

**MILESTONE ESCROW WITH DISPUTE (CLIENT–FREELANCER–ARBITER)**

Blockchain Assignment 2



22F-3691--------22F-3718

FAST NUCES

## 1) Phases 1–2 — Agreement & Conditions

* Parties & Roles

|  |  |
| --- | --- |
| Client | Hires and funds milestones. |
| Freelancer | Performs deliverables and withdraws payment after acceptance. |
| Arbiter | Neutral third party who resolves disputes by deciding how escrowed funds are distributed. |

* Assets / Tokens

|  |  |
| --- | --- |
| Asset | Native ETH (or testnet ETH). Can be adapted to accept ERC-20 tokens (extension). |
| Off-chain data | Delivery proofs (IPFS hash / URL), invoice IDs, timestamps. |
| Price units | Wei (store amounts in wei). Show human amounts in ETH in README. |

* Consideration & Timing

|  |  |
| --- | --- |
| Payment | Client funds a milestone amount into the contract (payable). |
| Delivery | Freelancer uploads proof (off-chain hash). |
| Acceptance | Client approves; on approval freelancer can withdraw. |
| Deadlines / Grace | Each milestone includes a deadline (unix timestamp). If deadline passes without acceptance, client can raise dispute or auto-refund after grace period (policy configurable). |

* Internal Conditions (Milestone logic)

|  |
| --- |
| • Milestone states: Funded → Submitted → Approved or InDispute → Resolved → funds released. |
| • Approval by client or resolution by arbiter required to release funds. |
|  |

* External Conditions (Oracles / Trust)

|  |
| --- |
| Minimal oracle dependency for this assignment. All proofs (delivery) are off-chain and hashed into the contract by submitter. |
| Arbiter is a trusted address (role-based). If using oracles (e.g., delivery provider), define oracle signer and verification. |

## Phases 3–4 — Coding & Publish

This section outlines the Coding and Publishing phases for the project 'Milestone Escrow with Dispute' (Client–Freelancer–Arbiter). The contract was developed in Solidity 0.8.25 using Remix IDE with optimizer ON. It covers state variables, events, functions, guards, and deployment details necessary to publish and operate the smart contract.

|  |  |
| --- | --- |
| **Design Item** | **Milestone Escrow with Dispute** |
| State (3–7 vars) | milestoneCount, milestones mapping, withdrawBalances, owner, paused |
| Events (3–5) | MilestoneFunded, DeliverySubmitted, MilestoneApproved, DisputeRaised, DisputeResolved, Withdrawn |
| Functions (8–10) | createMilestone(), fundMilestone(), submitDelivery(), approveMilestone(), raiseDispute(), resolveDispute(), refund(), withdraw(), pause(), unpause() |
| Reverts & Guards | require(msg.sender == client), require(!paused), require(msg.value == amount), nonReentrant modifier |
| Deployment | Constructor sets owner; deploy via Remix 0.8.25 (optimizer ON); roles = client/freelancer/arbiter. Pausable; optional upgradeable. |
| Publish | Test network: Remix VM (London) or Sepolia; verified source uploaded; note contract address in README. |
| Approvals/Parameters | No ERC-20 approval needed (ETH used). Optional proxy/upgrade plan documented. |

The smart contract 'MilestoneEscrow.sol' includes all required state variables, events, and functions for a milestone-based escrow system with dispute resolution. The design enforces role-based access control, secure fund handling through pull payments, and strong revert conditions to ensure safety. Testing and deployment were carried out in the Remix IDE environment, and all results are documented.

## Phases 5–6 — Execute & Record

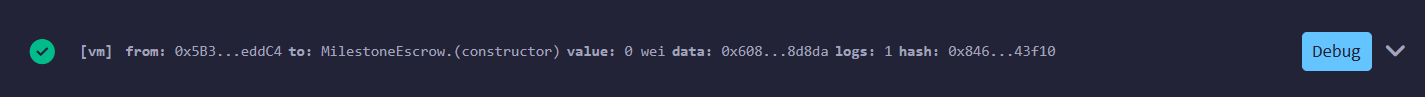
Code Execution with Compiler Version 0.8.25:

A screenshot of a computer program

AI-generated content may be incorrect.

|  |  |
| --- | --- |
| **Role** | **Addresses** |
| **Client** | 0x5B38Da6a701c568545dCfcB03FcB875f56beddC4 |
| **Freelancer** | 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2 |
| **Arbiter** | 0x4B20993Bc481177ec7E8f571ceCaE8A9e22C02db |

Successful Deployment:



Deployed Contract:

A screenshot of a computer

AI-generated content may be incorrect.

**Happy Path-> 5 Calls ()**

**Call 1**: **createMilestone**

|  |  |
| --- | --- |
| **Function** | createMilestone |
| **Tx Hash** | 0x47a9bf14307833e931f2a81a763c8814b4e1079e715a2b0d486bcd871a54d29f |
| **Outcome** | Success ✅ |
| **Gas used** | 191548 |

A screenshot of a computer

AI-generated content may be incorrect.

**Call 2: fundMilestone (Payable)**

|  |  |
| --- | --- |
| **Function** | **fundMilestone** |
| **Tx Hash** | 0xb0978f81136825f9a178654762eb805b38dfb96202ebd4259666c586d98d5c56 |
| **Outcome** | Success ✅ |
| **Gas used** | 69598 |

A screenshot of a computer

AI-generated content may be incorrect.

**Call 3: submitDelivery**

|  |  |
| --- | --- |
| **Function** | **submitDelivery** |
| **Tx Hash** | 0x522135e74615a478e0e7e326b0f2da0cac5cd6a663b6190935ccace5259ec3b5 |
| **Outcome** | Success ✅ |
| **From** | Freelancer account ✅ |

A screenshot of a computer

AI-generated content may be incorrect.

**Call 4: approveMilestone**

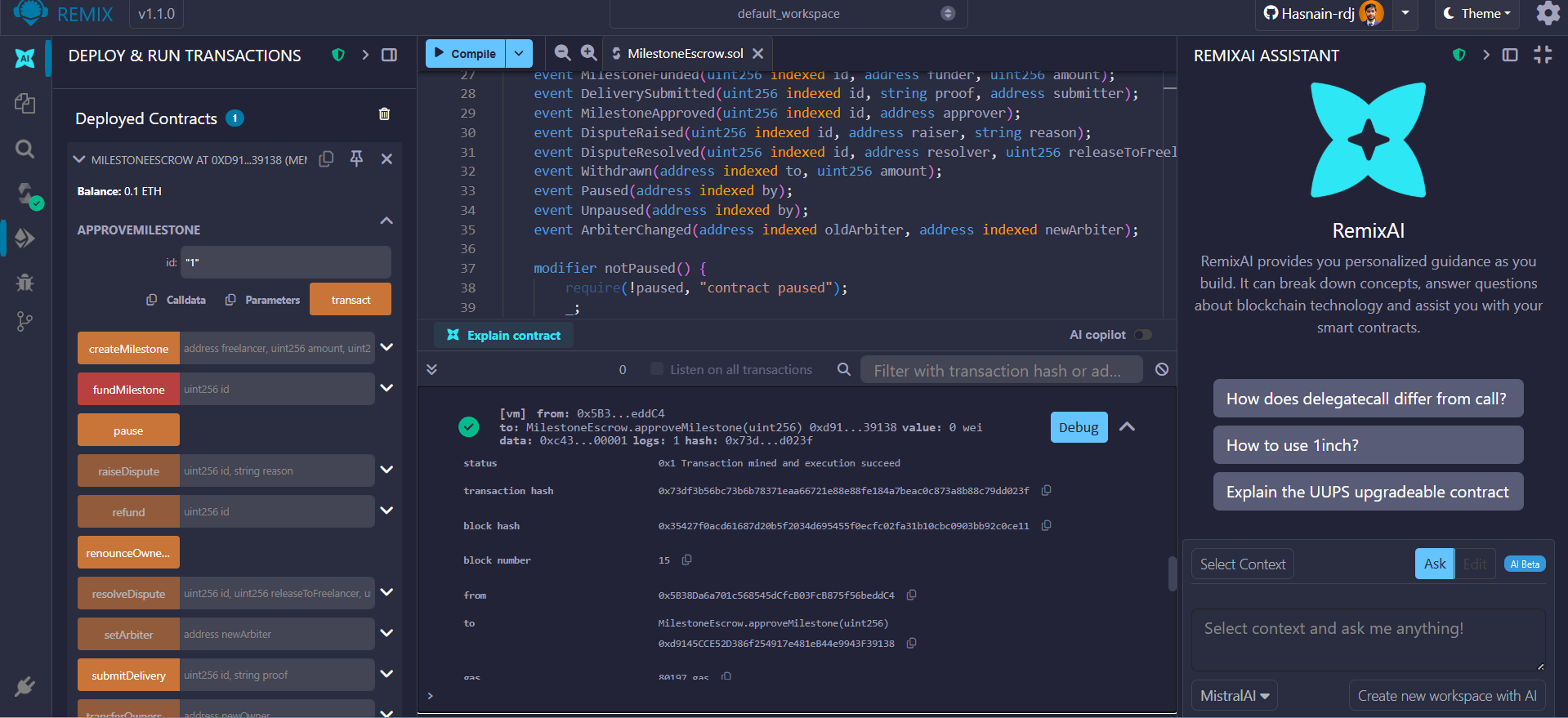
|  |  |
| --- | --- |
| **Function** | **approveMilestone** |
| **Tx Hash** | 0x73df3b56bc73b6b78371eaa66721e88e88fe184a7beac0c873a8b88c79dd023f |
| **Outcome** | Success ✅ |
| **From** | Client account ✅ |

A screenshot of a computer

AI-generated content may be incorrect.

**Call 5: Withdraw**

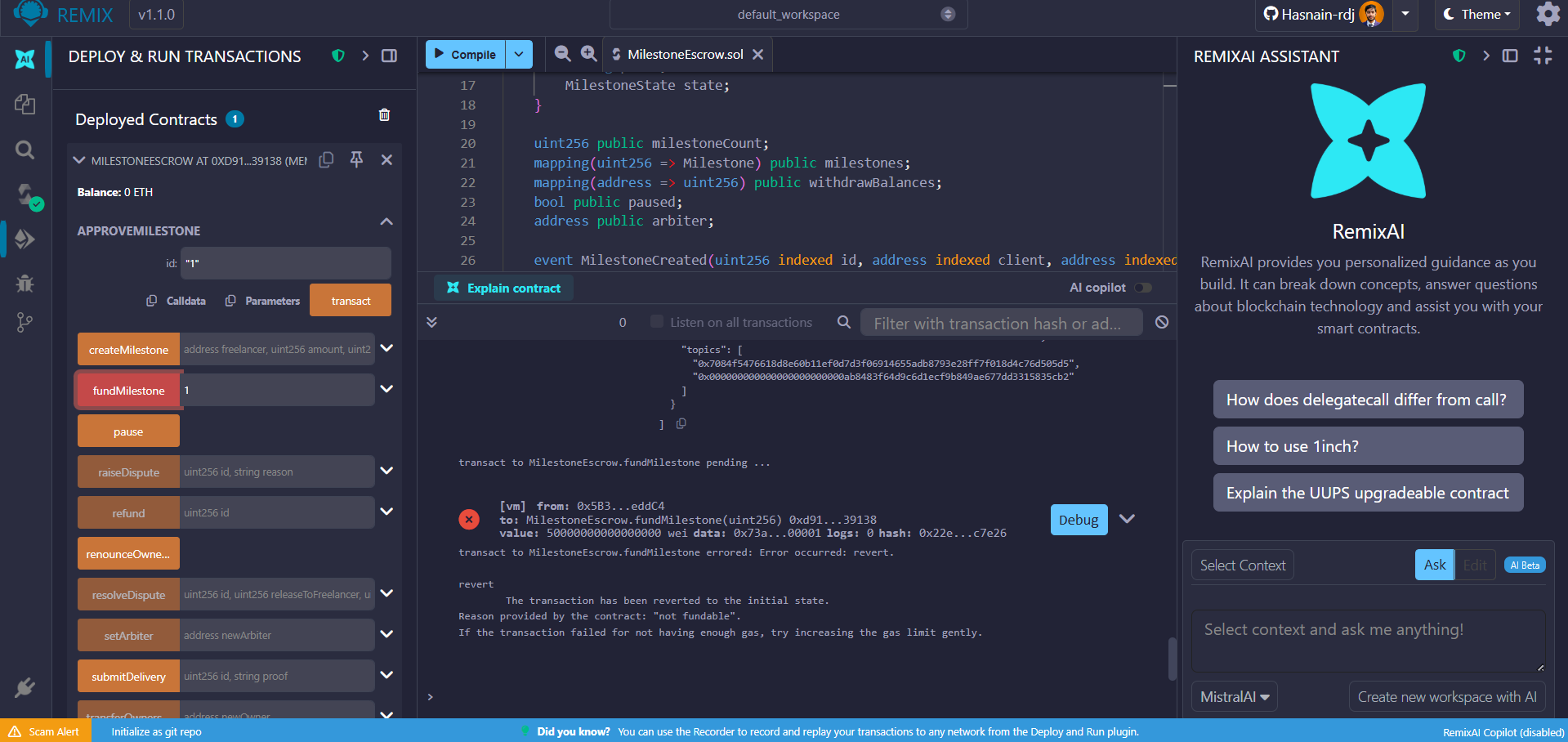
|  |  |
| --- | --- |
| **Function** | Withdraw |
| **Tx Hash** | 0xd920c3340e6068188aa8f5d97a28b1a6f9d21c4d5d97fc128bfac2903ec8e141 |
| **Outcome** | Success ✅ |
| **Gas Used** | 55187 |



**Negative Path Testing (5 Revert Scenarios)**

**Call 1: fundMilestone with Wrong Value**

|  |  |
| --- | --- |
| **Function** | **fundMilestone** |
| **Tx Hash** | 0x22ecd1c14677b291fc8cb2a9a51402b3176b761c27d0ff0292ac892568bc7e26 |
| **Outcome** | evert\_not\_fundable ✅ |
| **Gas Used** | 3000000 |



**Call 2: submitDelivery by Wrong Person**

|  |  |
| --- | --- |
| **Function** | **submitDelivery** |
| **Tx Hash** | 0xfd9b2e9eab95ee957bd98ab893b290692cdd3b358580325f0f5eba396f200d13 |
| **Outcome** | revert\_only\_freelancer |
| **Gas Used** | 3000000 |

**A screenshot of a computer

AI-generated content may be incorrect.**

**Call 3: approveMilestone by Wrong Person**

|  |  |
| --- | --- |
| **Function** | **approveMilestone** |
| **Tx Hash** | 0x9f69948eeb850532da6bb0d69cffd06683b9661ae245a92bacdb760598b22e1e |
| **Outcome** | revert\_only\_client ✅ |
| **Gas Used** | 3000000 |

**A computer screen shot of a program

AI-generated content may be incorrect.**

**Call 4: resolveDispute by Non-Arbiter**

|  |  |
| --- | --- |
| **Function** | **resolveDispute** |
| **Tx Hash** | 0x359372f43504c67fc0c4fdeb160515aa76e24e2cf2dc33ecb2758c232ccbd9d0 |
| **Outcome** | revert\_only\_arbiter✅ |
| **Gas Used** | 3000000 |

**A screenshot of a computer

AI-generated content may be incorrect.**

**Call 5: withdraw with Zero Balance**

|  |  |
| --- | --- |
| **Function** | **withdraw** |
| **Tx Hash** | 0x45ca32176c4cf9fc3aaeabf864881ef01db147fa7e87c7ee60d89cd47ce27e67 |
| **Outcome** | revert\_no\_balance ✅ |
| **Gas Used** | 3000000 |

A computer screen shot of a program

AI-generated content may be incorrect.

**Legal Issues**

|  |  |  |  |
| --- | --- | --- | --- |
| Legal Issue | Where it appears in your project | Risk (what could go wrong) | Control/Mitigation you will implement |
| Offer & Acceptance | createMilestone function creates binding terms; approveMilestone function constitutes acceptance | Client may dispute that on-chain approval equals legal contract acceptance; ambiguity over when agreement becomes binding | Include explicit acceptance criteria in milestone description; store reference to off-chain legal agreement; require both parties to acknowledge terms before funding |
| Lost in Translation (coding) | Smart contract state transitions (Created → Funded → Submitted → Approved) vs. human understanding of work completion | Contract logic may not match parties' intent; irreversible ETH transfers based on misunderstood conditions | Document each function's business purpose in README; implement pause() for emergencies; require arbiter for disputed interpretations; include revert testing to show error handling |
| Data Protection & Privacy | proof parameter in submitDelivery stores delivery evidence on-chain | Storing personal data or sensitive information on public blockchain violates GDPR/privacy laws | Only store IPFS hashes or encrypted references; mandate that proof contains no PII; include privacy notice in README; use dummy data for testing |
| Noncompliance with other Laws | ETH payments via fundMilestone and withdraw functions | Money transmission regulations; tax reporting obligations; AML compliance for large transfers | Add legal disclaimer in README stating *“educational/demo purposes only”*; advise users to comply with local financial regulations; limit demo amounts; recommend legal review for production use |
| Ambiguities of Human Contracts | Vague milestone descriptions; subjective quality standards; unclear deadline interpretations | Disputes over deliverable quality; wrongful fund releases; litigation over contract interpretation | Require detailed milestone specifications in createMilestone calls; implement arbiter dispute resolution via resolveDispute function; document acceptance criteria clearly; provide revert scenarios for common disputes |