**Требования к внешним интеграциям**

#### **1. Требования к безопасности**

* **Шифрование данных**:
  + Все данные, передаваемые между системами, должны шифроваться с использованием современных протоколов (например, TLS 1.2/1.3).
  + Данные, хранящиеся на стороне партнёров, также должны быть зашифрованы.
* **Контроль доступа**:
  + Ограничить доступ к API только авторизованным системам и пользователям.
  + Реализовать принцип минимальных привилегий: доступ предоставляется только к тем данным и функциям, которые необходимы для выполнения задач.
* **Валидация данных**:
  + Все входящие данные должны проверяться на корректность и безопасность (например, защита от SQL-инъекций, XSS-атак).
* **Мониторинг и аудит**:
  + Вести журналы всех запросов и ответов для последующего анализа и аудита.
  + Реализовать систему оповещений о подозрительной активности.
* **Защита от DDoS-атак**:
  + Использовать механизмы защиты от распределённых атак на отказ в обслуживании (например, ограничение количества запросов в единицу времени).
* **Согласование контрактов API**:
  + Все контракты API должны быть проверены на соответствие требованиям безопасности, включая запрет на передачу избыточных данных (например, персональных данных, не требующихся для выполнения задачи).

#### **2. Протоколы аутентификации и авторизации**

* **Аутентификация**:
  + Использовать протокол OAuth 2.0 для аутентификации внешних систем.
  + Для внутренних интеграций можно использовать API-ключи или JWT.
* **Авторизация**:
  + Реализовать ролевую модель для управления правами доступа к API.
  + Использовать Scope в OAuth 2.0 для ограничения доступа к определённым ресурсам.
* **Многофакторная аутентификация (MFA)**:
  + Для критически важных операций (например, изменение данных или доступ к финансовой информации) использовать MFA.
* **Срок действия токенов**:
  + Токены доступа должны иметь ограниченный срок действия (например, 1 час) и возможность отзыва.

#### **3.** **Организация взаимодействия между системами**

* **Стандартизация API**:
  + Использовать RESTful API или GraphQL для взаимодействия между системами.
  + Все API должны соответствовать единым стандартам компании (например, использование JSON для передачи данных).
* **API Gateway**:
  + Внедрить API Gateway для централизованного управления всеми внешними и внутренними API.
  + API Gateway должен обеспечивать:
    - Маршрутизацию запросов.
    - Ограничение скорости запросов (Rate Limiting).
    - Кэширование ответов.
    - Логирование и мониторинг.
* **Асинхронное взаимодействие**:
  + Для длительных операций (например, обработка данных или интеграция с госорганами) использовать асинхронное взаимодействие через очереди сообщений (например, RabbitMQ, Kafka).
* **Форматы данных**:
  + Использовать стандартные форматы данных (JSON, XML) для обеспечения совместимости.
  + Все данные должны быть валидированы на соответствие схеме (например, JSON Schema).
* **Обработка ошибок**:
  + Реализовать единый стандарт обработки ошибок (например, HTTP-коды ошибок и структурированные сообщения об ошибках).
* **Документация API**:
  + Предоставлять подробную документацию для всех API, включая:
    - Описание endpoints.
    - Примеры запросов и ответов.
    - Требования к аутентификации и авторизации.
* **Тестирование интеграций**:
  + Проводить регулярное тестирование интеграций (например, нагрузочное тестирование, проверка на уязвимости).
* **Резервирование и отказоустойчивость**:
  + Обеспечить отказоустойчивость интеграций за счёт резервирования каналов связи и использования нескольких провайдеров (если возможно).