**Пример Технического Задания на Разработку Веб-приложения «Система Управления Задачами»**

**1. Введение**

1.1 **Цель документа**

Данный документ описывает требования к разработке веб-приложения «NFT Биржа», предназначенного для покупки, продажи и коллекционирования невзаимозаменяемых токенов (NFT).

1.2 **Актуальность и назначение проекта**

Актуальность и назначение проекта

С ростом популярности NFT и их применения в различных сферах, таких как искусство, музыка и игры, возникает необходимость в создании удобной и безопасной платформы для торговли этими токенами. Основная цель проекта — предоставить пользователям возможность легко и безопасно обмениваться NFT.

1.3 **Основные пользователи системы**

Система предназначена для использования сотрудниками компании:

* **Покупатели — пользователи, желающие приобрести NFT.**
* **Продавцы — пользователи, желающие выставить свои NFT на продажу.**
* **Коллекционеры — пользователи, собирающие NFT и желающие управлять своей коллекцией**

**2. Цели и задачи проекта**

2.1 **Основные цели**

* Создание безопасной и удобной платформы для торговли NFT.
* Обеспечение прозрачности сделок и защиты прав пользователей.

2.2 **Конкретные задачи**

* Разработка функционала для создания, покупки и продажи NFT.
* Реализация системы аукционов для продажи NFT.
* Внедрение системы управления пользовательскими профилями и коллекциями.
* Обеспечение интеграции с криптовалютными кошельками для проведения транзакций.

**3. Требования к системе**

3.1 **Функциональные требования**

* Создание NFT: Пользователи должны иметь возможность создавать NFT, загружая цифровые файлы и заполняя метаданные.
* Торговля NFT: Возможность выставления NFT на продажу, включая фиксированные цены и аукционы.
* Управление коллекцией: Пользователи должны иметь возможность просматривать и управлять своей коллекцией NFT.
* Поиск и фильтрация: Реализация системы поиска и фильтрации NFT по различным критериям (категория, цена, популярность)

3.2 **Нефункциональные требования**

* Производительность: Время отклика интерфейса не должно превышать 2 секунд при загрузке страниц.
* Масштабируемость: Платформа должна поддерживать до 10000 активных пользователей одновременно.
* Безопасность: Все транзакции должны быть защищены с использованием современных методов шифрования.

**4. Требования к пользовательскому интерфейсу**

4.1 **Основные экраны**

* **Главная страница: Отображение популярных NFT, а также разделы для поиска и фильтрации.**
* **Страница NFT: Подробная информация о конкретном NFT, включая изображение, описание, цену и кнопку для покупки.**
* **Профиль пользователя: Информация о пользователе, его коллекции NFT и история транзакций.**
* **Панель администратора:** интерфейс для просмотра статистики и отчетов.

4.2 **Навигация**

Система должна включать меню навигации с разделами: "Главная", "Моя коллекция", "Продажа", "Настройки".

4.3 **Юзабилити**

* Интерфейс должен быть интуитивно понятным и доступным для пользователей с разным уровнем технической подготовки.
* Цветовая кодировка статусов NFT (например, "На продаже", "Продано", "В аукционе").*Пример*: На главной странице задачи отображаются в виде списка с возможностью сортировки по дате и приоритету.

**5. Требования к технической реализации**

5.1 **Языки и технологии**

* **Backend**: Laravel.
* **Frontend**: Vue.js.
* **База данных**: mySQL.

5.2 **Архитектура системы**

* Микросервисная архитектура с REST API для взаимодействия между модулями.
* Хранение данных о пользователях и NFT в реляционной или документной базе данных.

5.3 **Интеграции**

* Интеграция с криптовалютными кошельками (например, MetaMask) для проведения транзакций.
* Подключение к внешним API для получения информации о ценах на криптовалюту.
* Интеграция с системами авторизации, такими как Google OAuth.

**6. Требования к безопасности**

6.1 **Аутентификация и авторизация**

* Использование Auth для авторизации пользователей через социальные сети и криптовалютные кошельки.
* Разграничение прав доступа для различных ролей пользователей.

6.2 **Шифрование данных**

* **HTTPS** для всех передаваемых данных.
* **Хранение паролей**: хэширование через bcrypt.

6.3 **Политики доступа**

* Защита от SQL-инъекций и XSS.
* Ограничение по IP для административного доступа.

**7. Ограничения и допущения**

7.1 **Технические ограничения**

* Серверная часть должна работать на облачных платформах (например, AWS или Azure).
* Платформа должна быть доступна на мобильных устройствах и поддерживать адаптивный дизайн

7.2 **Финансовые ограничения**

* Бюджет проекта: до 300000 ₽, включая тестирование и документацию.

7.3 **Сроки выполнения**

* Полная реализация проекта — 1 месяцев с начала разработки.

**8. Требования к тестированию и приемке**

8.1 **Типы тестирования**

* **Функциональное тестирование**: проверка всех реализованных функций.
* **Нагрузочное тестирование**: тестирование на работе с 5000+ пользователями.
* **Тестирование безопасности**: проверка на уязвимости.

8.2 **Критерии приемки**

* Полное выполнение всех функциональных и нефункциональных требований.
* Прохождение не менее 95% тестов по результатам приемочных испытаний.

*Пример*: Если в процессе тестирования будет выявлено более 5 критических ошибок, система не может быть принята в эксплуатацию.

**9. Требования к документации**

9.1 **Пользовательская документация**

* Руководство пользователя с инструкциями по созданию, редактированию и управлению задачами.

9.2 **Техническая документация**

* Описание архитектуры, структура базы данных, API.

**10. План реализации**

10.1 **Этапы разработки**

1. Анализ и проектирование — 2 дня.
2. Разработка прототипа — 3 дня.
3. Полноценная разработка и тестирование — 2 недели.
4. Внедрение и обучение — 1 месяц(после реализации).

10.2 **Сроки выполнения этапов**

* Дата начала: ноябрь 2024 года.
* Дата завершения: декабрь 2024 года.

10.3 **Ответственные лица**

* Менеджер проекта: Хижняковав К.Д.
* Ведущий разработчик: Хижняковав К.Д.

**11. Приложения**

* **Приложение A**: Детализированные схемы архитектуры.
* **Приложение B**: Примеры интерфейсов.
* **Приложение C**: Технические спецификации для интеграции с API других систем.