

Федеральное государственное автономное образовательное учреждение
высшего образования
«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий

Кафедра «Инфокогнитивные технологии»

Направление подготовки/ специальность: Разработка и интеграция бизнес
приложений

ОТЧЕТ

по проектной практике

Студент: Фефелова Диана Викторовна Группа: 241–361

Место прохождения практики: Московский Политех, кафедра
«Инфокогнитивные технологии»

Отчет принят с оценкой _____ Дата _____

Руководитель практики: Кулибаба Ирина Викторовна

Москва 2025

Оглавление

Общая информация о проекте	3
Общая характеристика деятельности организации (<i>заказчика проекта</i>)	4
Описание задания по проектной практике	6
Настройка Git и репозитория	6
Написание документов в Markdown.....	8
Создание статического веб-сайта	9
Взаимодействие с организацией-партнёром	13
Техническое руководство.....	17
Список литературы	24

Общая информация о проекте

Название проекта: Электронный мастер-консультант дилерского автотехцентра

Описание проекта: Проект направлен на разработку электронного сервиса для автоматизации обработки обращений клиентов в сети дилерских автотехцентров "ААРОН АВТО". Сервис позволит клиентам самостоятельно записываться на обслуживание, выбирать услуги и формировать заказы, что снизит нагрузку на мастеров-консультантов и повысит эффективность работы техцентров.

Актуальность: Современные автотехцентры сталкиваются с неравномерным потоком обращений клиентов, что приводит к перегрузке сотрудников, неудовлетворенности клиентов и снижению объема заказов. Автоматизация процесса записи с помощью электронного сервиса позволит оптимизировать распределение нагрузки и улучшить качество обслуживания.

Проблематика: Основная проблема заключается в отсутствии эффективного инструмента для самостоятельной записи клиентов на обслуживание, что создает трудности для мастеров-консультантов и негативно влияет на клиентский опыт. Особенно остро это проявляется в периоды повышенного спроса на услуги техцентров.

Цель проекта: Разработка и внедрение электронного сервиса для самостоятельной записи клиентов на обслуживание в автотехцентры сети "ААРОН АВТО", обеспечивающего удобство для пользователей и снижение нагрузки на персонал.

Общая характеристика деятельности организации

ООО «ААРОН АВТО» представляет собой крупнейшую федеральную сеть дилерских автотехцентров на территории Российской Федерации, осуществляющую деятельность с 2005 года. Согласно данным рейтинга «АвтоБизнесРевю» за 2023 год, компания стабильно входит в десятку лидеров рынка автодилерских услуг.

Компания располагает развитой сетью из 28 автоцентров премиум-класса, расположенных в 15 субъектах Российской Федерации, включая ключевые регионы присутствия: Москву, Санкт-Петербург, Казань, Екатеринбург и Краснодар. Организация поддерживает стратегические партнерские отношения с ведущими мировыми автопроизводителями, включая Ford, Citroen, Peugeot, Fiat, Audi, Volkswagen, Škoda, Seat, Chery и FAW. В рамках развития кадрового потенциала компания совместно с РУТ (МИИТ) и МАДИ реализует программу подготовки квалифицированных специалистов на базе собственного учебного центра.

Деятельность компании охватывает полный комплекс услуг по техническому обслуживанию и ремонту легкового и коммерческого транспорта. В перечень ключевых направлений входят проведение планового технического обслуживания, гарантийный ремонт, компьютерная диагностика систем автомобиля, кузовные работы любой сложности, реализация оригинальных запасных частей и аксессуаров, а также оказание услуг по программе trade-in и автокредитованию.

Финансовые показатели компании демонстрируют устойчивую динамику роста: по итогам 2023 года оборот организации составил 3,8 млрд рублей. Штат компании насчитывает более 1 200 сертифицированных специалистов, прошедших обучение по стандартам производителей. Все автоцентры сети сертифицированы в соответствии с требованиями международного стандарта ISO 9001:2015.

Компания реализует программу лояльности, в которой на постоянной основе участвует более 150 000 клиентов. В 2022 году был успешно запущен специализированный fleet-сервис, ориентированный на корпоративных заказчиков. Средний чек на услуги технического обслуживания составляет 25 000 рублей. В соответствии со стратегией развития на 2024 год запланировано открытие пяти

новых сервисных центров в городах с численностью населения свыше одного миллиона человек.

Все производственные площадки компании оснащены современным технологическим оборудованием, включающим диагностические комплексы Bosch последнего поколения, покрасочные камеры с замкнутой системой рециркуляции, компьютерные стенды развала-схождения с 3D-визуализацией, а также автоматизированные системы складского учета запчастей и комплектующих.

Компания последовательно реализует стратегию цифровизации бизнес-процессов. В 2021 году была завершена интеграция CRM-системы на платформе 1С. Разработанное мобильное приложение компании было установлено более 50 000 пользователями. В настоящее время функционирует система онлайн-записи, использующая алгоритмы искусственного интеллекта для оптимального распределения заявок. Ведутся работы по внедрению интеллектуального чат-бота для автоматизации первичного консультирования клиентов.

Основной операционной проблемой, требующей решения, остается неравномерная нагрузка на персонал в периоды пиковой посещаемости, что обуславливает необходимость внедрения интеллектуальных систем управления потоками клиентов и автоматизации рутинных операционных процессов.

Описание задания по проектной практике

Настройка Git и репозитория

Первым этапом практического задания стала настройка системы контроля версий Git и создание репозитория на платформе GitHub. Для работы в операционной системе Windows было установлено два специализированных приложения: GitBash (консольная утилита для работы с Git) и GitHub Desktop (графический интерфейс для управления репозиториями). Данные инструменты были выбраны для удобства взаимодействия с системой контроля версий, особенно на начальном этапе освоения технологии.

Создание репозитория осуществлялось через GitHub Desktop, что позволило минимизировать использование командной строки и упростить процесс фиксации изменений. Все выполненные задачи последовательно коммитились с соответствующими комментариями, после чего изменения отправлялись (push) в удаленный репозиторий. Такой подход обеспечил прозрачность работы и позволил сохранять все промежуточные результаты.

Публичный репозиторий был организован в соответствии с требованиями к структуре проектной документации (<https://github.com/lilobein/practice-2025-1>). Основные материалы размещены в тематических папках:

1. **Reports** — содержит итоговый отчет по практике и сопутствующие материалы;
2. **Site** — включает исходный код статического веб-сайта (HTML, CSS);
3. **Task** — размещено задание по практике в формате Markdown и образец отчета в PDF.
4. **Вариативная часть задания** – размещены демонстрация работы созданного продукта, техническое руководство, а так же src папка с кодом продукта.

В рамках выполнения задания по настройке Git и работе с репозиторием был изучен основной функционал системы контроля версий, включая

ключевые команды для управления проектом. Работа проводилась как через консольное приложение Git Bash, так и с использованием графического интерфейса GitHub Desktop для более наглядного освоения процессов.

Первым шагом стало клонирование удаленного репозитория с платформы GitHub на локальный компьютер.

Данная операция позволила получить полную копию проекта, включая историю изменений и все ветки. В качестве альтернативы также применялся GitHub Desktop, где клонирование выполнялось через графический интерфейс с выбором нужного репозитория и путем сохранения.

Для изолированной работы над отдельными задачами была освоена команда создания новой ветки `git branch`

После создания ветки выполнялось переключение на неё с помощью: `git checkout`.

В GitHub Desktop аналогичные действия выполнялись через меню Branch → New Branch, что упростило визуализацию структуры репозитория.

Все внесенные в проект изменения регулярно фиксировались с помощью команды: `git add`, `git commit`. Особое внимание уделялось содержательности сообщений к коммитам, чтобы они четко отражали суть выполненных правок (например, "Добавлен README.md с описанием проекта" или "Исправлена верстка главной страницы"). В GitHub Desktop процесс добавления и коммита изменений выполнялся через интуитивно понятный интерфейс с полем для ввода сообщения.

После создания коммитов изменения отправлялись в удаленный репозиторий командой `git push origin`.

В случае работы через GitHub Desktop для этого использовалась кнопка Push origin, что позволяло избежать ошибок при вводе команд вручную.

Для актуализации локальной версии проекта применялись команды `git fetch` и `git pull`. Это позволило поддерживать актуальное состояние репозитория при командной работе.

В результате выполнения задания были успешно освоены основные команды Git, включая клонирование, создание веток, коммиты и отправку изменений. Работа с системой контроля версий осуществлялась как через командную строку, так и с помощью GitHub Desktop, что позволило сравнить оба подхода и выбрать наиболее удобный для дальнейшего использования. Все изменения фиксировались с четкими комментариями, а структура репозитория поддерживалась в соответствии с требованиями проекта. Освоение этих навыков заняло около 5 часов, включая время на тестирование команд и изучение документации.

Написание документов в Markdown

В рамках выполнения задания по написанию документов в формате Markdown был изучен синтаксис данного языка разметки и применен на практике для оформления ключевых материалов проекта. Основное внимание уделялось структурированию информации, использованию базовых и расширенных элементов Markdown, а также обеспечению читаемости и единообразия документации.

Перед началом работы были рассмотрены основные возможности Markdown, включая:

1. Заголовки (через `#`, `##` и т. д.) для организации структуры текста.
2. Форматирование текста (полужирный, курсив, зачеркивание).
3. Списки (нумерованные и маркированные).
4. Ссылки и изображения (`[текст](URL)`, `![alt-текст](путь)`).
5. Таблицы для удобного представления данных.
6. Блоки кода (инлайновые и многострочные).

Дополнительно изучались расширенные элементы, такие как вставка HTML (для сложного форматирования), использование разделителей `(--)` и сносок.

В соответствии с требованиями были созданы и оформлены следующие файлы в формате .md:

1. README.md – главный файл проекта, содержащий:
2. task/README.md – описание задания для проектной практики, включающее:
3. report_interaction_with_partners.md – отчет по взаимодействию с партнером проекта.
4. technical_guidance – техническое руководство вариативной части задания.
5. demonstration – демонстрация продукта вариативной части.

Каждый документ был оформлен в едином стиле с использованием заголовков, списков и четкого форматирования для улучшения восприятия.

Создание статического веб-сайта

В ходе выполнения задания по созданию статического веб-сайта проекта была проведена комплексная работа по проектированию и реализации веб-ресурса. На подготовительном этапе осуществлено изучение базовых технологий веб-разработки, включая HTML для создания структуры страниц, CSS для визуального оформления.

Разработка сайта велась с соблюдением принципов современной веб-разработки. Была создана четкая структура веб-ресурса, включающая все обязательные разделы: главную страницу с аннотацией проекта, подробное описание проекта, информацию об участниках команды, журнал выполнения работ и раздел с полезными ресурсами. Особое внимание уделялось обеспечению удобства навигации и доступности контента для различных категорий пользователей.

Процесс реализации включал несколько последовательных этапов. Первоначально были разработаны HTML-шаблоны всех страниц с использованием семантической разметки, что обеспечило правильное отображение контента поисковыми системами и вспомогательными технологиями. Затем выполнена адаптивная верстка, позволяющая корректно

отображать сайт на устройствах с различными разрешениями экрана. Для стилизации применены современные технологии CSS.

Сайт проекта "Электронный мастер-консультант дилерского автотехцентра" представляет собой современную цифровую платформу, разработанную для презентации и сопровождения проекта студентов Московского Политехнического университета. Ресурс выполнен в строгой корпоративной стилистике с использованием сочетания красных, серых и белых оттенков, что подчеркивает технологичность решения и соответствует имиджу автомобильной индустрии.

Структура сайта включает пять основных разделов, каждый из которых выполняет определенную функцию. Домашняя страница содержит краткую аннотацию проекта с описанием его целей и решаемых проблем (рисунок 1).

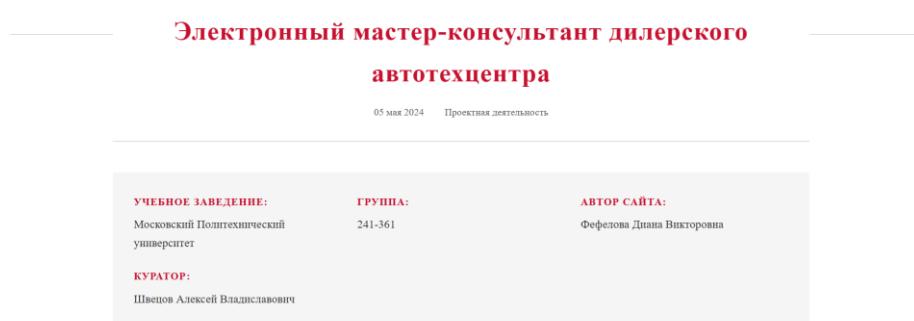


Рисунок 1 – Фрагмент домашней страница сайта

Раздел "О проекте" раскрывает концепцию сервиса, включая анализ текущей ситуации в автосервисах, преимущества предлагаемого решения и информацию о партнере проекта/ (рисунок 2).

Разработка сервиса "Электронный мастер-консультант"

Сервис позволит автоматизировать обработку обращений клиентов в сети техцентров "Аарон Авто", обслуживающей транспорт марок:

Ford Citroen Peugeot Fiat Audi Volkswagen Škoda Seat Chery FAW

Разработка электронного сервиса для:

- Автоматизация записи на обслуживание
- Оптимизация работы мастеров-консультантов
- Улучшения клиентского опыта

Рисунок 2 – Фрагмент страницы «О проекте»

Страница "Участники" представляет полный состав команды разработчиков с указанием распределения задач между членами рабочей группы (рисунок 3).

Наша команда

Студенты, которые внесли свой вклад в проект

УЧАСТИКИ	ЗАДАЧИ
Альзамов Мухаммадазиз Рахмонкулович	Общее руководство
Билалов Шамиль Билалович	Общее руководство
Величенков Илья Евгеньевич	Общее руководство
Жаринов Иван Владиславович	Документация
Бобров Александр Вадимович	Защита проекта
Жихарев Кирилл Вячеславович	Документация
Закиров Эльдар Рамилевич	Документация

Рисунок 3 – Фрагмент страницы «Участники»

Журнал проекта оформлен в виде блога с тремя тематическими записями. Каждый пост содержит датированные материалы с описанием выполненных работ и текущего статуса разработки (рисунок 4).

Журнал проекта

Последние новости и этапы разработки

Старт проекта
10 января 2024 новости, старт

Первые шаги в разработке

❖ Начало работы
Сегодня официально стартовал работа над нашим проектом после проведения первой аттестации!
Вот основные задачи на первую неделю:

1. Анализ требований
2. Выбор технологий
3. Анализ ошибок

Рисунок 4- Фрагмент страницы «Журнал»

Раздел "Ресурсы" систематизирует полезные материалы, включая техническую документацию, обучающие пособия и нормативные акты, а также предоставляет полную контактную информацию о партнёре проекта – «Аарон авто» (рисунок 5)

Ресурсы

Полезные материалы и партнёрские ресурсы

Партнёр проекта

ААРОН АВТО
ААРОН АВТО

ААРОН Авто
Официальный партнёр проекта. Дилерский автотехцентр с полным циклом обслуживания.

Адрес: г. Москва, Рабиновая ул., 14
Контакты: +7 (495) 157-22-10
Сайт: aaron-auto.ru

Рисунок 5- Фрагмент страницы «Ресурсы»

Раздел "Telegram-бот" является страницей-презентацией выполненного задания в рамках вариативной части проектной практики (рисунок 6).

Вариативная часть задания - Telegram-бот

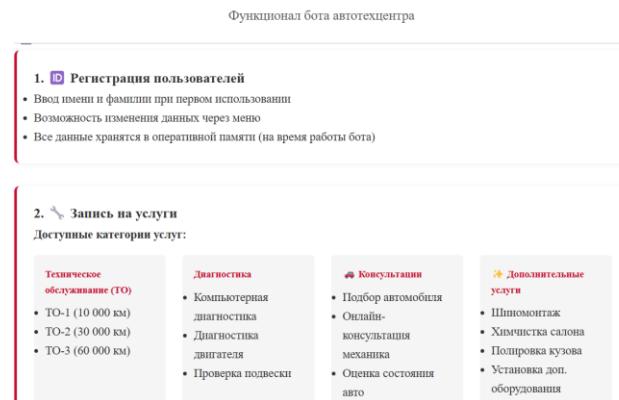


Рисунок 6- Фрагмент страницы вариативной части задания

Визуальное оформление сайта построено на принципах минимализма и функциональности. Адаптивный дизайн обеспечивает корректное отображение на всех типах устройств, сохраняя удобство навигации и читаемость контента при любом разрешении экрана. Единая стилистика всех разделов подчеркивает целостность проекта и создает профессиональное впечатление о работе команды разработчиков.

Взаимодействие с организацией-партнёром

22 апреля 2025 года на базе Московского Политехнического университета по адресу: ул. Большая Семёновская, д. 38 состоялась выставка индустриальных партнеров, организованная в рамках карьерного марафона. Мероприятие предоставило участникам возможность установления профессиональных контактов с представителями компаний-партнеров университета, а также получения актуальной информации о программах стажировок, практического обучения и перспективах трудоустройства. Отдельное внимание было уделено возможностям международного сотрудничества, представленным отделом международных программ учебного заведения.

В ходе выставки участники имели возможность детально ознакомиться с карьерными перспективами, предлагаемыми компаниями-партнерами. Основными темами для обсуждения стали программы стажировок в ведущих организациях, совместные образовательные инициативы, а также

современные требования к профессиональным компетенциям специалистов. Особую ценность представляла возможность непосредственного диалога с представителями работодателей, позволяющая составить объективное представление о текущих тенденциях на рынке труда.

Мероприятие отличалось высоким уровнем организации и представительным составом участников, объединив более 100 компаний из различных отраслей экономики. Экспозиция заняла все доступные площади университетского корпуса, создав насыщенную деловую атмосферу. Интерактивный формат мероприятия, включавший квестовую программу с системой поощрений, способствовал активному вовлечению посетителей. Представители компаний-участников продемонстрировали разнообразные подходы к презентации своих организаций, включая проведение мастер-классов, профессиональных игр и тематических опросов.

Посещение выставки позволило получить ценный опыт профессиональной ориентации, установить первичные контакты с потенциальными работодателями и составить комплексное представление о текущих возможностях профессионального развития. Организаторам мероприятия удалось создать эффективную платформу для диалога между представителями образовательного сообщества и бизнес-структур, что соответствует современным тенденциям развития системы профессионального образования. Полученная в ходе мероприятия информация представляет значительную ценность для дальнейшего профессионального самоопределения и планирования карьерного роста.

Более подробный отчет с фотографиями находится в папке «reports» репозитория по проектной практике.

Вариативная часть задания

Из списка, представленного в репозитории codecrafters.io/build-your-own-x был выбран проект [Python: How To Create a Telegram Bot Using Python.](#)

Исследование процесса создания Telegram-бота с использованием Python на основе руководства с сайта freeCodeCamp позволило детально изучить необходимые шаги для реализации подобного проекта. В первую очередь требуется зарегистрировать нового бота через платформу Telegram, для чего необходимо взаимодействовать с BotFather — официальным ботом для создания и управления ботами. После ввода команды /newbot и указания

имени бота и его username BotFather предоставляет токен доступа, который является ключевым элементом для интеграции с API Telegram.

Для разработки бота на Python используется библиотека python-telegram-bot, которая предоставляет удобные инструменты для обработки сообщений и команд. Установка осуществляется через менеджера пакетов pip командой pip install python-telegram-bot. После этого создается новый Python-файл, в котором инициализируется экземпляр бота с использованием полученного токена. Основная логика работы бота строится вокруг обработчиков (handlers), которые реагируют на определенные команды или текстовые сообщения.

Пример простого бота, отвечающего на команду /start, демонстрирует базовый функционал. Сначала импортируется необходимый модуль Application из библиотеки, для непосредственной работы с созданием продукта, затем создается асинхронная функция, которая будет обрабатывать команду. Далее экземпляр Application настраивается с использованием токена, и к нему добавляется обработчик для команды /start. Запуск бота осуществляется методом run_polling(), что позволяет ему непрерывно ожидать новые сообщения.

Для основного алгоритма работы бота использовалось существующее решение, разработанное в рамках дисциплины «Проектная деятельность», проект – «Электронный мастер-консультант дилерского автотехцентра» (рисунок 7).

АЛГОРИТМ САМОСТОЯТЕЛЬНОЙ ЗАПИСИ НА ОБСЛУЖИВАНИЕ В АВТОТЕХЦЕНТР

московский
политех

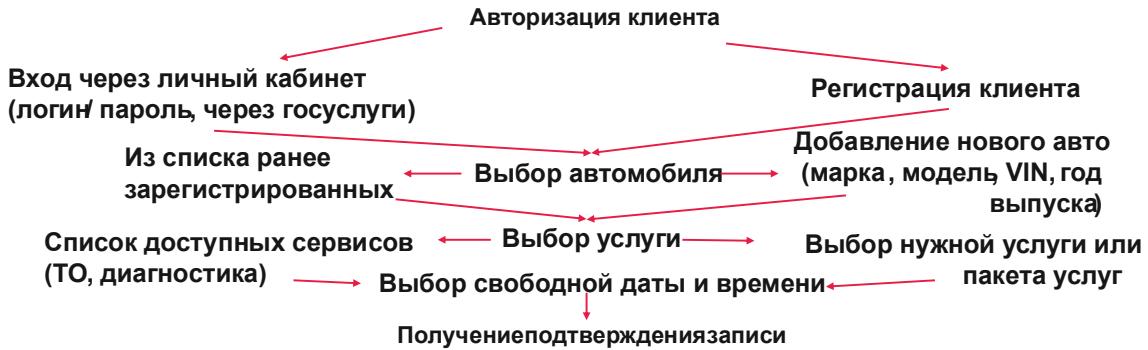


Рисунок 7—Алгоритм, использованный для реализации Телеграмм-бота

Для расширения функциональности можно добавить дополнительные обработчики, например, для ответа на произвольный текст или обработки callback-запросов от inline-кнопок. В руководстве также рассматривается возможность развертывания бота на облачном сервисе, таком как Heroku, чтобы обеспечить его круглосуточную работу. В заключение стоит отметить, что создание Telegram-бота с помощью Python является достаточно простым процессом благодаря хорошо документированным библиотекам и доступным руководствам, что делает эту технологию популярной среди разработчиков.

Техническое руководство

Данное руководство описывает процесс создания Telegram-бота, имитирующего работу электронного мастера-консультанта для дилерского автотехцентра. Бот позволяет пользователям записываться на услуги, просматривать свои записи и изменять личные данные.

Перед разработкой бота необходимо изучить основные функции, которые должны быть доступны клиентам автотехцентра. В данном случае были выделены следующие ключевые возможности:

1. Запись на услуги (техническое обслуживание, диагностика, консультации, дополнительные услуги).

2. Просмотр активных записей.

3. Изменение личных данных пользователя.

Для реализации этих функций использовался Python и библиотека python-telegram-bot, предоставляющая удобный интерфейс для работы с Telegram Bot API.

1. Установка необходимых инструментов

Для начала работы потребуется:

- Установить Python (рекомендуется версия 3.8+).
- Установить библиотеку python-telegram-bot:

2. Создание бота в Telegram

1. Откройте Telegram и найдите бота @BotFather.

2. Введите команду /newbot и следуйте инструкциям:

- Укажите имя бота (например, AutoServiceBot).
- Получите токен доступа (сохраните его в безопасном месте).

3. Настройка проекта

Создайте файл bot.py и добавьте следующий код:

```
import os
from telebot import TeleBot, types
from datetime import datetime, timedelta
BOT_TOKEN = os.getenv('BOT_TOKEN')
bot = TeleBot(BOT_TOKEN)
users = {}
user_states = {}
```

4. Определение структуры услуг

Бот должен предоставлять клиентам выбор из нескольких категорий услуг. Для этого используется словарь SERVICES:

```
SERVICES = {
    "Техническое обслуживание": [
        "T0-1 (10 000 км)",
        "T0-2 (30 000 км)",
        "T0-3 (60 000 км)"
    ],
}
```

```
"Диагностика": [
    "Компьютерная диагностика",
    "Диагностика двигателя",
    "Проверка подвески"
],
"Консультации": [
    "Подбор автомобиля",
    "Онлайн-консультация механика",
    "Оценка состояния авто"
],
"Доп. услуги": [
    "Шиномонтаж",
    "Химчистка салона",
    "Полировка кузова",
    "Установка доп. оборудования"
]
}
```

5. Создание клавиатур

Для удобства навигации используются Reply-клавиатуры:

```
python
Copy
Download
def main_menu_keyboard():
    keyboard
    = types.ReplyKeyboardMarkup(resize_keyboard=True)
    keyboard.add("Записаться на услугу", "Мои записи",
    "Изменить данные")
    return keyboard

def services_keyboard():
    keyboard
    = types.ReplyKeyboardMarkup(resize_keyboard=True)
    for service in SERVICES.keys():
        keyboard.add(service)
```

```
    keyboard.add("Отмена")
```

```
return keyboard
```

6. Обработка команд

Стартовая команда

```
python
```

```
Copy
```

```
Download
```

```
@bot.message_handler(commands=['start'])
```

```
def start(message):
```

```
    user_id = message.chat.id
```

```
    if user_id not in users:
```

```
        users[user_id] = {
```

```
            "full_name": None,
```

```
            "cars": [],
```

```
            "appointments": []
```

```
}
```

```
    user_states[user_id] = "GET_NAME"
```

```
    bot.send_message(user_id, "👋 Добро пожаловать!"
```

Ведите ваше имя и фамилию:")

```
else:
```

```
    user_states[user_id] = "MAIN_MENU"
```

```
    bot.send_message(user_id, "Главное меню:",
```

```
reply_markup=main_menu_keyboard())
```

Обработка ввода имени

```
@bot.message_handler(func=lambda msg:
```

```
user_states.get(msg.chat.id) == "GET_NAME")
```

```
def get_name(message):
```

```
    user_id = message.chat.id
```

```
    users[user_id]["full_name"] = message.text
```

```
    user_states[user_id] = "MAIN_MENU"
```

```
    bot.send_message(
```

```
        user_id,
```

```
        f"✅ Данные сохранены, {message.text}! \nВыберите
```

действие:",

```
    reply_markup=main_menu_keyboard()
)
```

7. Запись на услугу

Выбор категории услуги

```
@bot.message_handler(func=lambda msg:
user_states.get(msg.chat.id) == "SELECT_SERVICE")
def select_service(message):
    user_id = message.chat.id
    if message.text == "Отмена":
        user_states[user_id] = "MAIN_MENU"
        bot.send_message(user_id, "Главное меню",
reply_markup=main_menu_keyboard())
    elif message.text in SERVICES:
        users[user_id]["selected_service"] = message.text
        user_states[user_id] = "SELECT_SUBTYPE"
        bot.send_message(
            user_id,
            f"Выберите услугу ({message.text}):",
            reply_markup=subtypes_keyboard(message.text)
        )
```

Подтверждение записи

После выбора даты и времени бот сохраняет запись и отправляет подтверждение:

```
users[user_id]["appointments"].append({
    "service": users[user_id]["selected_service"],
    "subtype": users[user_id]["selected_subtype"],
    "date": date,
    "time": time
})

confirm_msg = (
    "✅ Запись оформлена!\n\n"
    f"👤 Клиент: {users[user_id]['full_name']}\n"
    f"🔧 Услуга: {users[user_id]['selected_subtype']}\n"
```

```
f"📅 Дата: {date}\n"
f"⌚ Время: {time}\n\n"
"Спасибо за доверие нашему автотехцентру!"
```

)

```
bot.send_message(
    user_id,
    confirm_msg,
    reply_markup=main_menu_keyboard()
)
```

8. Запуск бота

```
if __name__ == "__main__":
    print("🚗 Бот автотехцентра запущен!")
    bot.infinity_polling()
```

В данном руководстве рассмотрен процесс создания Telegram-бота для автотехцентра. Основные этапы включают настройку проекта, создание интерфейса взаимодействия с пользователем и реализацию логики записи на услуги. Бот может быть доработан добавлением интеграции с базой данных и уведомлений о записях.

Это же руководство продублировано в файле формата Markdown в репозитории проектной практики.

Вывод

В ходе выполнения проектной практики был получен комплексный опыт работы над ИТ-проектом, включающий все этапы от планирования до

реализации. Основное внимание уделялось освоению современных инструментов разработки и организации рабочего процесса, что позволило сформировать профессиональные компетенции в области управления проектами и командной работы.

Практика началась с настройки рабочей среды и изучения системы контроля версий Git, что стало фундаментом для эффективной организации всего рабочего процесса. Были освоены ключевые операции работы с репозиториями, включая создание веток, фиксацию изменений и синхронизацию с удаленным хранилищем. Особое внимание уделялось содержательности сообщений к коммитам, что позволило поддерживать прозрачность истории изменений проекта.

Значительная часть практики посвящалась освоению языка разметки Markdown для оформления проектной документации. В результате были созданы все необходимые документы, включая основной файл README.md, описание задания и отчетные материалы. Работа с Markdown позволила выработать навыки структурированного представления информации и оформления технической документации.

Особое значение имел этап разработки статического веб-сайта проекта, в ходе которого были применены на практике знания HTML, CSS и JavaScript. Созданный веб-ресурс соответствует современным требованиям к пользовательским интерфейсам и демонстрирует владение актуальными технологиями веб-разработки.

Важным аспектом практики стало участие в выставке индустриальных партнеров, которое позволило познакомиться с требованиями работодателей и современными тенденциями в ИТ-отрасли. Полученная информация была учтена при доработке проекта и планировании дальнейшего профессионального развития.

В результате прохождения практики достигнуты все поставленные цели: освоены современные инструменты разработки, получен опыт работы над реальным проектом, развиты профессиональные и коммуникативные навыки.

Выполненный проект демонстрирует готовность к решению практических задач в профессиональной деятельности и соответствует требованиям современного ИТ-рынка. Приобретенные знания и навыки будут использованы в дальнейшей учебной и профессиональной деятельности.

Список литературы

1. **CSS** [Электронный ресурс] // Дока. – URL: <https://doka.guide/css/>
2. **HTML** [Электронный ресурс] // Дока. – URL: <https://doka.guide/html/>

3. **Python Telegram Bot** [Электронный ресурс] // Официальная документация. – URL: <https://docs.python-telegram-bot.org/>
4. **Markdown** [Электронный ресурс] // Дока. – URL: <https://doka.guide/tools/markdown/>
5. **GitHub Desktop** [Электронный ресурс] // GitHub Docs. – URL: <https://docs.github.com/ru/desktop/overview/getting-started-with-github-desktop>
6. Руководство по разработке Telegram-ботов [Электронный ресурс] // Apress. – URL: <https://www.apress.com/gp/book/9781484238241>
7. Как создать Telegram-бота на Python [Электронный ресурс] // FreeCodeCamp. – URL: <https://www.freecodecamp.org/news/how-to-create-a-telegram-bot-using-python/>
8. Telegram Bot API [Электронный ресурс] // Официальная документация Telegram. – URL: <https://core.telegram.org/bots/api>