DeTrust

Smart Contracts with Community-led Dispute Resolution

Introduction

Contents

System Overview

Motivations and Objectives

Market gaps	Objectives
Limited Flexibility	Empower users to self-manage peer-to-peer agreements with flexible smart contracts
Reliance on centralised authority	Build a trusted ecosystem with transparent, community-led protocols

System Overview

Key Components

Dispute Resolution Mechanism (DRM)

Resolves disputes through decentralised, community-led approach

Smart Contracts

Facilitates peer-to-peer micro-agreements, making them affordable and accessible

Token (DTR)

Drives economic activity, incentivising engagement and investment into the system

Trust Score

Basis of trust that underpins entire system, incentivising trustworthy behaviour

System Overview

Trust Score

- **Conventional**: On a scale of 0-5
- **Neutral start**: Each user starts with 2.5/5
- **Tier-based**: Tiers influence many aspects
- **Calibration**: Increasing difficulty per tier
- **Trust staking**: Trust can be staked to initiate disputes or participate in voting

Tier	Trust Score	Description
A	4.5 - 5.0	Highly Trusted
B	3.0 - 4.49	Trusted
C	2.0 - 2.99	Neutral
D	0 - 1.99	Untrusted

System Overview

Token System

- **In-app currency**: Native token (DTR)
- **Backed by Ether**: 1 DTR ≈ 0.00001 ETH ≈ 0.0217 SGD
- **Signup Fee**: New users pay 10 SGD ≈ 460 DTR + 40 DTR (Bonus)
- **Sustainable economy**: Use DTR to create contracts, earn DTR by verifying them

System Overview

Smart Contracts

- **Flexibility**: ‘Smart’ VS general
  - Prioritise flexibility
- **Legitimacy**: What constitutes a ‘real’ contract?
  - Require actual financial investment
  - Develop contract verification mechanisms
- **Accountability**: How do we enforce outcomes?
  - Enable enforceability in terms of transactional outcomes
  - Rely on Trust and DRM to enforce general outcomes

System Overview

Dispute Resolution Mechanism

- **Dispute initiation**: Stake trust and state desired outcome
- **Community-led voting**: Community votes for the outcome, with rewards for those who vote for the majority
- **High stakes**: Parties incentivised to settle; voters incentivised to vote judiciously
- **Protocol versioning**: Allows for refinements and iterations to protocol

Challenges

- **Soundness of DRM protocol:** Is the voting system fair and sound?
- **User engagement:** Initial high user participation; subsequent drop-off
- **Further complications:** What if outcomes are unsatisfactory?

App Demo

App Demo

Key User Functions

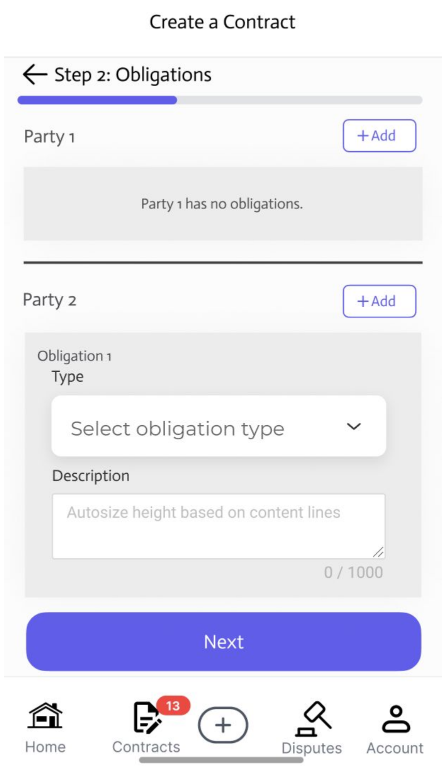
1. Create smart contract
2. Manage smart contracts
   a. View contract details
   b. Chat with the other party in the contract
3. Create dispute
4. Manage disputes
   a. View dispute details
   b. Add user’s side to the dispute
5. View unverified contract details and verify or report contract
6. View disputed contract details and vote

Smart Contracts

Smart Contracts

Common Contract

- Assumption of one initiator to one respondent
- A **customisable** contract which facilitates transactions of a smaller scale
- How do we enforce contract outcomes?
  - Defining **obligations** for both parties
  - Defining **payment terms** for both parties


Create a Contract

Step 2: Obligations

Party 1 +Add

Party 1 has no obligations.

Party 2 +Add

Obligation 1

Type

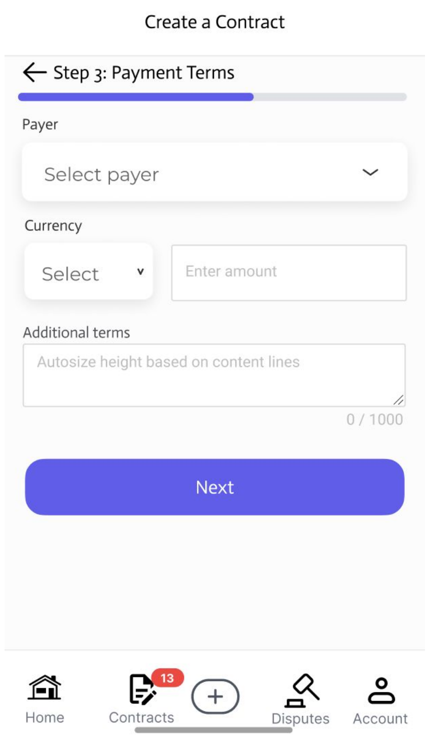
Select obligation type

Description

Autosize height based on content lines

0 / 1000

Next


Create a Contract

Step 3: Payment Terms

Payer

Select payer

Currency

Select

Enter amount

Additional terms

Autosize height based on content lines

0 / 1000

Next

Smart Contracts

Key Features

- Contract Creation
- Contract Signing
- Contract Verification
- Chat Interface
- Event History
- Contract Variations

Smart Contracts

Contract Creation

- **Contract Creation Fee:** Requires both the Initiator and Respondent to pay a fee corresponding to their respective trust score tiers for contract creation
- Contract remains in draft status when both Initiator and Respondent has not paid their contract creation fee and have not signed the contract
- Users cannot create contracts if they are involved in any disputes
- **Rationale:** Prevents users from creating many contracts for fun just to increase their trust scores

Tier	Contract Creation (DTR)	Contract Completion (Trust)
A	20	+ 1
B	40	+ 5
C	80	+ 10
D	100	+ 15

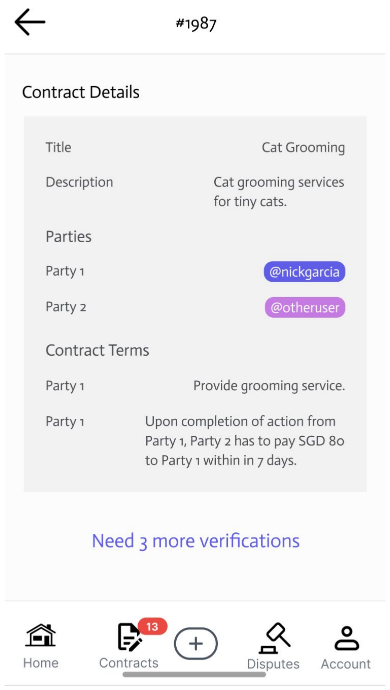
Contract Verification

- **Rationale:** Prevents users from creating fraudulent contracts
- Encourage decentralisation by allowing any users from the community to verify contracts

Tier	Number of verifiers required	Contract Verification (DTR)	Penalty for False Verification
A	4	+ 10	Parties: - 2 Trust, DTR burned Verifiers: - 1 Trust, DTR - 100
B	8		
C	10		
D	20		

Smart Contracts

Contract Verification



#1987

Contract Details

Title	Cat Grooming
Description	Cat grooming services for tiny cats.

Parties

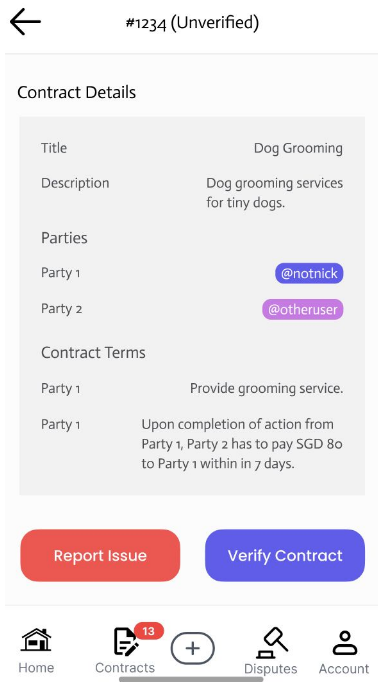
Party 1	@nickgarcia
Party 2	@otheruser

Contract Terms

Party 1	Provide grooming service.
Party 1	Upon completion of action from Party 1, Party 2 has to pay SGD 80 to Party 1 within in 7 days.

Need 3 more verifications

Contract owners' view



#1234 (Unverified)

Contract Details

Title	Dog Grooming
Description	Dog grooming services for tiny dogs.

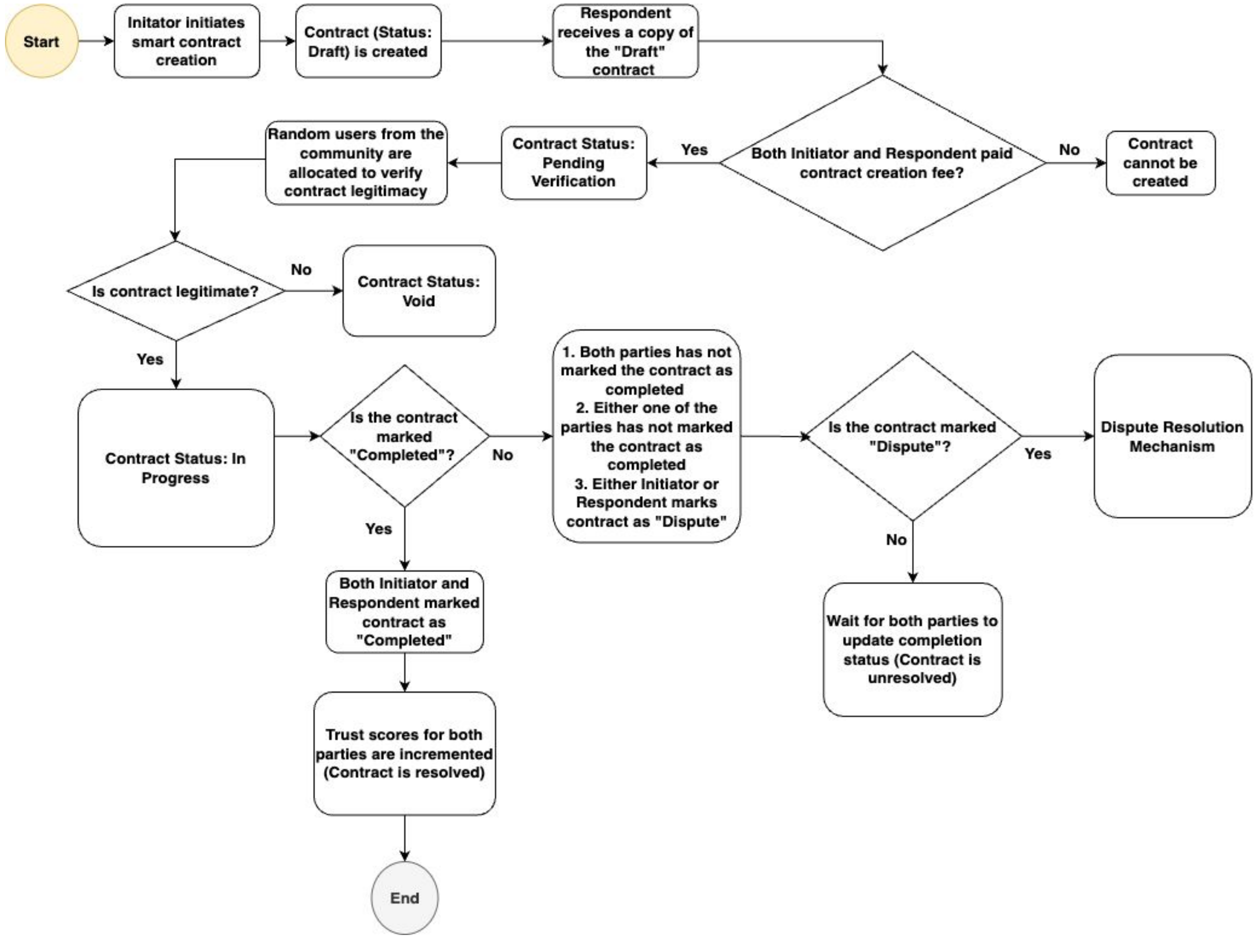
Parties

Party 1	@notnick
Party 2	@otheruser

Contract Terms

Party 1	Provide grooming service.
Party 1	Upon completion of action from Party 1, Party 2 has to pay SGD 80 to Party 1 within in 7 days.

Users' view


Start
Initator initiates smart contract creation
Contract (Status: Draft) is created
Respondent receives a copy of the "Draft" contract
Both Initiator and Respondent paid contract creation fee?
No
Contract cannot be created
Yes
Contract Status: Pending Verification
Random users from the community are allocated to verify contract legitimacy
Is contract legitimate?
No
Contract Status: Void
Yes
Contract Status: In Progress
Is the contract marked "Completed"?
No
1. Both parties has not marked the contract as completed
2. Either one of the parties has not marked the contract as completed
3. Either Initiator or Respondent marks contract as "Dispute"
Is the contract marked "Dispute"?
Yes
Dispute Resolution Mechanism
No
Wait for both parties to update completion status (Contract is unresolved)
Yes
Both Initiator and Respondent marked contract as "Completed"
Trust scores for both parties are incremented (Contract is resolved)
End

Smart Contract Lifecycle

Contract variations

- Financial contracts
- Intellectual Properties contracts
- Purchase agreements
- Service Contracts

Contracts may also have multiple parties, not just one initiator to one respondent.

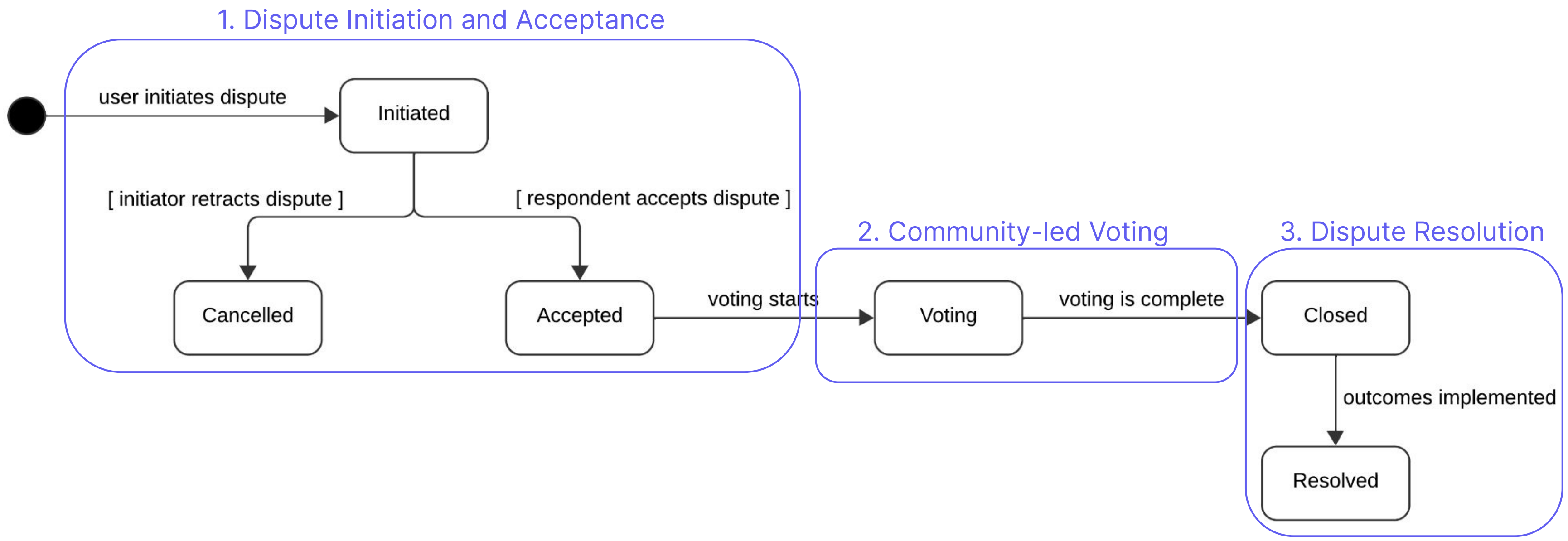
Dispute Resolution Mechanism v1
In-depth

Dispute Resolution Mechanism v1

Key Steps and Process

1. Dispute Initiation and Acceptance
2. Community-led Voting
3. Dispute Resolution

Dispute Resolution Mechanism v1



DRM v1 Statechart - General Flow

Dispute Resolution Mechanism v1

Dispute Initiation and Acceptance

- Either party in an ongoing contract can initiate a dispute
- **Initiator** will input dispute title, description and submit his/her desired outcome
    - Initiator is able to retract the dispute at this stage
- **Respondent** responds by submitting his/her desired outcome
    - 10 DTR will be deducted every 48 hours of failing to respond to the dispute
- Dispute status will be updated to *VOTING* and dispute will be broadcasted to all DeTrust users

Dispute Resolution Mechanism v1

Community-led Voting

- Users between Tiers A - C will be able to view the contract information and event history leading up to the dispute
- They have to stake between 5 - 10 Trust Score depending on their User Tier to vote on their chosen desired outcome

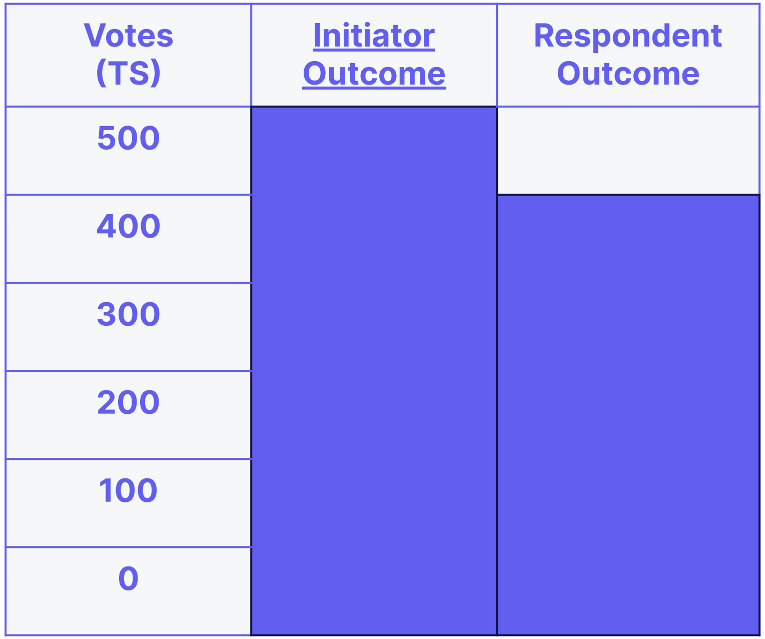
Tier	Minimum Trust Score staked in vote	Maximum Trust Score staked in vote
A	5	10
B	5	8
C	5	
D	Cannot vote!	

Dispute Resolution Mechanism v1

Community-led Voting (cont.)

```
struct VoteSum {
        uint256 time;
        uint256 score;
        uint256 prevScore;
}
```

1. 24-hour minimum voting window
2. Outcome with 500 Trust Score votes win
   a. If both outcomes do not have 500 Trust Score votes, continue voting process for another 24 hours
3. Votes will be counted at the end of 24/48 hours
   a. If majority vote exists, choose that corresponding outcome
   b. If no majority vote, select outcome with **last highest vote count**



Example: Initiator Outcome majority vote win
(Ongoing vote count is not visible to users)

Dispute Resolution Mechanism v1

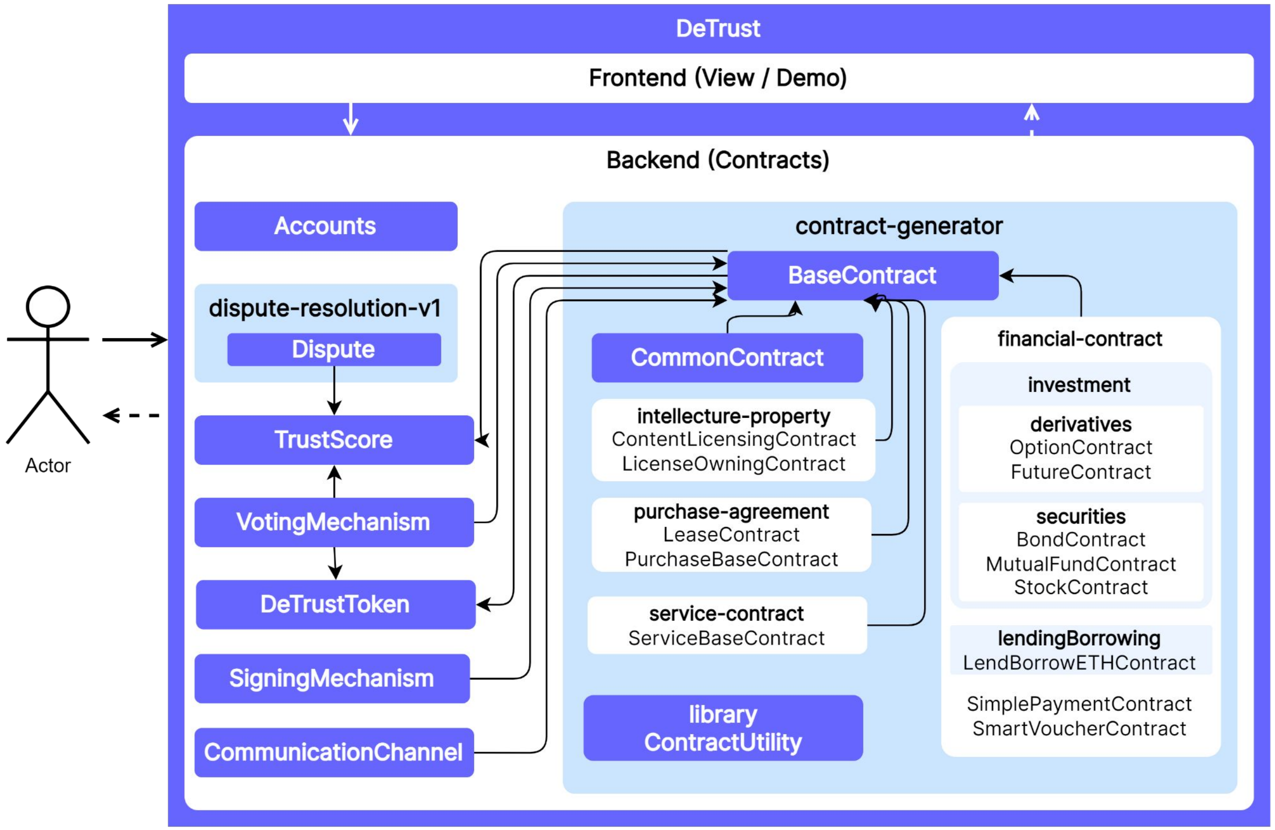
Dispute Resolution

- Deduct 50 Trust Score from the loser of the dispute
    - Significant adjustment of **-0.5** to a user's public Trust Score rating
    - E.g. 430 - 50 → 380, 4.3 - 0.5 → 3.8
- Users who voted correctly gain the amount of trust score staked
- Users who voted incorrectly lose the trust score staked
- Winner of the dispute does not gain or lose any trust score since he/she won the dispute and will get the outcome they submitted

Code Implementation

Code Implementation

Contracts Directory Structure



Code Implementation

Coding Approaches

Error: Stack too deep!

- Too much local variables - `contract` needs to clarify many details
- Use struct!
- UniswapV2Pair: Block scoping! = Placing the piece of code into a separate function

Struct:

Before:

```
constructor CommonContract (
      BaseContract _base,
      address payable[] _payers,
      address payable[] _payees,
      ...
      uint256 _totalObligations,
      ContractUtility.DisputeType _dispute
  ) { ... }
```

After:

```
struct commonInput {
      BaseContract _base;
      address payable[] _payers;
      address payable[] _payees;
      ...
      uint256 _totalObligations;
      ContractUtility.DisputeType _dispute;
   }
```

Block scoping:

```
uint256 sum = 0;

{
          sum = a + b + c;
}

{
          sum = d + e +f;
}
```

Code Implementation

Coding Approaches

Warning: Contract code size exceeds 24576 bytes!

- Dense contract with too much variables and functions
- Group and extract relevant functions into new contract
- BaseContract → BaseContract + SigningMechanism
  + CommunicationChannel + VotingMechanism

Quick Approach –→ Use optimizer to compile contracts

Dependencies Issue

- Set main contracts as Allowed Admin User (Init Allowed Mapping)

Implementation detail @ **https://github.com/jinghaoong/DeTrust**

Evaluation and Future Developments

Evaluation and Future Developments

General Evaluation

Achievements

- **Dispute Resolution Mechanism**: Follows the same concept as ASTRAEA with some differences to achieve our aim of personalised peer-to-peer governance.
- **Game Theory**: Consistent efforts to drive economic incentives and deter abuse of system through the complementary use of contracts, token and trust scores.

Limitations

- **Oracle Problem:** How do we gather reliable information on general transactions and enforce general outcomes?
- **Future Roadmap:** Do we want to work towards DAO? Are we a DAO?

Evaluation and Future Developments

Smart Contracts

Contract Creation

- **Contract Guidelines:** What constitutes a legitimate contract?
- **Verification:** Who gatekeeps the gatekeepers?

Governance

Game Theory

- **Economic Incentives:** Are our current incentives and penalties sufficient?
- **Fraud**: How do we stop fraudulent account creations?

Evaluation and Future Developments

Dispute Resolution Mechanism

Pre-Dispute

- **Mediation:** Encourage discussion of ideal outcomes before entering dispute

During Dispute

- **Option to Revote:** Retract within stipulated deadline
- **Community Outcome:** Alternative outcome proposed by community
- **Moderators:** Higher weightage for votes

Post-Dispute

- **Additional Dispute:** Initiate another round of dispute

DeTrust

Thank You!