Documentation (Solidity)

Μεταβλητές

- **`owner`**: Διεύθυνση του ιδιοκτήτη του συμβολαίου.
- **`campaignFee`**: Το κόστος δημιουργίας εκστρατείας (0.02 ether).
- **`platformFees`**: Συνολικά κέρδη της πλατφόρμας.
- **`coOwner`**: Συνιδιοκτήτης της πλατφόρμας.
- **`isDestroyed`**: Καθορίζει αν το συμβόλαιο έχει καταστραφεί.

Δομή Campaign

- **`campaignId`**: Αναγνωριστικό της εκστρατείας.
- **`entrepreneur`**: Δημιουργός της εκστρατείας.
- **`titles`**: Τίτλος της εκστρατείας.
- **`pledgeCost`**: Κόστος συμμετοχής (ανά pledge).
- **`pledgesNeeded`**: Απαιτούμενος αριθμός pledges.
- **`pledgesCount`**: Πλήθος pledges που έχουν γίνει.
- **`fulfilled`**: Αν έχει ολοκληρωθεί η εκστρατεία.
- **`cancelled`**: Αν έχει ακυρωθεί η εκστρατεία.
- **`backers`**: Λίστα επενδυτών (backers).
- **`shares`**: Μερίδια για κάθε επενδυτή.
- **`investments`**: Επενδύσεις για κάθε επενδυτή.
- **`investors`**: Λίστα επενδυτών που έχουν συμμετάσχει.

Άλλες Μεταβλητές

- **`campaignCount`**: Αριθμός συνολικών εκστρατειών.
- **`feesCollected`**: Αριθμός συλλεγμένων τελών
- **`bannedEntrepreneurs`**: Boolean λίστα αποκλεισμένων επιχειρηματιών.
- **`bannedEntrepreneursList`**: Λίστα διευθύνσεων αποκλεισμένων επιχειρηματιών.

Modifiers

Χρησιμοποιήθηκαν modifiers για όλους τους ελέγχους (requires) που απαιτούνται για τη διασφάλιση της ορθής λειτουργίας του συμβολαίου. Αυτά περιλαμβάνουν ελέγχους

εξουσιοδότησης (**onlyOwner**, **onlyEntrepreneur**, **onlyAuthorized**), εγκυρότητα δεδομένων (**campaignExists**, **isPledgeValid**, **canCompleteCampaign**, **hasInvestement**), κατάσταση του συμβολαίου (**contractActive**), κατάσταση καμπάνιας (**campaignCancelled** ,**campaignNotCancelled**, **campaignNotFulfilled**) απαγόρευση χρηστών (**notBanned**, **notOwner**) έλεχγος τέλους συμβολαίου (**correctFee**).

Συγκεκριμένα:

`contractActive`

- Μια μεταβλητή isDestroyed ως flag να ελέγχεται αν το συμβόλαιο είναι ενεργό. Τον τελεστή ! να εξασφαλίζει ότι το συμβόλαιο δεν έχει καταστραφεί.

`onlyOwner`

- Ο καλών (msg.sender) είναι ο owner του συμβολαίου.
- Ορισμός της owner όπου αποθηκεύεται ως μία μεταβλητή address κατά την αρχικοποίηση του συμβολαίου, διασφαλίζοντας την ταυτοποίηση του ιδιοκτήτη.

`notOwner`

- Μεταβλητή owner με τον τελεστή != για να διασφαλιστεί ότι ο καλών δεν είναι ο ιδιοκτήτης του συμβολαίου (δημιουργία καμπάνιας).

`notBanned`

- Αποτροπή στους αποκλεισμένους επιχειρηματίες από το να αλληλεπιδράσουν με συγκεκριμένες λειτουργίες. Συγκεκριμένα, την δημιουργία καμπάνιας καθώς χρησιμοποιείται σε αυτήν την συνάρτηση.
- Η δομή bannedEntrepreneurs με τον τελεστή ! για να ελέγχεται αν ο καλών **δεν ανήκει** στη λίστα των αποκλεισμένων χρηστών.

`correctFee`

- Έλεγχος αν το ποσό (msg.value) που αποστέλλεται είναι ίσο με το campaignFee.

`campaignExists(uint campaignId)`

- Έλεγχος αν το campaignId είναι μικρότερο από το campaignCount, ώστε να διασφαλιστεί ότι η καμπάνια υπάρχει.
- Η χρήση του campaignCount ως μετρητή καμπανιών εξασφαλίζει οτιδήποτε έξω από το εύρος 0...campaignCount-1 είναι μη έγκυρο.

`campaignNotFulfilled(uint campaignId)`

- Μεταβλητή fulfilled μέσα από τη δομή campaigns με τον τελεστή ! για να ελεγχθεί αν η καμπάνια δεν έχει ολοκληρωθεί.

campaignNotCancelled(uint campaignId)

- Χρήση της μεταβλητής cancelled μέσα από τη δομή campaigns με τον τελεστή ! για να επιβεβαιωθεί ότι η καμπάνια δεν έχει ακυρωθεί.

campaignCancelled(uint campaignId)

- Έλεγχος αν η μεταβλητή cancelled μέσα από τη δομή campaigns για να διασφαλιστεί ότι η καμπάνια έχει ακυρωθεί.

hasInvestment(uint campaignId)

- Χρήση του campaigns για πρόσβαση στη συγκεκριμένη καμπάνια μέσω του campaignId και στη συνέχεια το investments για να ελεγχθεί αν ο msg.sender έχει πραγματοποιήσει επένδυση.
- Η επαλήθευση βασίζεται στο ότι το ποσό της επένδυσης πρέπει να είναι μεγαλύτερο από 0.

isPledgeValid(uint campaignId, uint shares)

- Έλεγχος αν το ποσό (msg.value) είναι ίσο με τον αριθμό των μετοχών (shares) επί το κόστος ανά μετοχή (pledgeCost).

canCompleteCampaign

- Λαμβάνω τα δεδομένα της καμπάνιας με τη χρήση του `Campaign storage campaign = campaigns[campaignId]`.
- `!campaign.cancelled`: Ελέγχει ότι η καμπάνια δεν έχει ακυρωθεί. Μια ακυρωμένη καμπάνια δεν μπορεί να ολοκληρωθεί.
- `!campaign.fulfilled`: Εξασφαλίζει ότι η καμπάνια δεν έχει ήδη ολοκληρωθεί για να αποτρέψει διπλή ολοκλήρωση.
- `campaign.pledgesCount >= campaign.pledgesNeeded`: Βεβαιώνω ότι η καμπάνια έχει λάβει τον ελάχιστο απαιτούμενο αριθμό από pledges (υποσχέσεις). Μόνο οι καμπάνιες που εκπληρώνουν τους στόχους τους μπορούν να ολοκληρωθούν.

noPlatformFees

- Χρησιμοποιώ τη μεταβλητή `platformFees`, η οποία παρακολουθεί τα συσσωρευμένα τέλη της πλατφόρμας από τις καμπάνιες.
- Η συνθήκη `platformFees == 0` διασφαλίζει ότι όλα τα τέλη έχουν ήδη συλλεχθεί από τον ιδιοκτήτη του συμβολαίου.

onlyAuthorized(uint campaignId)

- Ελέγχω αν ο καλών (`msg.sender`) είναι είτε ο ιδιοκτήτης (`owner`) είτε ο `entrepreneur` της καμπάνιας.

`alreadyBanned(address _entrepreneur)`:

- Χρησιμοποίησα τη δομή `bannedEntrepreneurs` με τον τελεστή `!` για να διασφαλιστεί ότι ο επιχειρηματίας δεν έχει ήδη αποκλειστεί.

Δεν υπάρχουν προβλήματα λειτουργικότητας, οι modifiers εφαρμόστηκαν με επιτυχία.

ΣΥΝΑΡΤΗΣΕΙΣ ΛΕΙΤΟΥΡΓΕΙΩΝ ΕΠΙΧΕΙΡΗΜΑΤΙΑ

Υλοποιήθηκαν οι συναρτήσεις `createCampaign`, `cancelCampaign`, και `completeCampaign`. Η δημιουργία καμπάνιας απαιτεί την καταβολή τέλους (0.02 ether το οποίο οριστηκοποιείται από την μεταβλητή campaignFee) για την δημιουργία της, ένα όνομα, το κόστος μετοχών και το επιθυμιτό αριθμό μετοχών. Κατά την δημιουργία κάθε καμπάνιας αυξάνω μια μεταβλητή counter για να παρακολουθώ τον αριθμό των καμπανιών και προσθέστω την καμπάνια στην **struct λίστα campaigns.**

Η ακύρωση ακυρώνει την καμπάνια θέτοντας τη **campaign.cancelled = true** και εμποδίζει περαιτέρω επενδύσεις και μπλοκάρει την complete αν κάποιος την καλέσει (για την συγκεκριμένη καμπάνια), και η ολοκλήρωση αποδίδει τα έσοδα στον επιχειρηματία μετά από τις πράξεις που πραγματοποιούνται όταν οι επενδύσεις ικανοποιήσουν τον επιθυμήτο αριθμό μετοχών και μπλοκάρει την cancel αν κάποιος την καλέσει (για την συγκεκριμένη καμπάνια).

Στην complete τίθεται η **campaign.fulfilled = true** και αυξάνω τα έσοδα του συμβολαίου (platformFees). Η ακύρωση καμπάνιας μετά από τη μερική χρηματοδότηση **δεν επιστρέφει αυτόματα** τα χρήματα στους επενδυτές, αλλά απαιτεί να καλέσουν οι ίδιοι τη `refundInvestor` **χωρίς την συμπλήρωση** κάποιου ID, αφού καλώντας την refund οι επενδυτές λαμβάνουν τα χρήματα τους για **όλες τις ακυρωμένες** καμπάνιες στις οποίες έχουν επενδύσει.

Υλοποιήθηκαν πλήρως όλες οι λειτουργίες, και οι πληρωμές λειτουργούν με επιτυχία.

ΣΥΝΑΡΤΗΣΕΙΣ ΛΕΙΤΟΥΡΓΕΙΩΝ ΕΠΕΝΔΥΤΩΝ

Υλοποιήθηκαν οι συναρτήσεις `contributeToCampaign` και `refundAllInvestments`. Η πρώτη. απαιτεί το ID της καμπάνιας που θέλουμε να επενδύσουμε, τον αριθμός των μετοχών και την **τιμή κάθε μετοχής * τον αριθμό των μετοχών,** καταγράφοντας έτσι τις επενδύσεις στην λίστα investments και ενημερώνει τις μετοχές (pledges), και την λίστα των επενδυτών.

Ενώ η δεύτερη επιστρέφει κεφάλαια επενδυτών σε περίπτωση ακύρωσης καμπάνιας. Ο επενδυτής μπορεί να καλεί την `refundAllInvestments` **μόνο** μετά την ακύρωση μιας καμπάνιας για την επιστροφή των χρημάτων του από **όλες τις ακυρωμένες** καμπάνιες στις οποίες έχουν επενδύσει. Αυτό πραγματοποιείται από την for loop που διασχίζει όλες τις καμπάνιες, αν δεν υπάρχει επένδυση στην συγκεκριμένη καμπάνια ή δεν είναι ακυρωμένη τότε την **παραλείπει**, αλλιώς λαμβάνει την επένδυση και την προσθέτει στην συνολική τιμή των επιστροφών. Επίσης, έχω προσθέσει ένα `require(totalRefund > 0, "No refundable investments")` ώστε να ελέγχει αν γενικά έχει επενδύσει όταν καλείτε η refund.

Οι μεταφορές χρημάτων πραγματοποιούνται σωστά, εξίσου και οι άνω συναρτήσεις.

ΣΥΝΑΡΤΗΣΕΙΣ ΛΕΙΤΟΥΡΓΕΙΩΝ ΙΔΙΟΚΤΗΤΗ

Υλοποιήθηκαν οι συναρτήσεις `collectPlatformEarnings`, `banEntrepreneur`, `changeOwner`, και `destroyContract`. Η `collectPlatformEarnings` λαμβάνει τα έσοδα του συμβολαίου. Αρχικοποιώ την `totalEarnings` με τα έσοδα που έχουν προκύψει, μέσω μιας for διασχίζεται η λίστα των campaigns και για κάθε μία καταχωρώ μέσω της `feesCollected[i]` τις προμήθειες της καμπάνιας ως συλλεχθέντα, και τέλος μηδενίζω την `platformFees`. Έχω προσθέσει **τον modifier onlyOwner** ώστε να μπορεί να την καλέσει μόνο ο δημιουργός του contract. Επίσης, έχω προσθέσει το `require(totalEarnings > 0, "No funds to collect")`, ώστε να καλείτε μόνο αν υπάρχουν χρήματα να συλεχθούν ώστε να μην **ξοδεύεται λανθασμένα** gas.

Η `banEntrepreneur` απαγορεύει σε κάποιον χρήστη την λειτουργία της δημουργίας καμπάνιας. Αυτό πραγματοποιείται από την εισαγωγή της διεύθυνσης του χρήστη,

κάτι που υλοποιείται μόνο από τον owner του συμβολαίου όπως φαίνεται και από **τον modifier onlyOwner**. Έχοντας την διεύθυνση του, τον εισάγω στην λίστα bannedEntrepreneursList και θέτοντας το flag bannedEntrepreneurs[_entrepreneur] = true. Επίσης μέσω μίας for όσες καμπάνιας ανήκουν στον απεκλεισμένο επιχειρηματία **ακυρώνονται θέτοντας** campaign.cancelled[i] = true.

Στην `changeOwner` δίνοντας την διέυθυνση ενός χρήστη **μεταβιβάζεται** η ιδιοκτησία του συμβολαίου σε αυτόν. Ουσιαστικά ορίζοντας την μεταβλητή owner με την διεύθυνση του χρήστη.

Τέλος η `destroyContract,` καταστρέφει το συμβόλαιο και η μόνη λειτουργεία που είναι διαθέσιμη, είναι η `refundInvestor.` Αυτό συμβαίνει με την global μεταβλητή **isDestroyed = true,** όπου χρησιμποιείται στον modifier contractActive για να **μπλοκάρει** τις συναρτήσεις που τον χρησιμοποιούν. Επίσης, μέσω μίας for διασχίζοντας την λίστα των campaings, θέτω όλες τις campaign.cancelled με true ώστε να μπορούν οι επενδυτές να χρησιμοποιήσουν την refundInvestor. Και τέλος, όταν η **το balance φτάσει να ισούται με 0** (έχουν λάβει όλοι επενδυτές τα χρήματα τους πίσω) το συμβόλαιο καταστρέφεται ολοκληρωτικά.

Υλοποιήθηκαν πλήρως χωρίς κάποιο πρόβλημα λειτουργικότητας.

ΒΟΗΘΗΤΙΚΕΣ ΣΥΝΑΡΤΗΣΕΙΣ (GETTERS)

Στις βοηθητικές συναρτήσεις υλοποιήθηκαν οι `getActiveCampaigns`, `getCompletedCampaigns`, `getCancelledCampaigns`, `getInvestors,` `getBannedEntreprenuers` και `getContractFees` για τη λήψη δεδομένων και κατάστασης καμπάνιας και επενδυτών.

Όσον αφορά τις 3 πρώτες, λειτουργούν με τον ίδιο τρόπο. Για τον λόγο οτι η έκδοση 0.5.9 της solidity δεν επιτρέπει επιστροφή του struct, αποφάσισα να το υλοποιήσω με την **δημιουργία πινάκων**, ανάθεση των τιμών του struct σε αυτές και επιστροφή αυτών. Π.χ. η `getCancelledCampaigns.` Πρώτα, η συνάρτηση διατρέχει όλες τις καμπάνιες (από το 1 μέχρι το campaignCount), και ελέγχει αν είναι ακυρωμένες και αν δεν έχουν εκπληρωθεί `(!campaigns[i].fulfilled)`. Αν ισχύει αυτό, αυξάνει τον μετρητή **count**. Δημιουργεί πίνακες (arrays) για να αποθηκεύσει τα δεδομένα των ακυρωμένων καμπανιών. Οι πίνακες δημιουργούνται με το μέγεθος που έχει υπολογιστεί στο προηγούμενο βήμα (δηλαδή τον αριθμό των ακυρωμένων καμπανιών). Ξαναδιατρέχει τις καμπάνιες και, για κάθε ακυρωμένη και μη

εκπληρωμένη καμπάνια, αποθηκεύει τα δεδομένα της στους **αντίστοιχους πίνακες**. Τέλος, επιστρέφει όλους τους πίνακες με τα δεδομένα που έχει αποθηκεύσει για τις ακυρωμένες καμπάνιες.

Όσον αφορά την `getInvestors` εισάγωντας το ID της καμπάνιας, επιστρέφει τους επενδυτές και τις μετοχές που κατέχουν στην συγκεκριμένη καμπάνια. Αυτό επιτυγχάνεται μέσω της for, όπου διασχίζοντας την λίστα των επενδυτών μιας συγκεκριμένης καμπάνιας, προσθέτει σε μία άλλη λίστα όσους ανήκουν στην καμπάνια, και επιστρέφονται οι διευθύνσεις τους και οι μετοχές τους σε αυτές.

Στην `getContractFees` δημιουργείται μαι τοπική μεταβλητή η οποία λαμβάνει τα χρήματα που έχουν μαζευτεί από την `platformFees` και απλά επιστρέφει την τιμή της. Ωστόσο, αυτή η συνάρτηση λαμβάνει τα χρήματα μιας **δεδομένης στιγμής**, δηλαδή αν ο owner κάνει **ανάληψη**, τότε η **συνάρτηση θα επιστρέψει 0** καθώς στην `collectPlatformEarnings` **μηδενίζονται τα έσοδα** της πλατφόρμας.

Η `getContractBalance` επιστέφει το balance.

Η `getName` επιστέφει το όνομα μιας συγκεκριμένης καμπάνιας από το struct campaign.

Η `getDestroyed` επιστέφει την boolean μεταβλητή isDestroyed για τον έλεγχο αν το συμβόλαιο έχει καταστραφή.

Η `getShares` επιστρέφει τα μερίδια ενός επενδυτή σε συγκεκριμένη εκστρατεία μέσω της των shares του struct campaigns.

Τέλος στην `getBannedEntreprenuers` επιστρέφεται η λίστα `bannedEntrepreneursList`, στην οποία έχουν προστεθεί οι επιχειρηματίες από την `banEntrepreneur` συνάρτηση.

Υλοποιήθηκαν όλα τα getters πλήρως χωρίς κάποιο πρόβλημα λειτουργικότητας.

Γεγονότα (Events)

- **`CampaignCreated`**: Δημιουργία νέας εκστρατείας.
- **`InvestmentMade`**: Καταγραφή επένδυσης.
- **`CampaignCancelled`**: Ακύρωση εκστρατείας.
- **`CampaignCompleted`**: Ολοκλήρωση εκστρατείας.

- **`InvestorRefunded`**: Επιστροφή χρημάτων σε επενδυτή.
- **`EntrepreneurBanned`**: Αποκλεισμός επιχειρηματία.
- **`CollectPlatformEarnings`**: Ανάληψη κερδών της πλατφόρμας.
- **`DestroyContract`**: Καταστροφή του συμβολαίου.

Υλοποιήθηκαν πλήρως χωρίς κάποιο πρόβλημα λειτουργικότητας.

Εγχειρίδιο / Οδηγός Χρήσης (Solidity)

Ο επιχειρηματίας μπορεί από την συνάρτηση **createCampaign** να δημιουργήσει μια καμπάνια δίνοντας έναν τίτλο, κόστος μετοχής, απαιτούμενος αριθμός για ολοκλήρωση. Με την **contributeToCampaign** δίνοντας το ID της καμπάνιας και τις μετοχές που επιθύμει ο ίδιος, είτε ένας άλλος επιχειρηματίας μπορεί να επενδύσει σε αυτήν. Ο επιχειρηματίας δημιουργός καμπάνιας έιτε ιδιοκτήτες συμβολαίου μπορούν δίνοντας το ID της καμπάνιας να την ακυρώσουν μέσω της **cancelCampaign** είτε **completeCampaign** για ολοκλήρωση εφόσον αυτή τηρεί τον απαιτούμενο αριθμό μετοχών.

Οι επιχειρηματίες που έχουν επενδύσει σε ακυρωμένη καμπάνια με το **refundAllInvestments** μπορούν να λάβουν τα χρήματα τους πίσω.

Οι ιδιοκτήτες του συμβολαίου μπορούν να απαγορεύσουν έναν επιχρειματία από το να δημιουργήσει κάποια καμπάνια μέσω της **banEntrepreneur** δίνοντας την διεύθυνση του. Επίσης, οι ιδιοκτήτες μπορούν να αναθέσουν την ιδιοκτησία του συμβολαίου σε άλλον επιχειρηματία μέσω της **changeOwner** δίνοντας την διεύθυνση του. Επιπλέον, αυτοί μπορούν να καταστέψουν το συμβόλαιο μέσω της **destroyContract** αφού προηγουμένος έχουν κάνει ανάληψη των χρήματων τους μέσω της **collectPlatformEarning**.

Υπάρχουν και οι βοηθητικές συναρτήσεις των getters, ώστοσο μπορούν να χρησιμοποιηθούν μόνο στο Remix καθώς στο Dapp η χρήση τους γίνεται αυτόματα για την αναπαράσταση των δεδομένων. Οι λειτουργίες του αναφέρθηκαν στο τμήμα του Documentation στις βοηθητικες συναρτήσεις πιο πάνω,

Σχεδιαστικές Αποφάσεις (Solidity)

Αρχικά για ευκολία προσπέλασης, ανάθεσα ως global μεταβλήτες την owner και coOwner ώστε να είναι εύκολο από τους modifiers να κάνουν τους απαιτούμενους ελέγχους. Επιπρόσθετα, στο ίδιο μοτίβο η CampaignFee και platformFees ώστε όταν συμβαίνει μια αλλαγή με κάποια δημιουργία κάμπανιας ή επένδυση να υπάχει απευθείας ενημέρωση, το ίδιο πραγματοποίησα και με την isDestroyed για ευκολία στην προσπέλαση από τους modifiers.

Από τα πιο σημαντικά η υλοποίηση του struct Campaign με ανάθεση mapping (investors) για την καταραφεί των επνδυτών και (investments) των επενδύσεων. Boolean μεταβλητές τύπου flag για το εάν είναι ολοκληρωμένη, ενεργή ή ακυρωμένη (fulfiled και canceled) ή καμπάνια. Και τα υπόλοιπα στοιχεία id, title κτλπ.

Εξωτερικά του struct ένα mapping που θα αποθηκεύει όλες τις καμπάνιες. Δεν δημιούργησα κάποια λίστα για ακυρωμένες καμπάνιες και ολοκληρωμένες, καθώς στις συναρτήσεις για ολοκλήρωση και ακύρωση αλλάζω το εκάστοτε flag σε true. Και όταν είναι επιθυμητή η αναζήτηση τους, οι αντίστοιχες συνάρτησεις ελέγχουν αν το fulfilled ή το canceled αντίστοιχα είναι true.

Για να γνωρίζω πόσες καμπάνιες υπάρχουν αυξάνω έναν μετρητή στην συνάρτηση δημιουργίας καμπάνιας (createCampaign) και σε κάθε επένδυση έναν μετρητή που ανήκει στο struct για την να γνωρίζω πόσες μετοχές έχουν αγοραστεί για να χρησιμοποιείται στις αριθμητικές πράξεις για την κατανομή των χρημάτων όταν συμβεί κάποια ολοκλήρωση καμπάνιας, στα loops για την ανάρτηση των δεδομένων ώστε να γνωρίζει ο αλγόριθμος πόσες καμπάνιες να διατρέξει