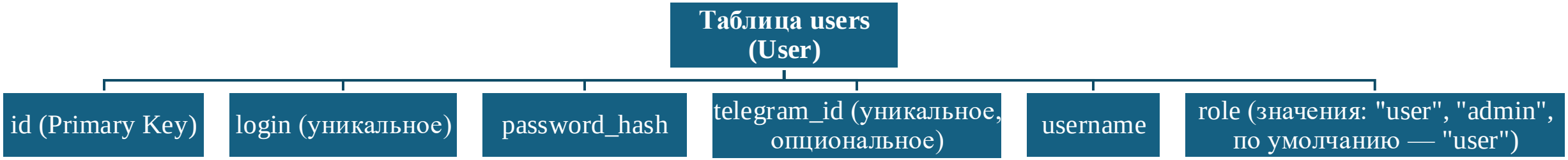




```python
if login_ and password:
    user = User.query.filter_by(login=login_).first()
    if not user:
        logger.info(f"User not found: {login_}")
        return jsonify({"error": "User not found"}), 404
    if not user.check_password(password):
        logger.info(f"Invalid password for user: {login_}")
        return jsonify({"error": "Invalid password"}), 401

    access_token = create_access_token(identity=user.id)
    refresh_token = create_refresh_token(identity=user.id)
    logger.info(f"User logged in: {login_}")
    return jsonify({
        "message": "Login successful",
        "access_token": access_token,
        "refresh_token": refresh_token
    }), 200
```

Схема базы данных



Redis-структуры

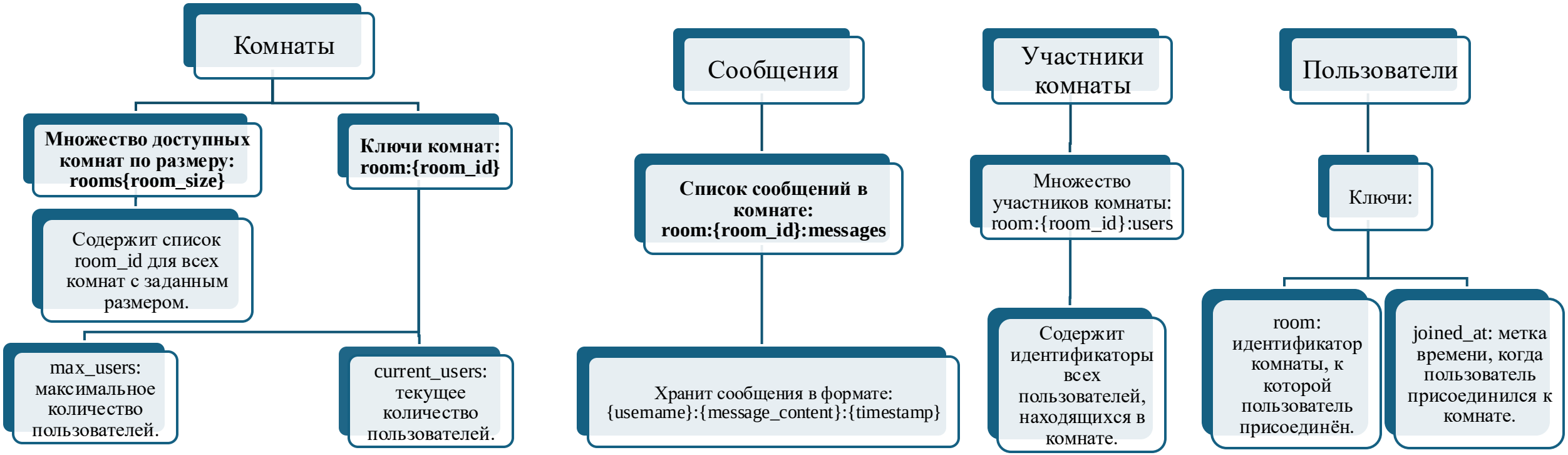
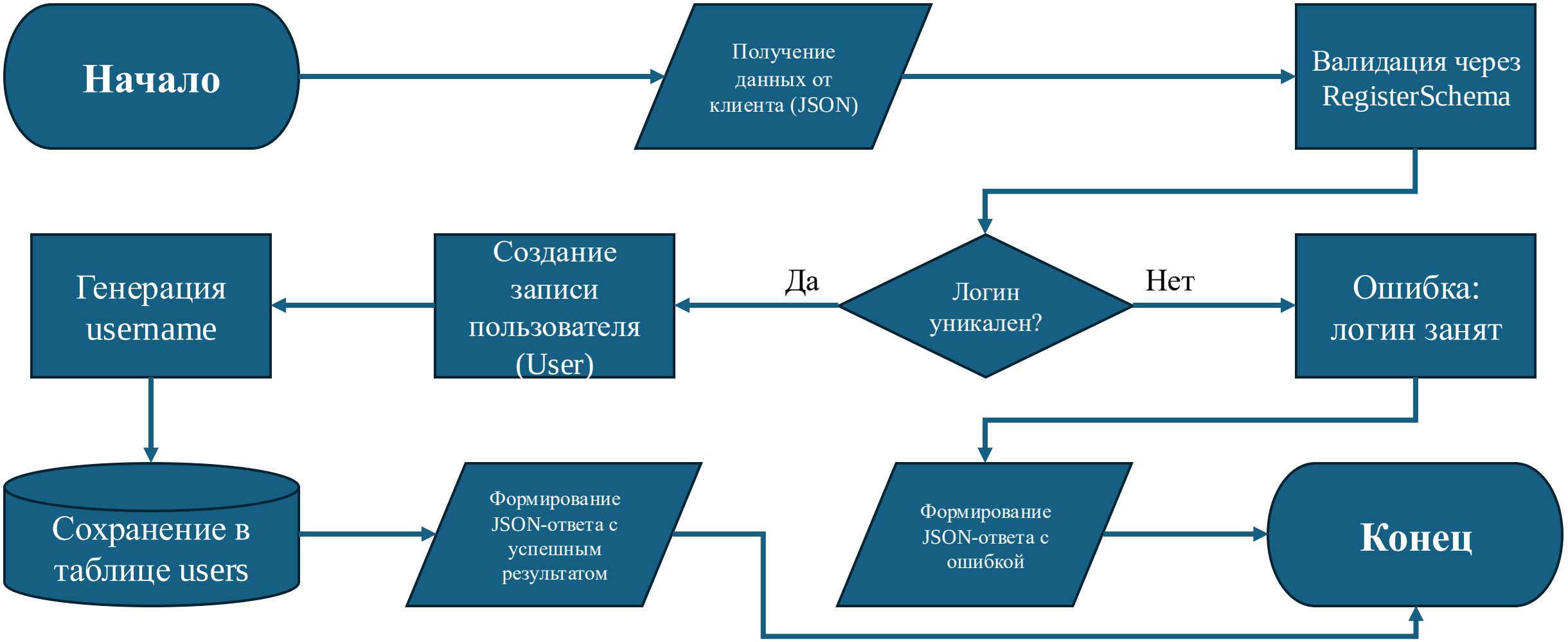


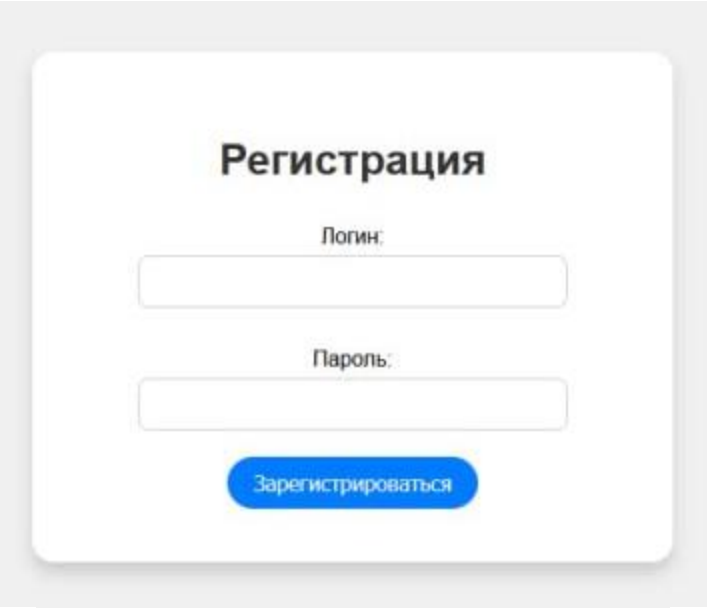
Схема слоёв приложения

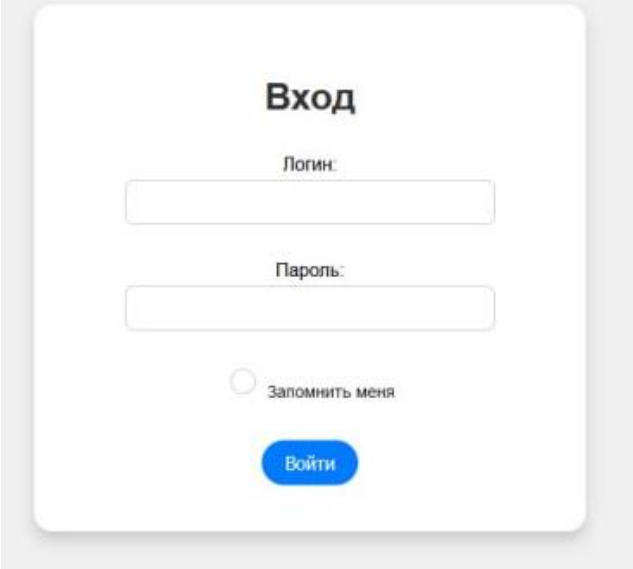
Контроллеры (Controllers)	Сервисы (Services)	Модели (Models)	Схемы (Schemas)	Хранилище (Storage)	Ядро (Core)
Отвечают только за: ▪ Получение HTTP-запросов от клиента. ▪ Валидацию данных запросов через Schemas. ▪ Вызов методов из Services. ▪ Формирование ответа клиенту (например, JSON).	**Функции сервисов**: ▪ Содержат всю бизнес-логику приложения. ▪ Взаимодействуют с базой данных через Models. ▪ Работают с временными данными через Redis. ▪ Подготавливают данные для передачи в контроллер.	Описывают структуру таблиц базы данных с использованием SQLAlchemy.	Валидация данных, поступающих в контроллеры.	**Redis**: ▪ Временное хранилище данных для комнат, используемое для быстрого доступа и управления временными данными. **SQLite**: ▪ Долгосрочное хранилище данных.	Включает инициализацию и управление системными компонентами: ▪ База данных. ▪ Redis. ▪ JWT. ▪ WebSocket.

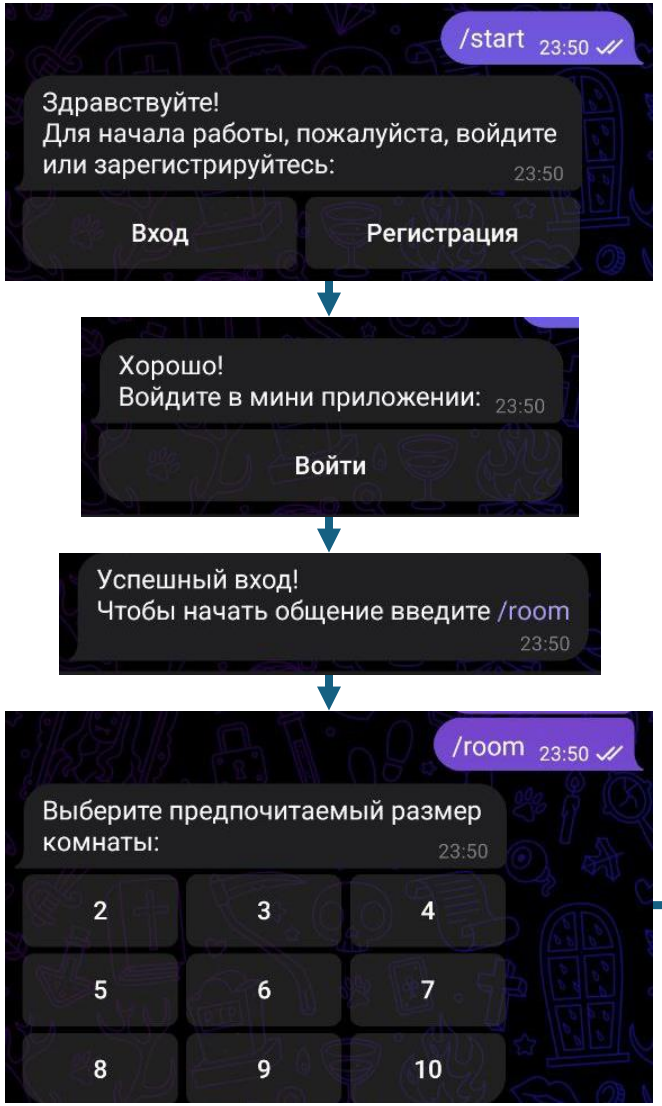
Пример регистрации пользователя (POST /register)

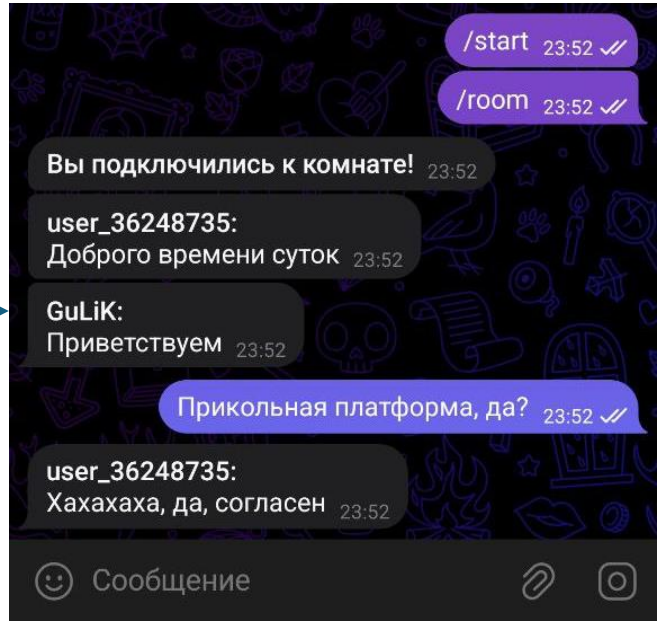


Демонстрация продукта









Оценка эффективности и результативности

Сопоставление с целью: реализованы гибкое API и безопасная среда общения.

Результативность: Redis повысил производительность, AI-модерация снизила токсичность.

Перспективы: расширение поддержки языков, возможно мобильное приложение.

Спасибо за внимание!

Использованные источники

1. Pallets. Flask: официальная документация [Электронный ресурс]. URL: https://flask.palletsprojects.com (дата обращения: 13.11.2024).
2. Flasgger. Flasgger: Easy OpenAPI specs and Swagger UI for your Flask API [Электронный ресурс]. URL: https://github.com/flasgger/flasgger (дата обращения: 04.12.2024).
3. NCNecros. Проектирование RESTful API с помощью Python и Flask [Электронный ресурс]. URL: https://habr.com/ru/articles/246699 (дата обращения: 22.11.2024).
4. Redis. Redis Documentation [Электронный ресурс]. URL: https://redis.io/docs (дата обращения: 20.12.2024).
5. Brad Solomon. **How to Use Redis With Python** [Электронный ресурс]. URL: https://realpython.com/python-redis (дата обращения: 20.12.2024).
6. Marshmallow. Marshmallow 3.25.1 documentation [Электронный ресурс]. URL: https://marshmallow.readthedocs.io (дата обращения: 13.12.2024).
7. Eternnoir. PyTelegramBotAPI: Python Telegram bot API [Электронный ресурс]. URL: https://github.com/eternnoir/pyTelegramBotAPI (дата обращения: 15.01.2025).
8. Real Python. Telegram Bot Python [Электронный ресурс]. URL: https://realpython.com/telegram-bot-python (дата обращения: 15.01.2025).
9. Hugging Face. Documentation [Электронный ресурс]. URL: https://huggingface.co/docs (дата обращения: 18.12.2024).
10. Ultralytics. YOLOv5: YOLOv5 in PyTorch > ONNX > CoreML > TFLite [Электронный ресурс]. URL: https://github.com/ultralytics/yolov5 (дата обращения: 25.12.2024)
11. Elliot Forbes. Creating A Simple Web Server With Golang [Электронный ресурс]. URL: https://tutorialedge.net/golang/creating-simple-web-server-with-golang (дата обращения: 10.01.2025).