

Проект кредитного пула на смарт-контрактах (аналог Compound.Finance)

Краткое описание проекта

Наш проект позволяет объединить возможности сервиса кредитования compound.finance (и некоторых его аналогов таких как dydx, dharm и др.) и возможности коллективного владения и управления через ДАО (блокчейн Ethereum). Тем самым токенхолдеры ДАО получают возможность получать доход с работы сервиса и управлять им для более эффективного реагирования на рыночную конъюнктуру.

Цель проекта:

- Создание улучшенного аналога сервиса compound.finance и устранение существующих недостатков.
- Улучшение сервиса будет осуществляться за счет передачи на ДАО (децентрализованная автономная организация) функций по управлению сервисом. Платформа для управления ДАО - Aragon.

Основные недостатки сервиса compound.finance:

1. У владельцев сервиса нет возможности регулировать ставки рефинансирования
2. У частных инвесторов нет возможности инвестировать в сервис и получать дивиденды с его работы.
3. Нет автоматизации установки плеча
4. Недостаточное количество активов, которые могут быть использованы.
5. Пользовательский интерфейс может быть проще и интуитивно понятнее

Ключевые технологии

- Децентрализованная автономная организация на платформе Aragon. Платформа Aragon позволяет создавать децентрализованные организации (ДАО) на блокчейн эфириум. При этом децентрализация включает также в себя использование децентрализованного доменного имени.
- Протокол кредитования (Lending protocol) Общее описание compound для разработчиков: <https://compound.finance/developers>
- White paper: <https://compound.finance/documents/Compound.Whitepaper.pdf>

Необходимые компетенции:

- Solidity, опыт разработки dapps-ов (<https://solidity.readthedocs.io/en/v0.5.13/>)
- node.js, web3-react (<https://www.npmjs.com/package/react-web3>)
- Aragon stack (<https://hack.aragon.org/docs/stack>)

Цели и задачи для приложения

Проведение форка проекта compound.finance, а также разработка веб интерфейса сети. Создание тестовой сети, а также инициализация смарт контрактов в сети ethereum по стандарту ERC-20.

MVP: Добавление функционала DAO Aragon к форку compound.finance

Конечный продукт должен быть построен на базе смартконтрактов, уметь начислять токены проекта, пользователям о которых заносится запись в блокчейн сеть ethereum.

Необходимо иметь возможность подключения кошельков, которые работают с ERC-20.

Целевая аудитория будущего приложения

Аудитория заказчика. Пользователи сервисов DeFi (decentralized finances).

Платформа приложения

Сеть ethereum, а также веб. Стек: php, js, solidity, aragon. Обязательно использование react технологий, фреймворки выбираем исполнитель. Продукт должен соответствовать стандартам web 3.0.

Устройство вывода приложения

Браузер (chrome, opera, firefox), поддержка ie не обязательно. Desktop, iOS, Android.

Языки, которые должны быть в приложении

Русский, английский создание веб интерфейса в виде ключей, с возможностью мультязычности. Документация проекта на русском/английском языке.

Дизайн-концепция

Согласовывается и определяется в процессе создания макетов. инструмент figma.

ОПИСАНИЕ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА (МОКАП)

<https://app.compound.finance/>

Обязательства сторон

Заказчик получает весь код а исполнитель отказывается от прав на код и подписывает соглашение о конфиденциальности (NDA)

Ключевые шаги

1. Анализ (формирование требований)
2. Техническое задание (Написание ТЗ для бэкенда, фронтенда, прототипирование)
3. Разработка макетов (активные прототипы)
4. “Ветвление” на отдельный продукт (поднимаем тестовую сеть Rinkeby)
5. Смартконтракты (Инициализация нового токена)
6. Смартконтракты (Разработка логики прочих проектов)
7. Разработка бекенд и фронтенд части
8. Написание документации, покрытие автотестами
9. Деплой в продакшен
10. Тестирование на нагрузку
11. Тестирование на взлом
12. Запуск проекта в продакшн

Команда

1. Технический руководитель / Devops
2. Разработчик смарт-контрактов
3. Разработчик веб (фронтенд)
4. Разработчик веб (бекенд)
5. Проектный менеджер (ТЗ, сбор требований)
6. Дизайнер (Макеты, прототипирование)
7. Дежурный администратор
8. Ответственный специалист от заказчика (хранение private & master keys)

Предварительная оценка

1. Описание технического задания -
2. Разработка макетов -
3. Проведение форк проекта -
4. Написание смартконтрактов -
5. Фронтенд + бекенд -
6. Документация -
7. Покрытие автотестами -
8. Тестирование на нагрузку -
9. Тестирование на взлом (расходы на вознаграждение независимым разработчикам) -

Итого оценочная стоимость:

Бриф проекта

Бриф на разработку приложения + [fork compound.finance]

Контактная информация

Заказчик	
Ф.И.О. ответственного лица и должность	
Контактный телефон (с кодом страны и города)	
Контактный E-mail для связи с Вами	
Домен на котором будет размещаться проект	
Локации использования продукта (мультиязычность, горизонтальная архитектура приложения)	

Цветовая гамма и название определяется заказчиком. Брендбук отсутствует. Дизайн систему разрабатывать нужно.

Авторизации/регистрации пользователей в приложении

Необходима страница авторизации/регистрации

Необходима возможность авторизации через соц. сети

Техническая поддержка в дальнейшем

После реализации необходимо поддерживать продукт.

Бюджет приложения + и сеть

Желаемые сроки сдачи проекта

Срочная разработка в 1 - 2 месяца.

Карта разработки продукта

Техническое задание + инфраструктура проекта

1. **Gitlab + документация**
2. **Slack**
3. **Тестовый контур проекта**

Проведение форк продукта + написание smart contract

1. **compound-protocol**
The Compound On-Chain Protocol
2. **compound-v2-subgraph**
3. **open-oracle**
The Compound Open Oracle
4. **saddle**
Ethereum Smart Contract Saddle
5. **compound-money-market**
Compound v1 Protocol [Copyright 2018 Compound]

6. eth-contract-metadata

Mapping of ethereum contract addresses to broadly accepted icons for those addresses.

7. tokens

Разработка или адаптация форка 542 смартконтрактов токенов

8. solidity

Разработка или адаптация форка смартконтрактов (блокчейн сети)

9. redis_stream

Redis Stream Processor in Elixir, built on redis

10. compound-oracle

The Compound Price Oracle

11. QuickBorrow

Borrow tokens simply by sending Ether

12. cloud-builders-community

Community-contributed images for Google Cloud Build

13. solidity-coverage

Code coverage for Solidity smart-contracts

14. redis

Docker Official Image packaging for Redis

Разработка frontend + backend продукта

1. Разработка backend веб интерфейса

2. Разработка frontend веб интерфейса

Релиз продукта

1. Тестирование

2. Проведение тестирования на взлом продукта

3. Релиз продукта