DocCoin
The best way to be healthy

Description of functions of smart contract DocCoin

DocCoin Escrow Contract

Escrow is an Ethereum contract for DocCoin. The main functionality it provides are:
- creating and storing a deal between client and consultant
- managing states of said deals
- temporal storage of funds (in coind), provided by client
- arbitership and transfering funds to one of the parties depending on the arbitership outcome

Structures and Enumerations

Deal Structure
Deal structure contains information about signular deal.

Field	Type	Description
patient	address	Address of patient's wallet
doctor	address	Address of doctor's wallet
amount	uint256	Amount of coin required from client
feeAmount	uint256	Amount of fee paid to the service
feeCalculationMethod	FeeCalculationMethod	Method of fee calculation
completionConfirmedByDoctor	bool	Shows if the consultant confirmed the deal
completionConfirmedByPatient	bool	Shows if the client confirmed the deal
requiresArbitration	bool	Shows if the deal requires arbitration
disputeResult	ArbitratorDecision	Result of arbitration
state	DealState	Inner state of the deal

DealState Enumeration
Represents the inner state of the deal

Field	Description
Placed	Deal is just created
AcceptedByDoctor	Deal is accepted by consultant
ConfirmedByPatient	Deal is confirmed by client
Disputed	Deal is currently being disputed by client
Closed	Deal is closed (either there was no dipute, or dispute was resolved)

ArbitratorDecision Enumeration
Represents possible decisions of arbitrator

Field	Description
None	Deal is not resolved
Patient	Deal is resolved pro client
Doctor	Deal is resolved pro consultant

FeeCalculationMethod Enumeration
Represents possible decisions of arbitrator

Field	Description
PlusPercent	Fee is calculated as percent of amount (percent is stored in the contact) and is not included in the amount
MinusPercent	Fee is calculated as percent of amount (percent is stored in the contact) and is included in the amount
PlusFixedAmount	Fee is taken from a value stored in the contract as is and is not included in the amount
MinusFixedAmount	Fee is taken from a value stored in the contract as is and is included in the amount

API

Only from owner

constructor(address _coinAddress, address _feeAccount, uint256 _feeAmount, FeeCalculationMethod _feeCalculationMethod)

Creates contract

* `_coinAddress` - address of the coin used with the contract
* `_feeAccount` - address of the fee account
* `_feeAmount` - initial amount of the fee
* `_feeCalculationMethod` - initial method of fee calculation

setFeeAccount(address _feeAccount) public onlyOwner
Changes address of the fee account
* `_feeAccount` - new fee account's address

setFeeCalculationParameters(uint256 _feeAmount, FeeCalculationMethod _feeCalculationMethod) public onlyOwner
Changes fee amount and fee calculation method
* `_feeAmount` - new amount of fee or percent
* `_feeCalculationMethod` - new method of fee calculation

withdrawFee(address _to) public onlyOwner
Withdraw fee to account
* `_to` - address of account to withdraw fee to

placeDeal(bytes16 _dealId, address _patient, address _doctor, uint256 _amount) public onlyOwner returns (uint256 _userExpense)
Creates new deal. Emits `DealPlaced` event

* `_dealId` - id of the new deal. Should be unique
* `_patient` - address of the participating client
* `_doctor` - address of the participating consultant
* `_amount` - amount to be paid by client (with or without fee depending on `feeCalculationMethod`)

removeDeal(bytes16 _dealId) public onlyOwner
Removes deal from the contract after it was closed
* `_dealId` - id of the deal

Only from client
getNumberOfSuccessfulDeals() view public onlyOwner
Returns number of successful deals
getNumberOfOpenDeals() view public onlyOwner
Returns number of open deals
getNumberOfDisputedDealsProDoctor() view public onlyOwner
Returns number of disputed deals that were resolved pro consultant
getNumberOfDisputedDealsProPatient() view public onlyOwner
Returns number of disputed deals that were resolved pro client
getSumAmountOfSuccessfulDeals() view public onlyOwner
Returns sum of amounts for all successful deals
getSumAmountOfDisputedDealsProDoctor() view public onlyOwner
Returns sum of amounts for all disputed deals that were resolved pro consultant
getSumAmountOfDisputedDealsProPatient() view public onlyOwner
Returns sum of amounts for all disputed deals that were resolved pro client
getSumAmountOfOpenDeals() view public onlyOwner
Returns sum of amounts for all open deals

confirmDeal(bytes16 _dealId) public
Marks the deal as confirmed by the client (caller of this function) and transfers client's funds to itself and to the fee account. The deal should be marked as accepted by the consultant beforehand and the client should approve the transfer of (amount + fee) funds before calling this function. Emits `DealConfirmed` event
* `_dealId` - id of the deal

confirmDealCompletionByPatient(bytes16 _dealId, bool _completed) public
Sets the deal's completion status from the point of view of the client. If client refuses to mark the deal as completed, the deal becomes disputed, otherwise funds are transfered to the consultant. Client should be the caller of this function
* `_dealId` - id of the deal
* `_completed` - completion status (_true_ - deal is completed)

Only from consultant

acceptDeal(bytes16 _dealId) public
Marks the deal as accepted by the consultant (caller of this function). The deal should be only created. Emits `DealAccepted` event
* `_dealId` - id of the deal

confirmDealCompletionByDoctor(bytes16 _dealId, bool _completed) public
Sets the deal's completion status from the point of view of the consultant. Consultant should be the caller of this function
* `_dealId` - id of the deal
* `_completed` - completion status (_true_ - deal is completed)

Only from arbitrator

resolveDispute(bytes16 _dealId, bool _patientWon) onlyArbitrator public
Resolves dispute and transfer the amount, stored in the contract to the client or to the consultant, depending on the outcome
* `_dealId` - id of the deal
* `_patientWon` - outcome of the arbitership (_true_ - deal is resolved pro patient nad amount is transfered back to him)