pragma solidity ^0.5.5;

import "./PupperCoin.sol";

import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/release-v2.5.0/contracts/crowdsale/Crowdsale.sol";

import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/release-v2.5.0/contracts/crowdsale/emission/MintedCrowdsale.sol";

import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/release-v2.5.0/contracts/crowdsale/validation/CappedCrowdsale.sol";

import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/release-v2.5.0/contracts/crowdsale/validation/TimedCrowdsale.sol";

import "https://github.com/OpenZeppelin/openzeppelin-contracts/blob/release-v2.5.0/contracts/crowdsale/distribution/RefundablePostDeliveryCrowdsale.sol";

// @TODO: Inherit the crowdsale contracts

contract PupperCoinSale is Crowdsale, MintedCrowdsale, CappedCrowdsale, TimedCrowdsale, RefundablePostDeliveryCrowdsale {

constructor(

// @TODO: Fill in the constructor parameters!

PupperCoin Coin , // name of the token

uint rate, // rate in PupperCoins

address payable wallet, // sale beneficiary

uint goal, //goal for crowdsale

uint starttime,

uint endtime // uint time Cap

)

Crowdsale(rate, wallet, Coin)

TimedCrowdsale(starttime,endtime)

CappedCrowdsale(goal)

RefundableCrowdsale(goal)

// @TODO: Pass the constructor parameters to the crowdsale contracts.

public

{

// constructor can stay empty

}

}

contract PupperCoinSaleDeployer {

address public Coin\_sale\_address;

address public Coin\_address;

constructor(

address payable wallet, // sale beneficiary

string memory name,

string memory symbol,

uint initial\_supply

// @TODO: Fill in the constructor parameters!

)

public

{

// @TODO: create the PupperCoin and keep its address handy

PupperCoin Coin = new PupperCoin(name, symbol, 0);

Coin\_address = address(Coin);

// @TODO: create the PupperCoinSale and tell it about the token, set the goal, and set the open and close times to now and now + 24 weeks.

PupperCoinSale Coin\_sale = new PupperCoinSale(Coin, 1, wallet, initial\_supply, now, now + 24 weeks);

Coin\_sale\_address = address(Coin\_sale);

// make the PupperCoinSale contract a minter, then have the PupperCoinSaleDeployer renounce its minter role

Coin.addMinter(Coin\_sale\_address);

Coin.renounceMinter();

}

}