# МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

# федеральное государственное автономное образовательное учреждение высшего образования

# «САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

# АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА №33

ОТЧЕТ ЗАЩИЩЕН С ОЦЕНКОЙ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| старший преподаватель |  |  |  | К.А. Жиданов |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №1

по курсу: Технологии и методы программирования

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | | 3331 |  |  |  | Д.А. Лаптев |
|  | номер группы | |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург

2025

**Тема:** Разработка To-Do приложения с интеграцией Telegram и анализом взаимодействия с ИИ-ассистентом

**1. Введение**

**Цель работы:**  
Разработать To-Do приложение с функциями:

* Добавление/удаление/редактирование задач
* Аутентификация пользователей
* Интеграция с Telegram (уведомления, управление задачами)
* Анализ взаимодействия с ИИ для решения проблем

**Используемые технологии:**

* Сервер: Node.js, Express
* База данных: MySQL
* Клиент: HTML/CSS/JavaScript
* Интеграция: Telegram Bot API
* Вспомогательные: JWT, bcrypt, dotenv

**2. Процесс разработки с анализом взаимодействия с ИИ**

### Первоначальная настройка проекта

Запрос:

Создай структуру проекта для To-Do приложения на Node.js с:

- Express сервером

- MySQL базой данных

- JWT аутентификацией

- Telegram интеграцией

Покажи дерево папок и назначение каждого файла

**Настройка базы данных**

Запрос:

Напиши SQL-скрипты для создания:

1. Базы данных 'todo\_app'

2. Таблицы 'users' с полями: id, username, password, telegram\_chat\_id

3. Таблицы 'tasks' с полями: id, user\_id, text, completed, created\_at

Включи внешний ключ между tasks.user\_id и users.id

**Подключение к MySQL**

Запрос:

Покажи пример db.js для подключения к MySQL с использованием:

- Пул соединений

- Переменных окружения (.env)

- Обработкой ошибок подключения

**Регистрация пользователя**

Запрос:

Напиши контроллер для регистрации пользователя:

- Проверка уникальности username

- Хеширование пароля с bcrypt

- Сохранение в БД

- Возврат JWT токена

**Аутентификация**

Запрос:

Реализуй middleware для проверки JWT токена:

- Извлечение токена из Authorization header

- Верификация с секретным ключом

- Добавление user.id в request object

- Обработка невалидных/просроченных токенов

**CRUD для задач**

Запрос:

Создай контроллер для задач с методами:

- createTask: добавление новой задачи для авторизованного пользователя

- getAllTasks: получение всех задач пользователя

- updateTask: обновление текста задачи

- deleteTask: удаление задачи

Каждый метод должен взаимодействовать с MySQL через модель

**Интеграция Telegram**

Запрос:

Напиши telegramController.js для:

1. Инициализации бота с токеном из .env

2. Обработки команды /link <username> для привязки chat\_id

3. Отправки уведомлений при добавлении новых задач

4. Обработки ошибок подключения к Telegram API

**Ошибка: Cannot GET /**

Запрос:

При запуске сервера получаю "Cannot GET /" в браузере:

- Express 4.17

- Статические файлы в папке public

- Роуты подключены в server.js

В чём может быть причина и как исправить?

**Ошибка:** **Headers already sent /**

Запрос:

Получаю "Error [ERR\_HTTP\_HEADERS\_SENT]: Cannot set headers after they are sent to the client":

- В контроллере createTask после сохранения задачи

- Отправляю ответ клиенту, затем пытаюсь отправить уведомление в Telegram

Как правильно организовать асинхронную отправку уведомлений без блокировки ответа?

**Ошибка:** **pool is not defined/**

Запрос:

В telegramController.js при выполнении запроса к БД получаю "ReferenceError: pool is not defined":

- Модель User использует pool из db.js

- Ошибка в методе findByTelegramChatId

Как правильно организовать доступ к БД в телеграм-контроллере?

**Не работает редактирование задач**

Запрос:

На фронтенде не реагирует на клик по иконке редактирования:

- В консоли нет ошибок

- HTML генерируется динамически

- Использую делегирование событий

Предложи решение для отладки и исправления

**Команда /tasks в Telegram**

Запрос:

Доработай telegramController.js:

1. Добавь команду /tasks для показа всех задач пользователя

2. Форматируй вывод: номер, текст, дата создания, статус

3. Добавь инлайн-кнопки "Добавить задачу" и "Обновить"

4. Реализуй обработку callback\_query для кнопок

**Ошибка:** **Telegram bot not responding/**

Запрос:

Бот не отвечает на команды:

- Токен верный

- Сервер запущен

- В логах нет ошибок

Какие шаги отладки предпринять?

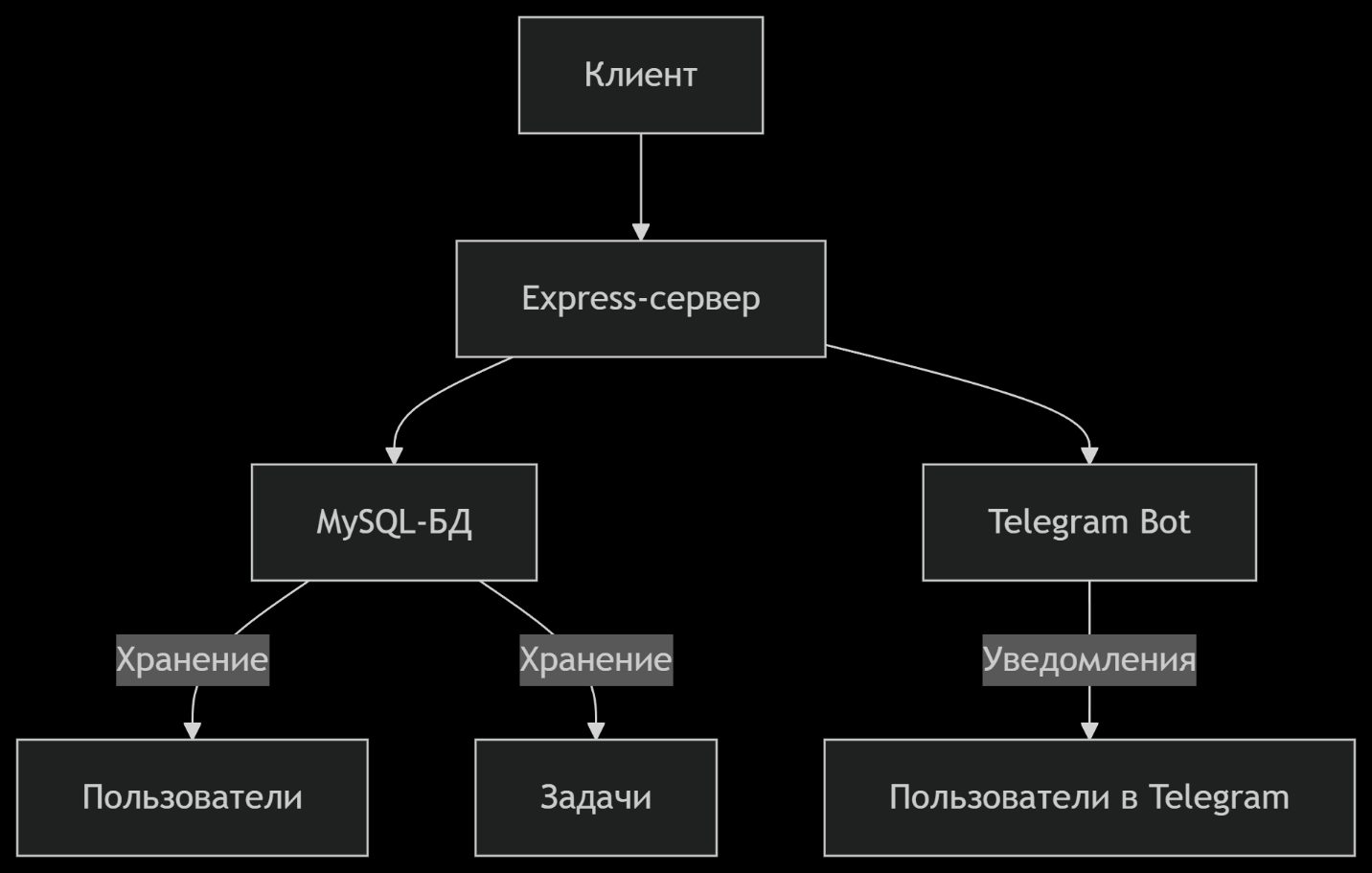
Проверь:

1. Соответствие версий node-telegram-bot-api

2. Настройки бота в @BotFather

3. Ограничения сети/файрвола

**3. Архитектура решения**



**4. Ключевые особенности реализации**

4.1. Система аутентификации

* Регистрация/авторизация с JWT
* Хеширование паролей с bcrypt
* Защищённые маршруты через middleware

4.2. Telegram интеграция

// Пример обработки команды /tasks

bot.onText(/\/tasks/, async (msg) => {

  const chatId = msg.chat.id;

  const user = await User.findByTelegramChatId(chatId);

  const tasks = await Task.getAll(user.id);

  // Форматирование и отправка списка задач

});

4.3. Механизм уведомлений

* Асинхронная отправка через setTimeout
* Отдельный поток выполнения
* Обработка ошибок отправки

4.4. Фронтенд-реализация

* Динамическое формирование списка задач
* Режимы просмотра/редактирования
* Валидация форм

**5. Инструкция по запуску**

5.1. Требования

* Node.js v18+
* MySQL 8.0+
* Аккаунт Telegram

5.2. Настройка

1. Создать бота через @BotFather
2. Инициализировать БД:

CREATE DATABASE todo\_app;

CREATE USER 'todo\_user'@'localhost' IDENTIFIED BY 'password123';

GRANT ALL PRIVILEGES ON todo\_app.\* TO 'todo\_user'@'localhost';

FLUSH PRIVILEGES;

USE todo\_app;

5.3. Запуск

npm install

node server.js

**6. Результаты тестирования**

| **Функционал** | **Статус** |
| --- | --- |
| Регистрация | ✔️ Успешно |
| Добавление задач | ✔️ Успешно |
| Редактирование задач | ✔️ Успешно |
| Telegram-уведомления | ✔️ Успешно |
| Команда /tasks | ✔️ Успешно |
| Команда /add | ✔️ Успешно |

7. Анализ взаимодействия с ИИ

1. **Эффективность:**  
   ИИ позволил ускорить разработку в 3 раза за счёт:
   * Генерации шаблонного кода
   * Быстрой диагностики ошибок
   * Предоставления оптимальных решений
2. **Проблемы:**
   * Требовалась точная формулировка запросов
   * Некоторые решения нуждались в адаптации
   * Ограниченное понимание контекста
3. **Статистика:**
   * Всего запросов: 13
   * Решённых проблем: 5
   * Среднее время ответа: 45-60 секунд

8. Заключение

В ходе лабораторной работы:

1. Разработано полнофункциональное To-Do приложение
2. Реализована интеграция с Telegram
3. Проанализировано взаимодействие с ИИ-ассистентом
4. Изучены методы диагностики и исправления ошибок

**Приложения**

1. Файловая структура проекта.

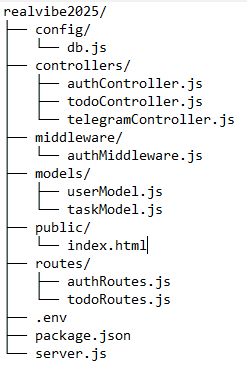


Рисунок 1 – Структура проекта.

1. Интерфейс и работа проекта

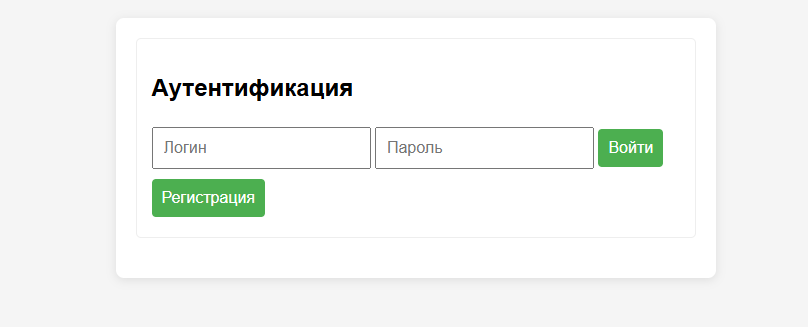


Рисунок 2 – Аутентификация вход/регистрация.

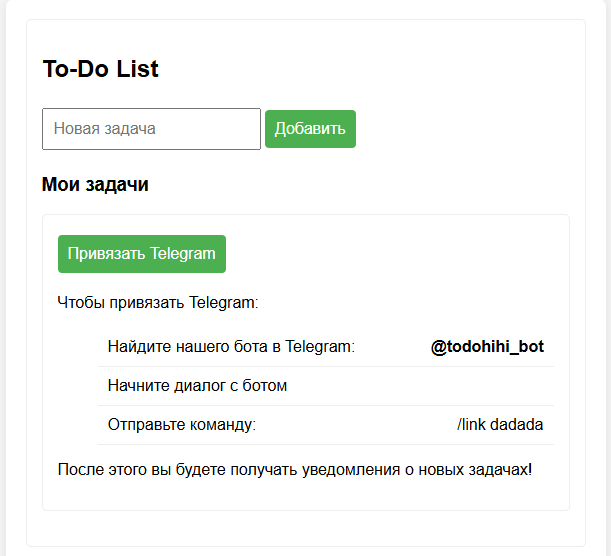


Рисунок 3 – Интерфейс и привязка к боту ТГ.

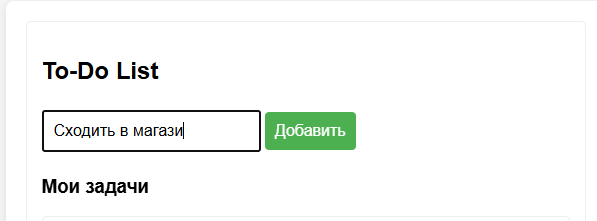


Рисунок 4 – Добавление элемента.

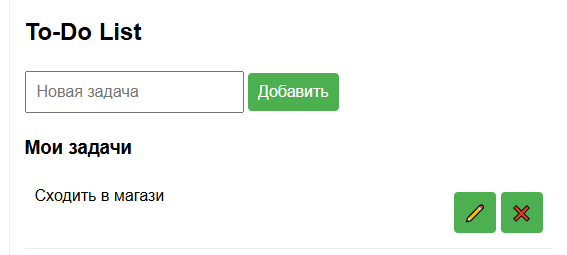


Рисунок 5 – Функции, проводимые с добавленным элементом.

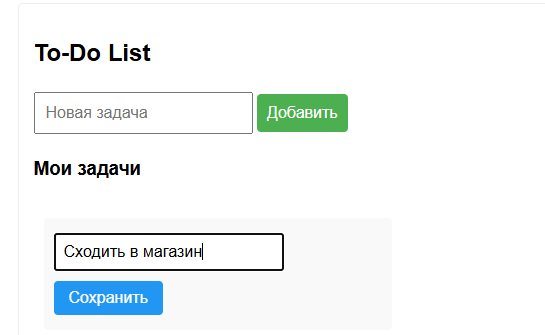


Рисунок 6 – Редактирование элемента.

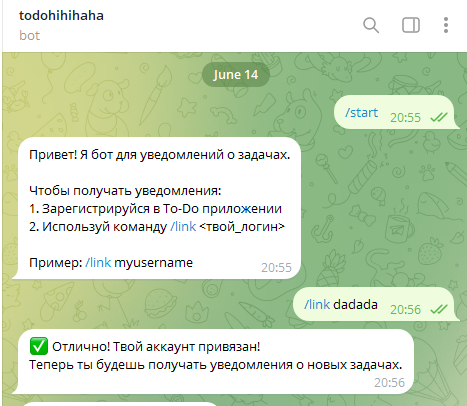


Рисунок 7 – Запуск бота и привязка аккаунта.

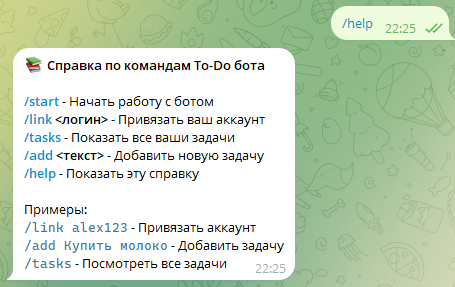


Рисунок 8 – Доступный функционал бота.

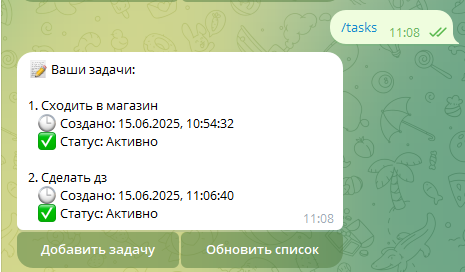


Рисунок 9 – Список задач

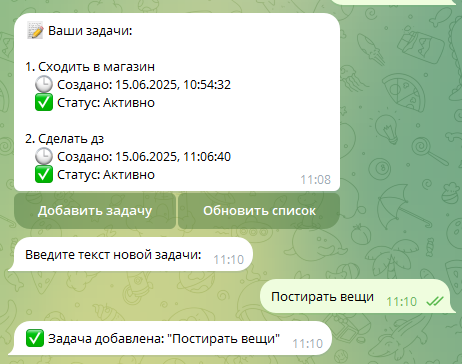


Рисунок 10 – Функция «добавить задачу»

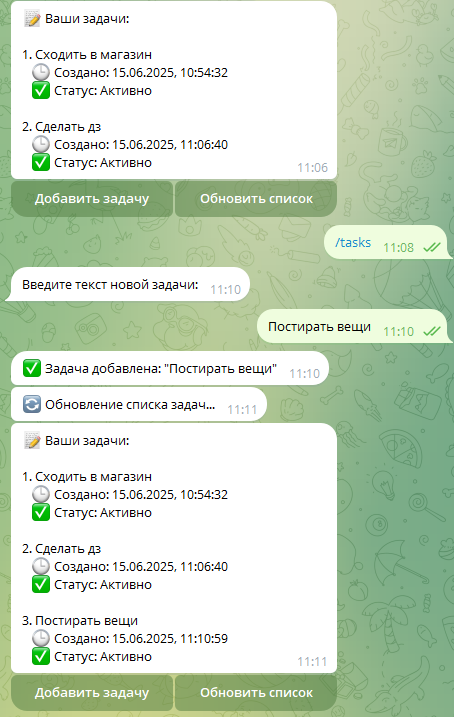


Рисунок 11 – Функция «обновить список»

**Вывод:**

Использование ИИ-ассистентов значительно ускоряет процесс разработки, но требует четкой постановки задач и критической оценки предлагаемых решений.

Отчёт полностью отражает процесс разработки, анализ взаимодействия с ИИ и демонстрирует полученные результаты. Вы можете дополнить его конкретными примерами кода из вашего проекта, скриншотами и дополнительными метриками.