**Инструкция первоначальных действий (обязательно к выполнению):**

1. Задеплоить токен (контракт ProjectToken)
2. Задеплоить RatingSystem
3. Задеплоить EventMaster
4. Вызвать функции SetEventMaster в ProjectToken (указать адрес EM)
5. Вызвать функции SetEventMaster в RatingSystem (указать адрес EM)

Вызвать функции в ЕМ :

- Вызвать функцию SetRS в EventMaster (указать RS)

- SetToken (указать токен),

- SetComission (указать комиссию (по умолчанию 0, можно так и оставить)),

- SetWallet (указать кошелек для сбора средств)  
  
  
**Создание мероприятия:**

1. CreateEvent - создать мероприятие, адреса созданных контрактов можно будет найти по порядковому номеру в функции Events.
2. Инициализировать FR и MP, (первый аргумент при вызове функций Init – порядковый номер)  
     
     
   **Стандартные функции владения контрактом:**
3. function owner() public view returns (address)

Возвращает адрес владельца

1. function isOwner() public view returns (bool) {

Возвращает true или false в зависимости является ли отправитель транзакции владельцем.

1. function transferOwnership(address newOwner) public onlyOwner {

Передать право владения другому адресу

1. mapping (address => bool) public admin;

Возвращает true или false в зависимости является ли отправитель транзакции админом.

1. function setAdmin(address addr) public onlyOwner {

сделать определенный адрес админом

1. function deleteAdmin(address addr) public onlyOwner {

убрать право администрирования у адреса

**Контракт токена: ProjectToken**

Стандартные функции ERC20

1. function name() public view returns(string) {  
   название токена
2. function symbol() public view returns(string) {  
   сокращение
3. function decimals() public view returns(uint8) {  
   количество знаков после запятой
4. function totalSupply() public view returns (uint256) {  
   эмиссия токена
5. function balanceOf(address owner) public view returns (uint256) {  
   баланс определенного кошелька
6. function allowance(address owner, address spender) public view returns (uint256) {  
   сколько токенов одобрено к выводу для определенного адреса
7. function transfer(address to, uint256 value) public returns (bool) {

трансфер токенов

1. function approve(address spender, uint256 value) public returns (bool) {  
   одобрение к выводу
2. function transferFrom(address from, address to, uint256 value) public returns (bool) {  
   перевести одобренные к выводу токены с определенного адреса
3. function increaseAllowance(address spender, uint256 addedValue) public returns (bool) {  
   увеличить количество одобренных к выводу токенов
4. function decreaseAllowance(address spender, uint256 subtractedValue) public returns (bool) {

увеличить количество одобренных к выводу токенов

1. function burn(uint256 value) public {  
   сжечь определенное количество токенов
2. function burnFrom(address from, uint256 value) public {  
   сжечь определенное количество одобренных к выводу токенов с адреса

Нестандартные функции токена:

1. EventMaster public EM;  
   Адрес EventMaster
2. function setEventMaster(address EM\_Addr) external onlyOwner {

Установить адрес EventMaster

1. function verifiedTransfer(address to, uint256 value) external returns(bool) {

Техническая функция, вызвать ее может только контракт EventMaster.

**Контракт RatingSystem:**

1. mapping (address => uint256) public rating;

Рейтинг определенного адреса

1. EventMaster public EM;  
   Адрес EventMaster
2. function setEventMaster(address EM\_Addr) public onlyOwner {  
   Установить адрес EventMaster
3. function increaseRating(address addr, uint256 value) public allowed {

Увеличить рейтинг адреса

1. function decreaseRating(address addr, uint256 value) public allowed {  
   Уменьшить рейтинг адреса
2. function deleteRating(address addr) public allowed {  
   Удалить рейтинг адреса

**Контракт EventMaster:**

1. ProjectToken public token;  
   Адрес токена
2. RatingSystem public RS;  
   Адрес RatingSystem
3. uint256 public comission;  
   Комиссия в %
4. address public wallet;  
   Кошелек для сбора комиссии
5. mapping (address => bool) public verified;  
   Является ли адрес официальным мероприятием проекта
6. EventSample[] public events;  
   Все мероприятия по порядку (отсчет с 0)
7. function createEvent() public returns(uint256) {  
   Создать мероприятие
8. function initFundRaising(uint256 index, uint256 startUNIX, uint256 finishUNIX, uint256 minValue, uint256 maxValue, uint256 costPrice, uint256 surcharge) public {  
   Активировать ПСС. Указывается: порядковый номер мероприятия, старт, финиш, минимальный и максимальные размеры инвестиций, стоимость закупки, величина наценки в %
9. function initMarketPlace(uint256 index, uint256 startUNIX, uint256 finishUNIX, uint256 \_investorsSurcharge, uint256 \_standardSurcharge, uint256 \_fiatSurcharge) public {  
   Активировать Этап Продаж. Указывается: порядковый номер мероприятия, старт, финиш, величина наценки в % для инвесторов, для обычных пользователей, для фиата.
10. function setToken(address tokenAddr) public restricted {  
    Установить адрес токена
11. function setRS(address RS\_Addr) public restricted {  
    Установить адрес RatingSystem
12. function setComission(uint256 newComission) public restricted {  
    Установить комиссию в %
13. function setWallet(address newWallet) public restricted {  
    Установить адрес приема средств
14. function increaseRating(address to, uint256 value) public {

Техническая функция.

**Контракт FundRaising**

1. address public admin;  
   Админ
2. MarketPlace public MP;  
   Адрес продаж
3. uint256 public start;  
   старт
4. uint256 public finish;  
   финиш
5. uint256 public goal;  
   цель
6. uint256 public minProfit;  
   минимальный размер прибыли
7. uint256 public minAmount;  
   минимальный размер инвестиции
8. uint256 public maxAmount;  
   максимальный размер инвестиции
9. uint256 public tokensRaised;  
   колво собранных токенов
10. mapping (address => uint256) public invested;  
    сколько инвестировал адрес
11. function init(address MP\_Addr, uint256 startUNIX, uint256 finishUNIX, uint256 minValue, ….  
    техническая функция, может быть вызвана только из контракта EM
12. function invest(uint256 tokens) public {  
    инвестировать определенное количество токенов
13. function MP\_withdraw(uint256 amount) external {  
    техническая функция, доступна для вызова только из контракта MP
14. function finalize() public {  
    Завершить сбор средств
15. function refund(address addr) external {  
    Вывести доступные средства для определенного адреса (возврат непотраченных средств)

**Контракт MarketPlace**

1. FundRaising public FR;  
   ПСС
2. uint256 public start;  
   старт
3. uint256 public finish;  
   Финиш
4. mapping (address => uint256) public purchased;  
   сколько купил адрес
5. uint256 public tokensRaised;  
   сколько собрано средств
6. uint256 public profit;  
   сколько собрано прибыли
7. function init(address FR\_Addr, uint256 startUNIX, uint256 finishUNIX, uint256….  
   техническая функция, может быть вызвана только из EM
8. function purchase(uint256 tokens) public {  
   Купить на определенное количество токенов
9. function fiatEntrance(uint256 tokens) public {  
   Перевести токены (дублирование фиатных покупок)
10. function finalize() external {  
    Завершить Продажи
11. function takeProfit(address addr) external {  
    Снять дивиденды (для инвесторов)
12. function admin() public view returns(address) {  
    адрес админа