Анализ требований: В этом этапе проводится анализ требований предприятия к системе учета продукции. Изучаются процессы производства, учета материалов, контроля качества, управления запасами и продажи продукции. Требования клиентов анализируются и документируются для последующего использования в проектировании системы.

Планирование и проектирование: На основе анализа требований разрабатывается архитектура и функциональность системы учета продукции. Определяются основные модули и подсистемы, такие как модуль учета материалов, модуль учета производства, модуль управления запасами и т. д. Также определяются требования к пользовательскому интерфейсу и функциональным возможностям системы.

Конфигурирование и настройка: На этом этапе типовая конфигурация 1С:Предприятие настраивается в соответствии с требованиями предприятия. Определяются параметры и настройки системы, создаются пользовательские формы и отчеты, настраивается логика работы системы с учетом требований учета продукции.

Разработка дополнительных модулей: Если требуется дополнительная функциональность, которая не предусмотрена типовой конфигурацией 1С:Предприятие, проводится разработка дополнительных модулей. Это может включать разработку модулей для учета специфических видов продукции, интеграцию с оборудованием или внешними системами, автоматизацию определенных процессов и т. д.

Тестирование: После завершения разработки системы проводится тестирование. Это включает функциональное тестирование, проверку правильности работы каждого модуля и подсистемы, а также интеграционное тестирование для проверки взаимодействия между модулями. Результаты тестирования анализируются и исправляются выявленные ошибки.

Внедрение и обучение: После успешного тестирования система учета продукции внедряется на предприятии. Этот этап включает установку и настройку системы на серверах предприятия, импорт начальных данных, обучение пользователей работе с новой системой и переход к полноценному использованию системы учета продукции.

Полученная система будет иметь достаточный функционал и являться надежным средством автоматизации учета продукции на любом предприятии.

А. В. Бунченко, Е. В. Рафалова
(ГГУ имени Ф. Скорины, Гомель)

ИСПОЛЬЗОВАНИЕ ИНТЕРФЕЙСА OPENAI ДЛЯ ИНТЕГРАЦИИ ЯЗЫКОВЫХ МОДЕЛЕЙ В ВЕБ-ПРИЛОЖЕНИЯ

С развитием искусственного интеллекта и машинного обучения, языковые модели, такие как OpenAI GPT открыли новые возможности для разработки веб-приложений. Эти модели могут генерировать текст, отвечать на вопросы, суммировать информацию, обеспечивать поддержку в реальном времени и выполнять множество других задач, связанных с обработкой естественного языка. Использование программного интерфейса, или API (Application Programming Interface), от OpenAI для интеграции языковых моделей в проекты открывает широкие возможности для разработчиков и предприятий в различных областях.

Первый шаг к работе с программным интерфейсом – регистрация на официальном сайте OpenAI и создание учетной записи. После регистрации будет предоставлен API-ключ, который необходим для аутентификации запросов к API. Этот ключ является секретным и должен храниться в безопасности, чтобы предотвратить несанкционированный доступ к аккаунту.

Для аутентификации запросов к программному интерфейсу нужно использовать полученный API-ключ. Как правило ключ указывается в заголовке запроса Authorization как Bearer токен. Важно гарантировать, что запросы отправляются через защищенное соединение (HTTPS), чтобы обеспечить конфиденциальность ключа.

API OpenAI позволяет отправлять текстовые запросы и получать ответы от языковых моделей. Запросы могут быть настроены для выполнения различных задач, таких как генерация текста, суммирование, перевод, ответы на вопросы и многое другое. При отправке есть возможность указать различные параметры, такие как длина ответа, температура (которая определяет уровень креативности или случайности в ответах), контекст (предшествующий текст, который модель должна учитывать при генерации ответа) и др.

Ответы от API возвращаются в формате JSON и содержат сгенерированный текст вместе с иной полезной информацией, такой как статус запроса или использованные токены. Разработчики должны обрабатывать эти ответы, чтобы извлекать нужную информацию и интегрировать ее в приложение.

Важно учитывать, что использование API OpenAI обычно подразумевает оплату в зависимости от количества запросов и объема использованных токенов (единиц обработанного текста). Поэтому необходимо отслеживать использование API для управления расходами, особенно при масштабировании приложения. OpenAI предлагает различные инструменты и информационные таблицы для мониторинга использования токенов и оптимизации стоимости.

При работе с программным интерфейсом OpenAI важно уделять внимание вопросам безопасности и конфиденциальности. Это включает в себя защиту API-ключа, обеспечение безопасности данных пользователей и соблюдение принципов этичного использования искусственного интеллекта. Компания OpenAI предоставляет рекомендации и лучшие практики по обеспечению безопасности и соблюдению этических норм при использовании их технологий.

API OpenAI предоставляет разработчикам доступ к передовым инструментам искусственного интеллекта, открывая новые возможности для создания инновационных приложений. Однако успешное использование технологии требует понимания основ работы с API, а также внимательного отношения к вопросам стоимости, безопасности и конфиденциальности.

Д. В. Василенко
(ГГУ имени Ф. Скорины, Гомель)
Науч. рук. **А. И. Кучеров**, ст. преподаватель

ТЕХНОЛОГИИ МИКРОСЕРВИСНОЙ АРХИТЕКТУРЫ

Давайте начнем с краткого введения в микросервисную архитектуру. Микросервисная архитектура – это методология разработки, которая структурирует приложения как набор небольших, независимых и взаимодействующих между собой сервисов. Каждый сервис выполняет определенную функцию и может быть развернут, обновлен и масштабирован независимо друг от друга.

В основной части статьи мы рассмотрим технологии, которые могут быть использованы для реализации микросервисной архитектуры, и в частности, посвятим Entity Framework Core(EF), JWT и базе данных PostgreSQL, Asp Net Core ,Unit Tests, Docker, RabbitMQ, Api Gateway (Ocelot).

EF – Эта технология используется для взаимодействия с базой данных. Каждый микросервис, который работает с данными, будет использовать EF для выполнения операций CRUD (создание, чтение, обновление, удаление) с базой данных.

В JWT содержится три части: заголовок, полезные данные и подпись. Заголовок содержит информацию о типе токена и алгоритме шифрования, полезные данные содержат данные о пользователе или другой информации, которую необходимо передать, а подпись используется для проверки подлинности токена.

PostgreSQL – Каждый микросервис, который работает с данными, будет использовать PostgreSQL для хранения и извлечения данных.