## Câu 1 Trình bày định nghĩa công nghệ Blockchain. Nêu rõ các đặc điểm chính giúp Blockchain trở nên tin cậy và khác biệt so với cơ sở dữ liệu truyền thống.

**Định nghĩa công nghệ Blockchain**

Blockchain là một công nghệ lưu trữ và truyền tải thông tin theo một chuỗi khối, trong đó mỗi khối chứa dữ liệu và liên kết với khối trước đó bằng một mã băm duy nhất. Công nghệ này hoạt động dựa trên nguyên tắc phân quyền, bảo mật cao và tính minh bạch, giúp đảm bảo dữ liệu không thể bị chỉnh sửa sau khi đã được ghi nhận.

**Đặc điểm chính giúp Blockchain trở nên tin cậy và khác biệt**

1. **Tính phân quyền** – Không có máy chủ trung tâm, dữ liệu được lưu trữ trên nhiều nút mạng (nodes) giúp giảm thiểu rủi ro bị tấn công hoặc thao túng bởi một cá nhân hay tổ chức.
2. **Tính bất biến** – Một khi dữ liệu được ghi vào chuỗi khối, nó không thể bị thay đổi hoặc xóa bỏ, trừ khi toàn bộ mạng đồng ý. Điều này làm tăng độ tin cậy của dữ liệu.
3. **Minh bạch** – Mọi giao dịch đều được ghi lại công khai và có thể kiểm tra trên sổ cái kỹ thuật số, giúp nâng cao tính minh bạch và giảm gian lận.
4. **Bảo mật cao** – Dữ liệu trong Blockchain được mã hóa và chỉ có thể được giải mã bởi những người có quyền truy cập, giúp bảo vệ khỏi các cuộc tấn công mạng.
5. **Tự động hóa với hợp đồng thông minh** – Blockchain có thể triển khai **smart contracts**, giúp tự động thực hiện các thỏa thuận mà không cần bên thứ ba.

Blockchain không chỉ khác biệt so với các cơ sở dữ liệu truyền thống nhờ tính bảo mật và minh bạch mà còn tạo nền tảng cho nhiều ứng dụng mới như tiền mã hóa, quản lý chuỗi cung ứng và xác thực danh tính kỹ thuật số. Bạn có muốn đi sâu hơn vào ứng dụng của Blockchain không?

## Câu 2. So sánh mô hình Client/Server với mô hình Peer-to-Peer (P2P) trong bối cảnh ứng dụng Blockchain. Ưu nhược điểm của từng mô hình là gì?

| **Đặc điểm** | **Mô hình Client/Server** | **Mô hình Peer-to-Peer (P2P)** |
| --- | --- | --- |
| **Cấu trúc hệ thống** | Dữ liệu tập trung ở máy chủ, các client chỉ truy cập và sử dụng dữ liệu. | Dữ liệu được phân tán trên tất cả các node trong mạng, mỗi node hoạt động như một phần của hệ thống. |
| **Khả năng chịu lỗi** | Nếu máy chủ bị lỗi, toàn bộ hệ thống có thể bị ảnh hưởng. | Không có điểm lỗi trung tâm, mạng vẫn hoạt động ngay cả khi một số node gặp sự cố. |
| **Bảo mật** | Dữ liệu dễ bị tấn công nếu máy chủ bị xâm nhập. | Mỗi node đều có bản sao dữ liệu, giúp chống giả mạo và gian lận. |
| **Hiệu suất** | Có thể xử lý nhanh hơn với máy chủ mạnh nhưng dễ bị quá tải. | Phân tán tài nguyên giúp hệ thống mở rộng linh hoạt nhưng có thể chậm hơn do việc đồng bộ dữ liệu. |
| **Ứng dụng trong Blockchain** | Không phù hợp vì Blockchain yêu cầu tính phi tập trung và bảo mật cao. | Là nền tảng của Blockchain vì đảm bảo dữ liệu phân tán, an toàn và minh bạch. |

**Ưu và Nhược điểm của từng mô hình**

**Mô hình Client/Server**

**Ưu điểm**:

* Dễ quản lý và kiểm soát dữ liệu.
* Hiệu suất cao trong điều kiện máy chủ mạnh.

**Nhược điểm**:

* Phụ thuộc vào máy chủ trung tâm, dễ bị tấn công.
* Hạn chế khả năng mở rộng khi số lượng client tăng lên quá lớn.

**Mô hình Peer-to-Peer (P2P)**

**Ưu điểm**:

* Không có điểm lỗi trung tâm, giúp tăng độ tin cậy.
* Dữ liệu minh bạch và chống giả mạo tốt hơn.
* Phù hợp với các ứng dụng phi tập trung như Blockchain.

**Nhược điểm**:

* Quá trình đồng bộ hóa dữ liệu có thể chậm hơn.
* Quản lý mạng phức tạp hơn do không có máy chủ trung tâm.

## Câu 3. Mô tả cấu trúc của một khối trong Blockchain. Giải thích vai trò của phần header và phần dữ liệu giao dịch.

**Cấu trúc của một khối trong Blockchain**

Mỗi khối (**block**) trong Blockchain bao gồm hai phần chính: **phần header** (tiêu đề khối) và **phần dữ liệu giao dịch**. Cấu trúc này giúp Blockchain đảm bảo tính bảo mật, liên kết giữa các khối, và khả năng xác minh dữ liệu.

**1. Phần header (Block Header)**

Header chứa thông tin giúp định danh khối và liên kết với các khối trước đó. Các thành phần quan trọng trong header gồm:

* **Mã băm của khối trước (Previous Block Hash)** – Liên kết khối hiện tại với khối trước, giúp tạo thành chuỗi khối liên tục và bảo vệ khỏi sự giả mạo.
* **Mã băm của khối hiện tại (Block Hash)** – Được tạo dựa trên dữ liệu khối, đảm bảo tính duy nhất và bất biến của khối đó.
* **Merkle Root** – Giá trị đại diện cho toàn bộ các giao dịch trong khối, giúp xác minh tính toàn vẹn của dữ liệu giao dịch một cách nhanh chóng.
* **Nonce** – Một số ngẫu nhiên được sử dụng trong quá trình khai thác (mining) để tìm giá trị băm hợp lệ.
* **Dấu thời gian (Timestamp)** – Ghi lại thời điểm khối được tạo, giúp theo dõi lịch sử giao dịch.
* **Độ khó (Difficulty Target)** – Mức độ phức tạp trong việc tìm giá trị băm hợp lệ, ảnh hưởng đến quá trình khai thác.

**2. Phần dữ liệu giao dịch (Transaction Data)**

Đây là phần chứa danh sách các giao dịch được ghi vào khối. Mỗi giao dịch bao gồm các thông tin như:

* **Người gửi và người nhận** – Địa chỉ ví của hai bên thực hiện giao dịch.
* **Số lượng tài sản giao dịch** – Ví dụ: số lượng Bitcoin được chuyển đi.
* **Thông tin chữ ký số** – Giúp xác minh quyền sở hữu và tính hợp lệ của giao dịch.

**Vai trò của từng phần**

* **Phần header**: Giữ vai trò bảo mật, giúp liên kết các khối trong Blockchain, đảm bảo tính toàn vẹn và bất biến của dữ liệu.
* **Phần dữ liệu giao dịch**: Lưu trữ các giao dịch thực tế, đảm bảo mọi hoạt động được ghi lại minh bạch và có thể kiểm tra.

## Câu 4. Các giao thức đồng thuận như PoW, PoS và PBFT hoạt động như thế nào? Phân tích ưu nhược điểm của từng loại giao thức trong việc bảo vệ tính nhất quán của chuỗi khối

**Các giao thức đồng thuận trong Blockchain**

**1. Proof of Work (PoW)**

PoW là cơ chế đồng thuận phổ biến nhất, được sử dụng trong Bitcoin.

**Cách hoạt động:**

* Các thợ đào (**miners**) cạnh tranh giải một bài toán mật mã phức tạp bằng cách tìm giá trị **nonce** thích hợp.
* Người tìm ra lời giải sẽ được quyền thêm khối mới vào Blockchain và nhận phần thưởng.
* Quá trình này đòi hỏi tài nguyên tính toán lớn, giúp ngăn chặn các cuộc tấn công giả mạo.

**Ưu điểm:**  
Bảo mật cao, chống giả mạo hiệu quả.  
Không cần xác minh danh tính của các nút tham gia.

**Nhược điểm:**  
Tiêu tốn nhiều năng lượng và tài nguyên phần cứng.  
Tốc độ xử lý giao dịch chậm do đòi hỏi tính toán phức tạp.

**2. Proof of Stake (PoS)**

PoS là cơ chế đồng thuận thay thế PoW với mục tiêu giảm tiêu thụ năng lượng.

**Cách hoạt động:**

* Người tham gia phải nắm giữ một lượng tiền mã hóa nhất định để có quyền tạo khối mới.
* Người có nhiều tài sản hơn sẽ có xác suất cao hơn được chọn để xác nhận giao dịch.
* Không cần giải bài toán phức tạp như PoW, giúp tiết kiệm năng lượng.

**Ưu điểm:**  
Tiết kiệm năng lượng hơn so với PoW.  
Tốc độ xử lý giao dịch nhanh hơn.

**Nhược điểm:**  
Người sở hữu nhiều tài sản có thể kiểm soát mạng lưới (tính phi tập trung có thể giảm).  
Một số mô hình PoS yêu cầu thời gian khóa tài sản, gây hạn chế tính thanh khoản.

**3. Practical Byzantine Fault Tolerance (PBFT)**

PBFT được sử dụng trong các hệ thống blockchain dành cho doanh nghiệp, chẳng hạn như Hyperledger Fabric.

**Cách hoạt động:**

* Các nút mạng xác nhận giao dịch thông qua hệ thống bỏ phiếu đồng thuận.
* Giao thức yêu cầu các nút tham gia phải đồng ý với nhau trước khi thêm khối mới.
* Thiết kế nhằm chống lại các nút có hành vi gian lận (có thể chấp nhận một phần tử độc hại nhưng vẫn duy trì tính nhất quán của hệ thống).

**Ưu điểm:**  
Tốc độ xử lý giao dịch cao, không cần sức mạnh tính toán lớn như PoW.  
Phù hợp với các hệ thống doanh nghiệp vì đảm bảo hiệu suất và tính bảo mật.

**Nhược điểm:**  
Yêu cầu xác minh danh tính của các nút tham gia, không thực sự phi tập trung như PoW hoặc PoS.  
Khi quy mô mạng mở rộng, quá trình xác nhận giao dịch có thể chậm lại do cần sự đồng thuận từ nhiều nút.

## Câu 5. Vì sao Blockchain có thể đảm bảo tính toàn vẹn dữ liệu? Trình bày vai trò của thuật toán SHA-256 trong việc mã hóa và xác thực giao dịch.

**Tại sao Blockchain đảm bảo tính toàn vẹn dữ liệu?**

Blockchain đảm bảo tính toàn vẹn dữ liệu nhờ các cơ chế bảo mật mạnh mẽ và nguyên tắc bất biến của hệ thống. Các yếu tố chính giúp duy trì tính toàn vẹn của dữ liệu bao gồm:

1. **Chuỗi liên kết bằng mã băm** – Mỗi khối chứa mã băm của khối trước đó, tạo thành chuỗi khối liên tục. Nếu một khối bị thay đổi, toàn bộ chuỗi sẽ bị ảnh hưởng, giúp ngăn chặn giả mạo.
2. **Cấu trúc phân tán** – Dữ liệu được sao lưu trên nhiều nút mạng thay vì tập trung vào một máy chủ, giúp chống lại các cuộc tấn công và giảm rủi ro mất mát dữ liệu.
3. **Giao thức đồng thuận** – Blockchain sử dụng các thuật toán như **Proof of Work (PoW)** hoặc **Proof of Stake (PoS)** để đảm bảo mọi giao dịch được xác minh bởi nhiều bên, tránh gian lận.
4. **Merkle Tree** – Cấu trúc cây giúp xác minh toàn vẹn dữ liệu giao dịch nhanh chóng mà không cần kiểm tra toàn bộ dữ liệu, giúp tăng hiệu suất.
5. **Thuật toán băm SHA-256** – Đóng vai trò then chốt trong việc mã hóa và đảm bảo tính bất biến của dữ liệu.

**Vai trò của SHA-256 trong mã hóa và xác thực giao dịch**

SHA-256 (**Secure Hash Algorithm 256-bit**) là thuật toán băm được sử dụng rộng rãi trong Blockchain, đặc biệt là trong Bitcoin. Nó đảm bảo tính toàn vẹn và bảo mật của dữ liệu theo các cách sau:

1. **Mã hóa dữ liệu giao dịch** – SHA-256 chuyển đổi dữ liệu giao dịch thành một mã băm duy nhất, giúp đảm bảo rằng dữ liệu không thể bị thay đổi sau khi được ghi vào blockchain.
2. **Xác minh tính hợp lệ của khối** – Mỗi khối có một **Block Hash**, được tạo từ dữ liệu của khối và thuật toán SHA-256. Nếu ai đó cố gắng chỉnh sửa nội dung, giá trị băm sẽ thay đổi, báo hiệu sự gian lận.
3. **Duy trì liên kết giữa các khối** – Mỗi khối lưu trữ **mã băm của khối trước**, giúp bảo vệ chuỗi khỏi sự sửa đổi trái phép.
4. **Hỗ trợ quá trình khai thác (mining)** – SHA-256 đóng vai trò trong việc giải bài toán mật mã mà các thợ đào cần xử lý để tạo khối mới trong cơ chế Proof of Work.

## Câu 6. Smart Contract là gì? Trong thực tế, hợp đồng thông minh có thể được sử dụng trong những lĩnh vực nào? Lấy ví dụ minh họa.

Hợp đồng thông minh (Smart Contract) là một chương trình máy tính hoặc giao thức giao dịch dựa trên công nghệ **Blockchain**. Nó giúp thực hiện các điều khoản của hợp đồng một cách tự động mà không cần bên trung gian. Các điều khoản trong hợp đồng thông minh được viết bằng ngôn ngữ lập trình và sẽ tự động thực thi khi các điều kiện được đáp ứng.

**Ứng dụng của hợp đồng thông minh trong thực tế**

Hợp đồng thông minh có thể được sử dụng trong nhiều lĩnh vực khác nhau, bao gồm:

* **Tài chính & Ngân hàng**: Tự động hóa các giao dịch, thanh toán, cho vay và bảo hiểm mà không cần sự can thiệp của bên thứ ba.
* **Bất động sản**: Hợp đồng thông minh giúp thực hiện giao dịch mua bán nhà đất một cách minh bạch và nhanh chóng.
* **Chuỗi cung ứng**: Theo dõi và xác minh nguồn gốc sản phẩm, đảm bảo tính minh bạch trong quá trình vận chuyển.
* **Y tế**: Quản lý hồ sơ bệnh nhân, bảo hiểm y tế và thanh toán chi phí khám chữa bệnh một cách tự động.
* **Giải trí & Bản quyền**: Quản lý quyền sở hữu trí tuệ, thanh toán tiền bản quyền cho nghệ sĩ và nhà sáng tạo nội dung.

**Ví dụ minh họa**

Một ví dụ điển hình về hợp đồng thông minh là **máy bán hàng tự động**. Khi bạn bỏ tiền vào máy và chọn sản phẩm, máy sẽ tự động kiểm tra số tiền, xác nhận giao dịch và đưa ra sản phẩm mà không cần sự can thiệp của nhân viên bán hàng. Tương tự, hợp đồng thông minh hoạt động dựa trên các điều kiện được lập trình sẵn, giúp giao dịch diễn ra nhanh chóng và minh bạch.

## Câu 7. Khi triển khai hợp đồng thông minh bằng Python với thư viện Web3.py, cần thực hiện những bước kỹ thuật nào? Nêu các bước chính

Để triển khai hợp đồng thông minh bằng Python với thư viện **Web3.py**, bạn cần thực hiện các bước chính sau:

**1. Chuẩn bị môi trường**

* Cài đặt Python và thư viện Web3.py bằng lệnh:

pip install web3

* Thiết lập kết nối với mạng Ethereum (ví dụ: sử dụng Infura hoặc chạy một node Ethereum cục bộ).

**2. Viết hợp đồng thông minh**

* Hợp đồng thông minh thường được viết bằng **Solidity**.
* Ví dụ một hợp đồng đơn giản:

pragma solidity ^0.8.0;

contract SimpleContract {

uint public value;

function setValue(uint \_value) public {

value = \_value;

}

}

**3. Biên dịch và triển khai hợp đồng**

* Sử dụng **Solc** hoặc **Remix IDE** để biên dịch hợp đồng thành **bytecode** và **ABI**.
* Dùng Web3.py để triển khai hợp đồng lên mạng Ethereum.

**4. Tương tác với hợp đồng**

* Kết nối với hợp đồng bằng Web3.py:

from web3 import Web3

web3 = Web3(Web3.HTTPProvider("https://mainnet.infura.io/v3/YOUR\_INFURA\_PROJECT\_ID"))

contract\_address = "0xYourContractAddress"

abi = "YourContractABI"

contract = web3.eth.contract(address=contract\_address, abi=abi)

Gọi các hàm của hợp đồng:

tx\_hash = contract.functions.setValue(10).transact({'from': web3.eth.accounts[0]})

receipt = web3.eth.wait\_for\_transaction\_receipt(tx\_hash)

print("Transaction successful:", receipt)

**5. Kiểm thử và bảo trì**

* Kiểm thử hợp đồng bằng **Truffle** hoặc **Hardhat**.
* Theo dõi và bảo trì hợp đồng sau khi triển khai.

## Câu 8. Trong một dự án Khoa học dữ liệu có yêu cầu xác minh nguồn gốc dữ liệu đầu vào, Anh/Chị sẽ tích hợp Blockchain như thế nào? Trình bày đề xuất quy trình ứng dụng cụ thể

**Tích hợp Blockchain vào xác minh nguồn gốc dữ liệu trong Khoa học Dữ liệu**

Blockchain là một công nghệ **sổ cái phân tán**, giúp đảm bảo tính minh bạch, bất biến và truy xuất nguồn gốc dữ liệu một cách chính xác. Trong một dự án Khoa học Dữ liệu, việc tích hợp Blockchain có thể giúp xác minh nguồn gốc dữ liệu đầu vào, đảm bảo tính toàn vẹn và độ tin cậy của dữ liệu.

**Quy trình ứng dụng cụ thể**

**1. Xác định yêu cầu và thiết kế hệ thống**

* Xác định loại dữ liệu cần xác minh (ví dụ: dữ liệu tài chính, y tế, chuỗi cung ứng).
* Lựa chọn nền tảng Blockchain phù hợp (Ethereum, Hyperledger, Solana…).
* Thiết kế cấu trúc lưu trữ dữ liệu trên Blockchain (on-chain hoặc off-chain).

**2. Tạo hợp đồng thông minh để ghi nhận dữ liệu**

* Viết hợp đồng thông minh bằng **Solidity** để lưu trữ thông tin nguồn gốc dữ liệu.
* Các thông tin quan trọng cần lưu trữ:
  + **Nguồn gốc dữ liệu** (người cung cấp, thời gian thu thập).
  + **Chữ ký số** để xác thực dữ liệu.
  + **Băm dữ liệu (hash)** để đảm bảo tính toàn vẹn.

**3. Ghi nhận dữ liệu lên Blockchain**

* Khi dữ liệu được thu thập, hệ thống sẽ tạo một **băm (hash)** của dữ liệu và ghi vào Blockchain.
* Dữ liệu gốc có thể được lưu trữ trên hệ thống ngoài (off-chain) và liên kết với Blockchain bằng mã định danh duy nhất.

**4. Xác minh và truy xuất nguồn gốc dữ liệu**

* Khi cần kiểm tra nguồn gốc dữ liệu, hệ thống sẽ:
  + Truy xuất thông tin từ Blockchain.
  + So sánh **băm dữ liệu** với dữ liệu gốc để xác minh tính toàn vẹn.
  + Kiểm tra chữ ký số để đảm bảo dữ liệu không bị giả mạo.

**5. Ứng dụng thực tế**

* **Chuỗi cung ứng**: Xác minh nguồn gốc sản phẩm từ nhà sản xuất đến người tiêu dùng.
* **Y tế**: Đảm bảo tính chính xác của hồ sơ bệnh nhân và lịch sử điều trị.
* **Dữ liệu mở**: Quản lý và xác minh nguồn gốc dữ liệu nghiên cứu khoa học.

## Câu 9. Một sinh viên đã viết thành công một Smart Contract để lưu trữ thông điệp chuỗi văn bản, nhưng khi triển khai gặp lỗi "invalid opcode". Theo Anh/Chị, nguyên nhân có thể là gì? Hãy đề xuất các bước kiểm tra khắc phục.

Lỗi **"invalid opcode"** trong Smart Contract thường xảy ra khi có vấn đề trong quá trình thực thi mã hợp đồng trên Ethereum Virtual Machine (EVM). Dưới đây là một số nguyên nhân phổ biến và cách khắc phục:

**Nguyên nhân có thể gây lỗi "invalid opcode"**

1. **Lỗi cú pháp hoặc logic trong hợp đồng**
   * Hợp đồng có thể chứa các lệnh không hợp lệ hoặc không được hỗ trợ bởi phiên bản EVM hiện tại.
2. **Sử dụng opcode không hợp lệ**
   * Một số opcode có thể bị loại bỏ hoặc không được hỗ trợ trong phiên bản Solidity hoặc EVM mà bạn đang sử dụng.
3. **Lỗi khi truy cập bộ nhớ hoặc biến không hợp lệ**
   * Nếu hợp đồng cố gắng truy cập một vùng bộ nhớ không hợp lệ hoặc sử dụng biến chưa được khởi tạo, lỗi này có thể xảy ra.
4. **Thiếu gas khi thực thi hợp đồng**
   * Nếu giao dịch không có đủ gas để hoàn thành, EVM có thể dừng thực thi và trả về lỗi "invalid opcode".
5. **Lỗi khi gọi một hàm không tồn tại**
   * Nếu hợp đồng cố gắng gọi một hàm không được định nghĩa hoặc không có trong ABI, lỗi này có thể xuất hiện.

**Các bước kiểm tra và khắc phục**

1. **Kiểm tra mã nguồn hợp đồng**
   * Đảm bảo hợp đồng không chứa lỗi cú pháp hoặc logic sai.
   * Kiểm tra phiên bản Solidity có tương thích với EVM hiện tại không.
2. **Kiểm tra ABI và bytecode**
   * Đảm bảo ABI và bytecode của hợp đồng khớp với phiên bản đã triển khai.
3. **Kiểm tra lượng gas cung cấp**
   * Tăng lượng gas khi gửi giao dịch để đảm bảo hợp đồng có đủ tài nguyên để thực thi.
4. **Sử dụng công cụ debug**
   * Dùng **Remix IDE** hoặc **Ganache** để kiểm tra lỗi và theo dõi quá trình thực thi hợp đồng.
5. **Kiểm tra opcode được sử dụng**
   * Đảm bảo hợp đồng không sử dụng opcode đã bị loại bỏ hoặc không được hỗ trợ trong phiên bản EVM hiện tại.
6. **Kiểm tra quyền truy cập và biến**
   * Đảm bảo tất cả biến được khởi tạo đúng cách và không truy cập vùng bộ nhớ không hợp lệ.