


# Solidity Smart Contract Design

How to design Smart Contracts for a protocol

Daniel (Sơn) PHAM 

Solidity Developer Bootcamp



Ngày 28 tháng 7 năm 2023

1. Why do we need to design before coding?
2. What needs to do when designing smart contracts for a protocol?
3. What are Smart contract design patterns?
4. QnA

## 1. Why do we need to design before coding?

- 1.1 Clarity and Purpose
- 1.2 Efficient Development
- 1.3 Collaboration and Communication

## 2. What needs to do when designing smart contracts for a protocol?

## 3. What are Smart contract design patterns?

## 4. QnA

Design trước khi code giúp:

- Xác định rõ ràng goals, requirements, and overall structure.
- Xác định scope, functionalities, and user interactions
- Đảm bảo quá trình phát triển có mục đích đã được định nghĩa rõ ràng.

Với một thiết kế chi tiết (detail design), chúng ta có thể xác định các challenges, bottlenecks tiềm năng từ sớm.

Điều này giúp lập kế hoạch và tối ưu quá trình triển khai, từ đó việc development sẽ hiệu quả hơn, và giảm khả năng xảy ra costly error hoặc remake.

Bản thiết kế giúp cho sự hợp tác hiệu quả giữa các thành viên nhóm (đội researcher, phát triển, QA/QC).

Nó tạo điều kiện thuận lợi cho việc giao tiếp tốt hơn bằng cách cung cấp một hình ảnh về kiến trúc và chức năng, từ đó tạo điều kiện thuận lợi cho việc làm việc nhóm và phối hợp trơn tru trong suốt vòng đời phát triển.

1. Why do we need to design before coding?

**2. What needs to do when designing smart contracts for a protocol?**

- 2.1 Define the Protocol Requirements
- 2.2 Implement Design Patterns if possible
- 2.3 Identify Contract Components
- 2.4 Document Thoroughly

3. What are Smart contract design patterns?

4. QnA

# Define the Protocol Requirements

---



Xác định rõ yêu cầu và mục tiêu của giao thức.

Hiểu rõ những gì các hợp đồng thông minh cần đạt được và tương tác mà chúng phải hỗ trợ trong giao thức.

Cân nhắc các yếu tố như scalability, security, and gas optimization.



# Implement Design Patterns if possible

---



Thiết kế cấu trúc của hợp đồng thông minh có tác động lớn đến chi phí execution. Design patterns có thể tái sử dụng, các giải pháp thông thường được sử dụng để giải quyết các lỗi thiết kế tái diễn.

# Identify Contract Components

---



Phân chia chức năng của giao thức thành các thành phần hợp đồng thông minh modul và được định nghĩa rõ ràng.

Tiếp cận theo hướng modul này giúp mã nguồn dễ quản lý hơn, dễ bảo trì hơn và thuận tiện cho việc tái sử dụng mã nguồn.

# Document Thoroughly

---



Cung cấp tài liệu chi tiết và rõ ràng cho các hợp đồng thông minh, giải thích chức năng, giao diện và bất kỳ rủi ro tiềm tàng nào.

Các hợp đồng được lập tài liệu kỹ lưỡng sẽ giúp các nhà phát triển khác hiểu và tương tác với giao thức một cách dễ dàng.

1. Why do we need to design before coding?

2. What needs to do when designing smart contracts for a protocol?

**3. What are Smart contract design patterns?**

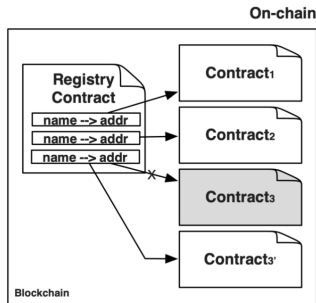
- 3.1 Contract Registry
- 3.2 Data Contract
- 3.3 Diamond - Multi-Facet Proxy
- 3.4 Factory Contract
- 3.5 Proxy - Contract Relay

4. QnA

# Contract Registry

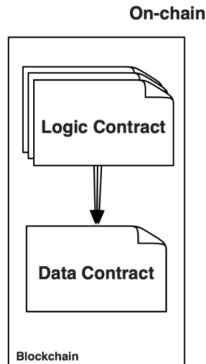
## Problem

- Deployed smart contracts are immutability.
- Human-readable names.
- Transparency.



## Problem

**Cost:** If a public blockchain is used, storing data on the blockchain costs money. Thus, copying data from an old version of a smart contract to a new version should be avoided or minimised.

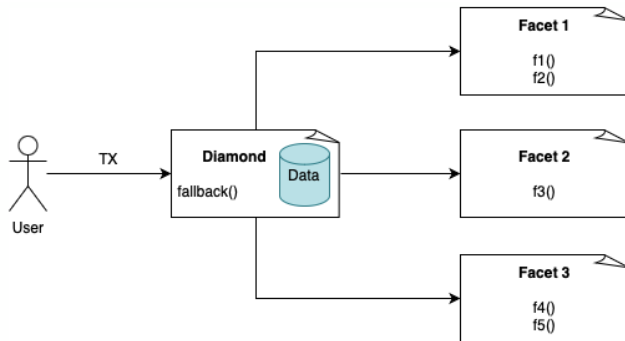


# Diamond - Multi-Facet Proxy



## Problem

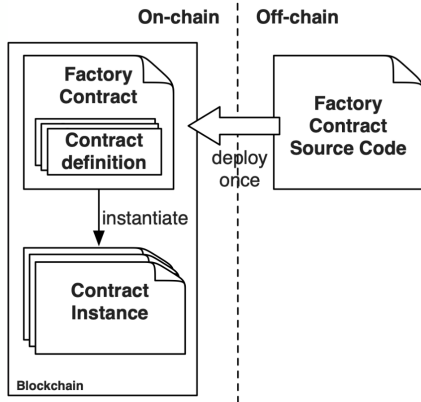
A smart contract is over size limit.



# Factory Contract

## Problem

The smart contracts following the same functionality can be implemented as a template. New contracts can be deployed by an operator, not a dev.

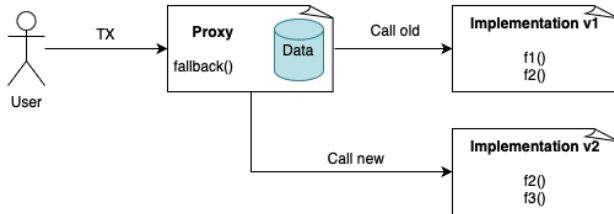




# Proxy - Contract Relay

## Problem

- Immutability.
- Upgradability.
- Flexibility.



1. Why do we need to design before coding?
2. What needs to do when designing smart contracts for a protocol?
3. What are Smart contract design patterns?
4. QnA

