УДК 004.512

ПРИМЕНЕНИЕ ЧАТ-БОТА ДЛЯ УПРАВЛЕНИЯ ПРОИЗВОДСТВЕННЫМИ ЗАКАЗАМИ НА ПРОМЫШЛЕННОМ ПРЕДПРИЯТИИ

Данилюк Т.И., студент гр. ИТб-212, IV курс
Научный руководитель Алексеева Г.А., старший преподаватель
Кузбасский государственный технический университет имени Т.Ф. Горбачёва
г. Кемерово

Современные промышленные предприятия сталкиваются с необходимостью оптимизации управления производственными процессами, включая контроль заказов, мониторинг их выполнения и оперативное взаимодействие между сотрудниками. Внедрение цифровых решений, таких как телеграмм-боты, позволяет автоматизировать рутинные операции, повысить прозрачность производства и сократить время на согласование задач. В данной статье рассматривается разработка и внедрение телеграмм-бота для управления производственными заказами на промышленном предприятии. Описаны ключевые функции системы, архитектура взаимодействия с пользователями, а также преимущества использования подобных решений в условиях цифровой трансформации производства.

Цифровизация промышленных предприятий является одним из ключевых трендов. Внедрение интеллектуальных систем управления производством позволяет сократить издержки, повысить скорость принятия решений и минимизировать человеческий фактор в критически важных процессах. Одним из инструментов, способствующих автоматизации взаимодействия между сотрудниками и производственными системами, являются чат-боты, интегрированные в популярные мессенджеры, такие как Telegram [1].

Предлагаемый телеграмм-бот предназначен для управления заказами на производство, предоставляя пользователям следующие возможности:

- просматривать текущие заказы (находящие в работе),
- отправлять новые заказы в производство,
- завершать производство заказов,
- добавлять комментарии для уточнения деталей.

Применение данного решения позволит снизить нагрузку на корпоративные системы учета (ERP, MES) и обеспечит мобильность доступа к информации.

Чат-бот написан на языке программирования C# при помощи среды разработки Visual Studio, при создании использовалась СУБД Microsoft SQL Server.

Бот взаимодействует с начальником производства, выводя ему полный список заказов, которые находятся в производстве или готовы к производству. Пользователь может выбрать действие, которое необходимо выполнить с конкретным заказом: отправить в производство, завершить производство, добавить комментарий к заказу.

Процесс создания бота включает следующие этапы:

1. Регистрация бота в Telegram с помощью BotFather – официального инструмента для разработки ботов (рисунок 1).
2. Назначение имени бота и получение уникального API-токена для авторизации.
3. Настройка списка команд.

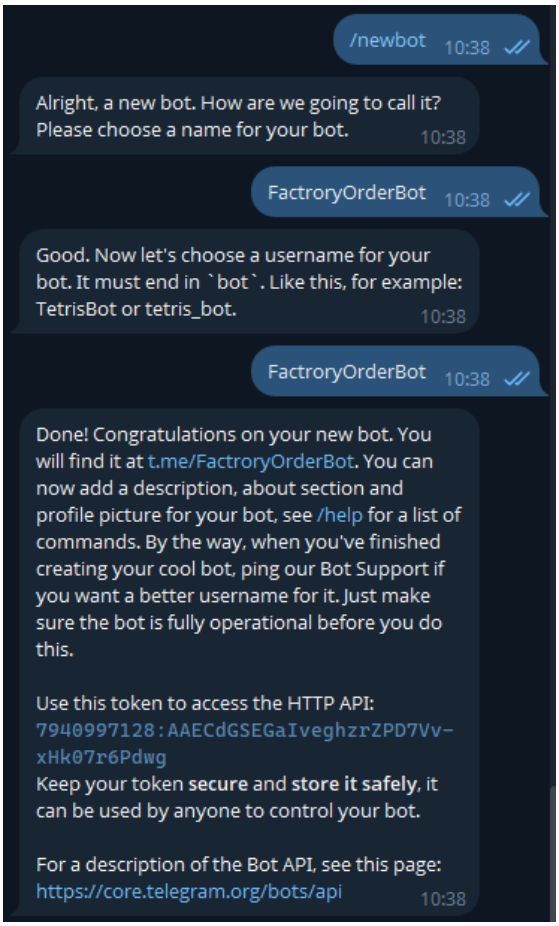

Рисунок 1 – Процесс создания бота через BotFather

Система состоит из трех основных компонентов:

1. Telegram Bot API – принимает запросы пользователей и передает их backend-сервису [2].
2. Backend-сервис (C#) – обрабатывает команды, взаимодействует с базой данных и формирует ответы [3].
3. База данных (MSSQL) – хранит информацию о заказах и пользователях.

Основные команды бота:

- /start – запуск бота и отображение списка доступных команд;
- /view_orders – просмотр заказов в производстве;
- /start_order – отправка нового заказа в производство;
- /complete_order – завершение производства заказа;
- /add_comment – добавление комментария к заказу.

Например, при отправке команды /start_order, после чего бот запрашивает номер заказа, который необходимо отправить в производство и после подтверждения данные сохраняются в базе данных, а статус заказа в системе обновляется.

На рисунке 2 приведён фрагмент кода для подключения к чат-боту.

```
// Строка подключения
private ITelegramBotClient bot = new TelegramBotClient("7940997128:AAECdGSEGaIveghzrZPD7Vv-xHk07r6Pdwg");
// Создание экземпляра бота (ID главного чата)
private long MainChatId = 473811738;
```

Рисунок 2 – Подключение к чат-боту

Для того, чтобы пользователь мог взаимодействовать с ботом нужно прописать команды в коде приложения (рисунок 3).

```
case "заказывпроизводстве":
    await ShowProductionOrders(botClient, chat, OrderStatus.InProduction);
    return;
case "заказыготовыекпроизводству":
    await ShowProductionOrders(botClient, chat, OrderStatus.ReadyToProduction);
    return;
case "/production":
    await botClient.DeleteMessageAsync(chat.Id, message.MessageId);
    await ShowProductionMenu(botClient, chat);
    return;
```

Рисунок 3 – Команды взаимодействия с ботом

Так же важно прописать обработку самих запросов, которые может отправлять пользователь (call-back запросы) (рисунок 4).

```
private static async void RequestOrderComment(ITelegramBotClient bot, Chat chat, int orderId)
{
    using (var context = new ThContext())
    {
        var order = context.SallerOrders.Find(orderId);
        if (order == null)
        {
            await bot.SendTextMessageAsync(chat, $"Заказ с номером {orderId} не найден");
            return;
        }

        await bot.SendTextMessageAsync(chat,
            $"Введите комментарий для заказа №{orderId}:");
    }
}
```

Рисунок 4 – Пример call-back запроса для добавления комментария

Разработанный чат-бот на C# с использованием MSSQL демонстрирует высокую эффективность в управлении производственными заказами. Его внедрение позволяет значительно оптимизировать процессы на

промышленном предприятии, а так же сделать их более мобильными, благодаря возможности взаимодействия с ним при помощи мобильного телефона. В перспективе планируется расширение функционала за счет:

- интеграции с IoT-датчиками для автоматического обновления статусов;
- внедрения аналитики на основе машинного обучения для прогнозирования сроков выполнения заказов.

Список литературы:

1. Боты в Telegram: Возможности и преимущества. [Электронный ресурс] – Режим доступа: https://workspace.ru/blog/boty-v-telegram-vozmozhnosti-i-preimuschestva/, свободный (дата обращения: 23.03.2025).

2. Справочник по Telegram Bot API. [Электронный ресурс] – Режим доступа: https://tlgrm.ru/docs/bots/api, свободный (дата обращения: 23.03.2025).

3. Пишем бота telegram на C#. [Электронный ресурс] – Режим доступа: http://aftamat4ik.ru/pishem-bota-telegram-na-c/, свободный (дата обращения: 23.03.2025).