Group Project Bidding Token



Prepared for: Dave McKay, Paul Chafe

By: Shilpa Kaushik, Mohammad Jamshed Qureshi, Andrew Starling

February 8, 2020

# Executive Summary

## Objective

The objective of this assignment is to build a crowdfunding application where projects can be submitted, and multiple parties can vote on projects to be funded.

## Goals

The goal of this project is to build an efficient and trustworthy crowd funding bidding platform, which is straightforward and provides a simple user interface for stakeholders.

## Problem

Individual organizations may not have a large budget, but collectively with other small organizations they can have a large budget. However, getting them to all agree on what to fund is difficult. Organizations have different amounts budgeted for a project. There can be multiple projects to fund. How do you set up a fair vote?

## Solution

Our solution is designed around the concept of trust and openness. It utilizes blockchain technologies like Ethereum, Solidity, and Smart Contracts.

Entities involved in the project are as follows:

* Projects
* Artists (Creators)
* Organizations
* Workflow

# Requirements Gathering

## Goals

* Projects receive funding via their participation in the dApp, offering an innovative form of kickstarting where artists/creators publish a project, and organizations can fund the project.
* Success is measured by the funding threshold reached by individual projects. If a particular project does not reach the funding threshold within an allotted time period, the donations are retrievable by the donor.

## Stakeholders

* includes the project creators and funding organizations.

| State Data | |
| --- | --- |
| Data structure | State {  Fundraising,  Expired,  Successful  } |
| State variables | address payable public creator |
|  | uint public amountGoal  Required to reach at least this much funding, otherwise all contributors receive a refund. |
|  | uint public completeAt |
|  | uint256 public currentBalance |
|  | uint public raiseBy |
|  | string public title |
|  | string public description |
|  | State public state = State.Fundraising  This is initialized on creation of project. |
|  | mapping (address => uint) public contributions |

## Restrictions

* project cannot be funded by the creator of the project
* project must be fully funded by a particular date/time, else donors receive a refund of their donation

## Exceptions

* Rules cannot be broken under any circumstances outside of the programming logic.

## Solidity Smart Contract

| Contract: Crowdfunding.sol | |
| --- | --- |
| Function | Description |
| startProject | Function to start a new project.   * @param title: Title of the project to be created * @param description: Brief description about the project * @param durationInDays: Project deadline in days * @param amountToRaise: Project goal in wei |
| returnAllProjects | Function to get all projects' contract addresses.   * @return A list of all projects' contract addresses |
| contribute | Function to fund a certain project. |
| checkIfFundingCompleteOrExpired | Function to change the project state depending on conditions. |
| payOut | Function to give the received funds to project starter. |
| getRefund | Function to retrieve donated amount when a project expires. |
| getDetails | Function to get specific information about the project. |

## Data Definitions

|  | |
| --- | --- |
| Entity | Description |
| event ProjectStarted | Event emitted when a new project is started.  address contractAddress,  address projectStarter,  string projectTitle,  string projectDesc,  uint256 deadline,  uint256 goalAmount |
| Project[] | Array of projects |
| constructor | address payable projectStarter,  string memory projectTitle,  string memory projectDesc,  uint fundRaisingDeadline,  uint goalAmount |
| function getDetails | Returns all of the project's details.  address payable projectStarter,  string memory projectTitle,  string memory projectDesc,  uint256 deadline,  State currentState,  uint256 currentAmount,  uint256 goalAmount |

## Project Plan

| Task | Start | End | Duration | Assigned To |
| --- | --- | --- | --- | --- |
| Design: Architecture | 01-31-2020 | 01-31-2020 | 1 day | Shilpa, Jamshed, Andrew |
| Development: Smart Contracts | 02-01-2020 | 02-03-2020 | 3 days | Jamshed, Andrew |
| Development: UI | 02-04-2020 | 02-06-2020 | 3 days | Jamshed, Shilpa, Andrew |
| Testing: Smart Contracts / Truffle | 02-03-2020 | 02-03-2020 | 1 day | Jamshed |
| Testing: UI | 02-04-2020 | 02-05-2020 | 1 day | Shilpa, Andrew |
| Deploy | 02-08-2020 | 02-08-2020 | ½ day | Shilpa |