BAB II

KAJIAN PUSTAKA

A. Kajian Teoritis

1. Sistem

Sistem merupakan suatu kesatuan yang tersusun atas berbagai komponen, elemen, atau subsistem yang saling berhubungan dan berinteraksi untuk mencapai suatu tujuan, di mana setiap bagian memiliki fungsi masing-masing namun seluruhnya bekerja sama sehingga membentuk satu kesatuan yang utuh (Soufitri, 2023:3). Sejalan dengan pendapat tersebut, Widarti et al., (2024:2) menjelaskan bahwa sistem disebut sebagai suatu entitas yang terdiri dari elemen atau komponen yang saling berhubungan dan berinteraksi secara berkesinambungan dengan lingkungannya. Setiap elemen memiliki mekanisme kerja yang berbeda, tetapi tetap saling mendukung, bergantung, dan berkolaborasi sehingga tujuan sistem dapat tercapai secara efektif..

Hartono, (2021:3) juga mengemukakan bahwa sistem merupakan sekumpulan komponen atau elemen, baik bersifat fisik maupun nonfisik, yang saling berhubungan, berinteraksi, serta bekerja sama secara teratur untuk mencapai tujuan tertentu. Setiap sistem bersifat integratif karena terdiri dari subsistem yang saling ketergantungan dan tidak dapat dipisahkan. Selain itu, suatu sistem dapat menjadi bagian dari sistem lain yang memiliki cakupan lebih luas, yang dikenal sebagai supersistem atau suprasistem.

Berdasarkan beberapa pendapat di atas, dapat disimpulkan bahwa sistem adalah suatu kesatuan yang terdiri dari elemen, komponen, atau

subsistem yang saling berhubungan, berinteraksi, dan bekerja sama secara terstruktur guna mencapai tujuan tertentu. Setiap elemen dalam sistem memiliki fungsi dan cara kerja masing-masing, namun tetap saling bergantung dan bekerjasama sehingga membentuk satu kesatuan yang utuh. Sistem bersifat integratif karena terdiri dari subsistem yang tidak dapat dipisahkan, serta dapat menjadi bagian dari sistem yang lebih besar (supersistem atau suprasistem) yang terus berkembang sesuai dengan kebutuhan dan lingkungannya.

2. *File Sharing*

Dalam konteks akademik, *file sharing* didefinisikan sebagai aktivitas berbagi, bertukar, maupun mendistribusikan materi pembelajaran seperti catatan pelajaran, tugas, hingga kunci jawaban melalui platform berbasis internet, baik secara gratis maupun melalui mekanisme pertukaran atau pembayaran tersebut (Rogerson & Basanta dalam Lancaster & Cotarlan, 2021:2). Di sisi lain, Chong & Harun (2025:98) menjelaskan bahwa *file sharing* memiliki peran penting dalam berbagai bidang, di mana sistem ini memungkinkan pengguna untuk mengunggah, mengakses, dan membagikan informasi digital seperti dokumen maupun berkas multimedia, baik secara privat maupun publik. Kemudahan tersebut menjadikan teknologi ini sebagai salah satu langkah utama dalam mendukung kerja sama dan pertukaran informasi pada era digital ini.

Berdasarkan kedua pendapat tersebut, *file sharing* tidak hanya berfungsi sebagai media distribusi informasi digital, tetapi juga menjadi

sarana untuk mendukung kolaborasi dan pertukaran informasi. Meskipun demikian, kemudahan dalam proses distribusi berkas juga berpotensi menimbulkan berbagai permasalahan, seperti penyalahgunaan data dan pelanggaran integritas akademik. Oleh sebab itu, diperlukan mekanisme *file sharing* yang mampu memberikan jaminan keamanan, keaslian, serta pengendalian akses terhadap data, salah satunya melalui pemanfaatan teknologi *blockchain* dan IPFS.

3. Karya Digital

Karya digital merupakan hasil ciptaan yang dibuat, disimpan, dinikmati, serta didistribusikan dalam bentuk digital melalui pemanfaatan teknologi informasi. Menuru Jaman et al. (2021:10), karya digital dapat berupa *e-book*, musik, video, perangkat lunak, gambar, aplikasi, *font*, maupun berbagai produk digital lainnya yang memiliki nilai ekonomi dan kreativitas. Selain memiliki nilai ekonomi, karya digital juga merepresentasikan perkembangan teknologi dan kreativitas manusia. Jefta et al. (2024:380) menyatakan bahwa karya seni digital memiliki nilai tambah karena mudah dipublikasikan, disebarluaskan, dan diakses melalui internet maupun berbagai perangkat digital.

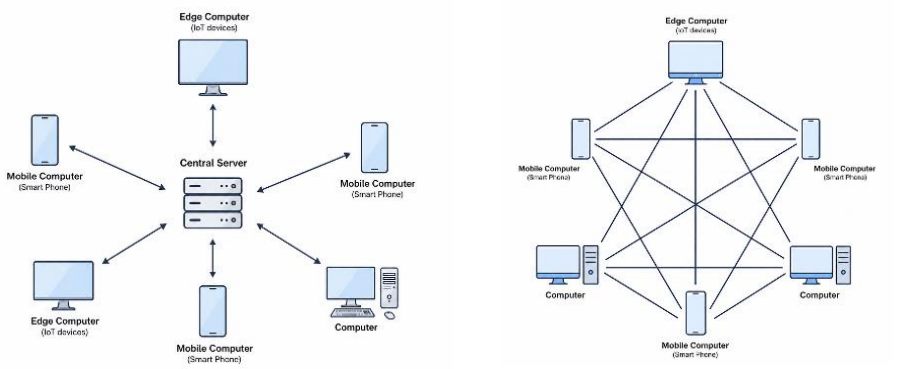
Kemudahan dalam proses distribusi tersebut di sisi lain meningkatkan risiko terjadinya pelanggaran hak cipta, seperti penyalinan, pencurian, maupun penggunaan karya tanpa persetujuan pemiliknya. Agusman & Fathoni (2025:387) menjelaskan bahwa integrasi NFT dengan

smart contract dapat menjadi solusi perlindungan hak cipta karena mampu menciptakan sistem yang bersifat otomatis, transparan, dan tahan manipulasi.

Mengacu pada berbagai pendapat tersebut, karya digital dapat dipahami sebagai hasil kreativitas yang diwujudkan dalam format digital dan memiliki nilai ekonomi maupun intelektual. Meskipun menawarkan kemudahan dalam penyimpanan serta distribusi, karya digital tetap memerlukan mekanisme perlindungan yang memadai untuk menjaga keaslian dan hak kepemilikannya. Dalam konteks tersebut, penerapan teknologi *blockchain* dan *smart contract* menjadi salah satu pendekatan yang relevan untuk meningkatkan keamanan serta perlindungan terhadap karya digital.

4. *Decentralized Application* (DApp)

Menurut Homoliak, (2026:15), *Decentralized Application* (DApp) merupakan aplikasi yang beroperasi pada jaringan terdesentralisasi dengan memanfaatkan protokol *trustless*, Pendekatan ini memungkinkan aplikasi dijalankan oleh banyak pengguna tanpa bergantung pada satu otoritas pusat, sehingga mengurangi risiko terjadinya *single point of failure*. Ahmed et al., (2026:797) juga berpendapat bahwa peralihan dari model terpusat menuju model terdesentralisasi merupakan suatu perubahan paradigma yang berpengaruh terhadap proses penyimpanan, pengelolaan, dan tata kelola data. Model tersebut dinilai mampu meningkatkan keamanan dan integritas data sekaligus memperluas akses informasi secara lebih terbuka dan demokratis.



Centralized Storage System *Decentralized Storage System*

Gambar 2.1 *Centralized* dan *Decentralized Storage System*

Sumber: Ahmed et al., (2026:797)

Popchev & Radeva, (2024:122) juga berpendapat bahwa *Decentralized Application* (DApp) merupakan aplikasi perangkat lunak yang berjalan di atas jaringan *blockchain* dengan karakteristik utama berupa kode yang bersifat *open source,* mekanisme keamanan berbasis kriptografi, serta pemanfaatan token sebagai bagian dari operasional sistem. Selain itu, DApp juga dapat mengimplementasikan mekanisme konsensus, seperti *Proof-of-Work* (PoW) maupun *Proof-of-Stake* (PoS), untuk mendukung proses validasi dalam jaringan *blockchain*.

Berdasarkan sintesis dari berbagai pendapat tersebut, DApp dapat dipahami sebagai aplikasi yang dibangun di atas infrastruktur *blockchain* dengan prinsip desentralisasi sehingga tidak bergantung pada satu pihak sebagai pusat kendali. Karakteristik tersebut memberikan sejumlah keunggulan, antara lain transparansi, keamanan, integritas data, serta ketahanan terhadap kegagalan terpusat. Oleh karena itu, DApp menjadi salah

satu fondasi utama dalam pengembangan berbagai layanan digital modern yang mengedepankan keamanan, kepercayaan, dan desentralisasi.

5. *Blockchain*

Blockchain merupakan teknologi pencatatan transaksi digital yang bersifat terdesentralisasi dan tidak dapat diubah (*immutable*), sehingga proses penyimpanan serta verifikasi data dilakukan tanpa bergantung pada *server* terpusat. Konsep dasar *blockchain* pertama kali diperkenalkan oleh Satoshi Nakamoto melalui makalah berjudul *Bitcoin: A Peer-to-Peer Electronic Cash System* pada tahun 2008, yang menjelaskan mekanisme pencatatan transaksi tanpa memerlukan pihak ketiga seperti bank atau lembaga keuangan lainnya (Nakamoto, 2008, dalam Widiyasono, 2025:1).

Menurut Raharjo, (2022:1) *blockchain* pada dasarnya dapat dipahami sebagai suatu sistem basis data yang digunakan untuk menyimpan catatan transaksi dan berbagai informasi bernilai. Berbeda dengan basis data konvensional yang mengandalkan pihak ketiga sebagai penyedia kepercayaan, *blockchain* menerapkan mekanisme desentralisasi sehingga proses pencatatan dan verifikasi dilakukan secara kolektif oleh seluruh jaringan. Pendekatan tersebut memungkinkan keamanan, keakuratan, dan transparansi data dijaga tanpa bergantung pada satu lembaga sebagai pengendali utama.

Mengacu pada berbagai pendapat tersebut, *blockchain* dapat dipahami sebagai teknologi penyimpanan data digital yang memanfaatkan mekanisme desentralisasi untuk meningkatkan kepercayaan dalam

pengelolaan informasi. Sifat *immutable* serta proses validasi oleh jaringan menjadikan *blockchain* mampu menjaga integritas, keamanan, dan transparansi data secara lebih baik dibandingkan dengan sistem basis data terpusat. Karakteristik tersebut menjadikan *blockchain* relevan untuk diterapkan pada berbagai aplikasi yang membutuhkan jaminan keaslian dan keamanan data, termasuk sistem *file sharing* karya digital yang dikembangkan dalam penelitian ini.

6. Ethereum

Ethereum merupakan pengembangan dari teknologi *blockchain* yang tidak hanya berfungsi sebagai pencatatan transaksi, tetapi juga mendukung eksekusi *smart contract* serta pengembangan *Decentralized Applications* (DApps) (Raharjo, 2022:41). Dengan kemampuan tersebut, pengembang dapat membangun aplikasi terdesentralisasi tanpa perlu merancang jaringan *blockchain* dari awal. Menurut Syarif et al., (2026:48), Ethereum dirancang sebagai platform *blockchain* yang mampu menjalankan *smart contract* dengan pengelolaan *state* yang lebih kompleks dibandingkan Bitcoin. Platform ini diperkenalkan oleh Vitalik Buterin pada tahun 2013 dan mulai beroperasi secara resmi pada tahun 2015. Dalam ekosistem Ethereum, aset kripto Ether (ETH) digunakan sebagai biaya transaksi atau *gas fee* yang diperlukan untuk menjalankan transaksi maupun eksekusi *smart contract*. Seiring berkembangnya teknologi *blockchain*, Ethereum menjadi salah satu platform yang paling banyak dimanfaatkan dalam pengembangan layanan

decentralized finance (DeFi) maupun berbagai aplikasi Web3 (Hardiyanto et al., 2023:51-52).

Berdasarkan kajian tersebut, Ethereum dapat dipahami sebagai platform *blockchain* yang menyediakan lingkungan untuk menjalankan *smart contract* dan mengembangkan aplikasi terdesentralisasi secara fleksibel. Dukungan terhadap pengelolaan *state*, penggunaan Ether (ETH) sebagai biaya transaksi, serta ekosistem pengembang yang luas menjadikan Ethereum sebagai salah satu fondasi utama dalam pengembangan berbagai solusi berbasis *blockchain*.

7. *Proof of Stake* (PoS)

Proof-of-Stake (PoS) merupakan mekanisme konsensus dalam *blockchain* yang menentukan validator berdasarkan jumlah aset (*stake*) yang dimiliki dan dikunci dalam jaringan. Semakin besar nilai aset yang di-*stake*, semakin besar peluang validator untuk memperoleh hak dalam memvalidasi transaksi dan membentuk blok baru. Berbeda dengan mekanisme berbasis komputasi, proses pemilihan validator pada PoS dilakukan tanpa memerlukan daya komputasi yang tinggi (Homoliak, 2026:15).

Naz & Lee, (2024:1) menjelaskan bahwa PoS merupakan mekanisme konsensus yang dirancang untuk menghemat energi dengan menggantikan proses penambangan berbasis komputasi menjadi sistem *staking* token. Validator dipilih berdasarkan jumlah aset yang dikunci sebagai bentuk partisipasi dalam menjaga keamanan jaringan. Pendapat serupa dikemukakan oleh (Mohamed & Khaifah, 2026:10) yang menyatakan bahwa

PoS menentukan validator berdasarkan kepemilikan aset sehingga konsumsi energi yang dibutuhkan jauh lebih rendah dibandingkan mekanisme *Proof-of-Work* (PoW).

Berdasarkan uraian tersebut, dapat dipahami bahwa PoS merupakan mekanisme konsensus yang mengutamakan kepemilikan aset sebagai dasar pemilihan validator. Pendekatan tersebut tidak hanya meningkatkan efisiensi penggunaan energi, tetapi juga tetap mampu menjaga keamanan dan integritas jaringan melalui partisipasi validator dalam proses validasi transaksi. Oleh karena itu, PoS banyak diadopsi pada *blockchain* modern yang berorientasi pada efisiensi dan keberlanjutan.

8. *Smart Contract*

Smart contract merupakan program komputer yang berjalan pada jaringan *blockchain* dan dirancang untuk mengeksekusi aturan atau perjanjian secara otomatis ketika suatu kondisi terpenuhi. Widiyasono, (2025:67) menjelaskan bahwa konsep *smart contract* pertama kali diperkenalkan oleh Nick Szabo pada tahun 1994 sebagai mekanisme untuk mengurangi ketergantungan terhadap pihak perantara dalam transaksi digital.

Selanjutnya, Syarif et al. (2026:94-95) menyatakan bahwa *smart contract* menjadi salah satu komponen utama yang memungkinkan *blockchain* berkembang dari sistem pencatatan transaksi menjadi platform komputasi terdesentralisasi, Melalui mekanisme tersebut, kode program dapat dijalankan secara otomatis sekaligus diverifikasi secara publik oleh

jaringan *blockchain* sehingga meningkatkan transparansi dan kepercayaan terhadap proses yang berlangsung.

Berdasarkan kajian tersebut, *smart contract* dapat dipahami sebagai mekanisme yang mengatur pelaksanaan suatu proses secara otomatis sesuai aturan yang telah diprogram sebelumnya. Dalam penelitian ini, *smart contract* dimanfaatkan untuk mengelola pencatatan metadata dan proses verifikasi terhadap *file* secara otomatis, transparan, dan sulit dimanipulasi.

9. Solidity

Solidity merupakan bahasa pemrograman tingkat tinggi berorientasi objek yang digunakan untuk mengimplementasikan *smart contract* dan berjalan pada *Ethereum Virtual Machine* (EVM) (Solidity, 2026). Syarif et al. (2026:104) juga menyatakan bahwa Solidity menjadi standar utama dalam pengembangan DApps karena memiliki dukungan komunitas yang luas, dokumentasi yang matang, serta kompatibilitas langsung dengan EVM. Solidity juga bersifat berorientasi objek, sehingga *smart contract* dapat diperlakukan sebagai entitas yang memiliki state dan fungsi.

Mengacu pada berbagai pendapat tersebut, Solidity memiliki peran penting dalam pengembangan aplikasi berbasis Ethereum karena menyediakan sarana untuk membangun *smart contract* yang dapat dijalankan secara terdesentralisasi. Meskipun demikian, implementasinya tetap memerlukan pemahaman mengenai arsitektur EVM, mekanisme *gas fee*, serta praktik pengembangan yang aman agar kontrak yang dibuat dapat berjalan secara optimal.

10. Jaringan *Peer to Peer* (P2P)

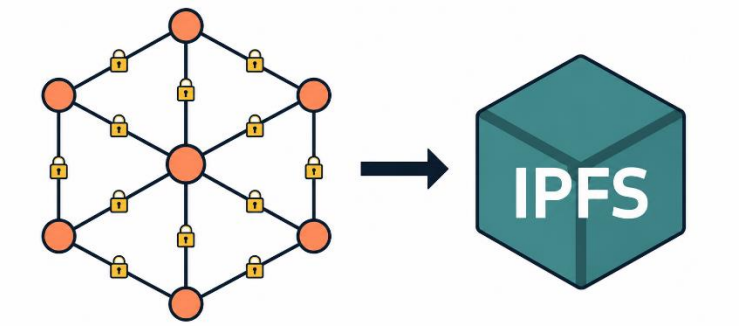
Jaringan *Peer-to-Peer* (P2P) merupakan model jaringan terdistribusi yang memungkinkan setiap perangkat atau node saling terhubung dan berbagi data tanpa bergantung sepenuhnya pada *server* pusat. Trautwein et al. (2022:12) menjelaskan bahwa IPFS merupakan jaringan P2P berbasis *content-addressable* untuk menyimpan dan mengirim data secara terdistribusi. Sementara itu, Syarif et al. (2026:75-76) menjelaskan bahwa jaringan P2P memungkinkan ribuan hingga jutaan node tersebar secara global untuk berkomunikasi, menyebarkan data, dan mencapai kesepakatan tanpa bergantung pada otoritas pusat.

Berdasarkan berbagai pendapat tersebut, jaringan P2P dapat dipahami sebagai infrastruktur komunikasi yang mendukung pertukaran data secara langsung antarpengguna dalam lingkungan terdistribusi. Karakteristik tersebut menjadikan jaringan P2P sangat sesuai untuk mendukung implementasi *blockchain* dan IPFS karena mampu meningkatkan ketersediaan data, mengurangi ketergantungan terhadap *server* tunggal, serta memperkuat keandalan sistem *file sharing* terdesentralisasi.

11. *Interplanetary file system* (IPFS)

Menurut Rajaraman dan Ullman dalam Stiller et al. (2021:85), IPFS adalah sistem yang dirancang untuk menyimpan dan mengakses berbagai data seperti berkas, situs web, serta aplikasi melalui jaringan terdistribusi. Alsudani et al. dalam Al-Kaabi & Abdullah (2023:587) Berpendapat bahwa sistem ini mengidentifikasi berkas berdasarkan isinya melalui pembuatan

tanda *hash* yang unik, sehingga memungkinkan pengecekan duplikasi data sebelum disimpan untuk menghemat ruang penyimpanan.



Gambar 2.2 *Inter-Planetary File System* (IPFS) P2P *Structure*

Sumber: (Al-Kaabi & Abdullah, 2023:588)

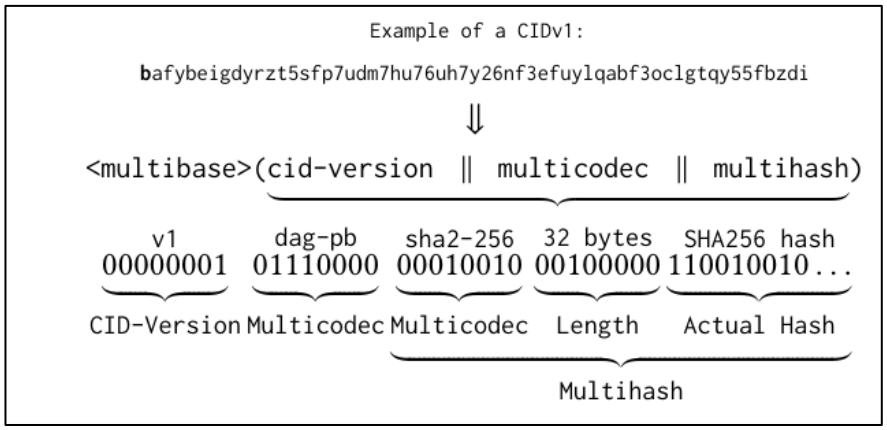
Berdasarkan berbagai pendapat tersebut, IPFS dapat dipahami sebagai sistem penyimpanan terdistribusi yang mengutamakan identifikasi data berdasarkan isi (*content-addressed storage*) dibandingkan lokasi penyimpanannya. Karakteristik tersebut memungkinkan data disimpan dan diakses secara efisien melalui jaringan *peer-to-peer*, sekaligus menjaga integritas data melalui penggunaan *hash* yang unik. Oleh karena itu, IPFS menjadi teknologi yang sesuai untuk mendukung penyimpanan berkas pada sistem *file sharing* berbasis *blockchain*.

12. *Content Identifier* (CID)

Content Identifier (CID) merupakan identitas unik yang digunakan dalam IPFS untuk mengenali dan mengambil suatu konten berdasarkan isi data. Doan et al. (2022:3) menjelaskan bahwa IPFS menggunakan mekanisme *multihash* sebagai CID setiap objek, sehingga suatu berkas dapat diunduh dari beberapa *peer* yang menyimpan salinan data tersebut di dalam jaringan. Pendapat serupa yang dikemukakan oleh Al-Kaabi & Abdullah (2023:587),

juga menjelaskan bahwa setiap berkas yang disimpan pada IPFS akan diberikan nilai *hash* unik sehingga pengguna dapat mengakses berkas tersebut melalui alamat *hash* yang dimilikinya

Trautwein et al. (2022:2) juga menjelaskan bahwa konsep utama dari IPFS adalah skema *content-based addressing* yang menggunakan CID berbasis *hash*, sehingga nama suatu konten dapat dipisahkan dari lokasi penyimpanannya.



Gambar 2.3 Struktur CID

Sumber (Trautwein et al., 2022:3)

Berdasarkan berbagai pendapat tersebut, *Content Identifier* (CID) dapat dipahami sebagai identitas unik yang digunakan IPFS untuk mengidentifikasi dan menemukan suatu konten berdasarkan isi data, bukan berdasarkan lokasi penyimpanannya. Dengan mekanisme *content-based addressing*, CID memungkinkan berkas diakses dari berbagai *peer* selama konten tersebut masih tersedia di dalam jaringan IPFS.

13. Algoritma *Hashing* SHA-256

Menurut Ahmad et al., (2025:18-21) dalam buku *Fostering Machine Learning and IoT for Blockchain Technology,* SHA-256 adalah fungsi *hash*

kriptografi yang mengubah data input dengan panjang yang bervariasi menjadi nilai *hash* tetap sebesar 256 bit. Sebelum proses *hashing* dilakukan, data akan melalui tahapan *padding* sehingga panjang data menjadi kelipatan 512 bit, kemudian diproses dalam blok-blok berukuran 512 bit.

Proses komputasi SHA-256 memanfaatkan delapan nilai *hash* awal (*initial hash values*) yang diperoleh dari akar kuadrat bilangan prima serta 64 konstanta yang berasal dari akar pangkat tiga bilangan prima. Selama proses tersebut berlangsung, algoritma menerapkan berbagai operasi kriptografi, seperti fungsi Boolean (x, y, z) dan $Maj(x, y, z)$, operasi rotasi bit, serta penjumlahan modulo 2^{32}.

Karakteristik utama SHA-256 terletak pada sifat *one-way function*, yang membuat nilai *hash* tidak dapat dikembalikan menjadi data asli, serta *collision resistance*, yaitu sangat kecil kemungkinannya dua data berbeda menghasilkan nilai *hash* yang sama. Selain itu, SHA-256 juga memiliki sifat *avalanche effect*, yaitu perubahan sekecil apa pun pada data masukan akan menghasilkan perubahan nilai *hash* yang sangat berbeda. Contoh hasil komputasi SHA-256 ditunjukkan pada Tabel 2.1.

Tabel 2.1 Contoh Hasil Komputasi Algoritma SHA-256.

Input $(\boldsymbol{M})$	***Hash* SHA-256** $(\boldsymbol{H}(\boldsymbol{M}))$
0	5feceb66ffc86f38d952786c6d696c79c2dbc239dd4e91b46729d73a27fb57e9
1	6b86b273ff34fce19d6b804eff5a3f5747ada4eaa22f1d49c01e52ddb7875b4b
00	f1534392279bddbf9d43dde8701cb5be14b82f76ec6607bf8d6ad557f60f304e
11	4fc82b26aecb47d2868c4efbe3581732a3e7cbcc6c2efb32062c08170a05eeb8
a	ca978112ca1bbdcafac231b39a23dc4da786eff8147c4e72b9807785afee48bb

Input (M)	***Hash* SHA-256 ($H(M)$)**
b	3e23e8160039594a33894f6564e1b1348bbd7a0088d42c4acb73eeaed59c009d
A	559aead08264d5795d3909718cdd05abd49572e84fe55590eef31a88a08fdffd
B	df7e70e5021544f4834bbee64a9e3789febc4be81470df629cad6ddb03320a5c

Sumber: (Ahmad et al., 2025:21)

Karakteristik *one-way function* dan *collision resistance* menjadikan SHA-256 sebagai algoritma yang banyak dimanfaatkan dalam berbagai implementasi *blockchain*. Pada penelitian ini, SHA-256 digunakan untuk menghasilkan nilai *hash* yang merepresentasikan identitas setiap karya digital. Nilai *hash* tersebut dimanfaatkan sebagai dasar proses verifikasi dengan membandingkannya terhadap data yang telah tercatat pada *blockchain*. Apabila nilai *hash* telah ditemukan, sistem dapat mengidentifikasi bahwa karya digital tersebut telah terdaftar sehingga proses unggah dapat dicegah untuk menghindari duplikasi. Selain itu, mekanisme ini juga memungkinkan pengguna melakukan pengecekan keberadaan suatu karya digital pada *blockchain* tanpa perlu mengunggah berkas ke dalam sistem.

14. Metamask

MetaMask merupakan dompet digital (*crypto wallet*) berbasis *blockchain* yang berfungsi sebagai penghubung antara pengguna dengan aplikasi terdesentralisasi (DApp). Berdasarkan dokumentasi resmi MetaMask (2026), fitur *MetaMask Connect* memungkinkan pengguna menghubungkan akun *blockchain* yang dimiliki melalui ekstensi maupun aplikasi *mobile* sehingga interaksi dengan DApp dapat dilakukan pada berbagai perangkat.

Torres et al. (2023:1) menyatakan bahwa e-*wallet* memiliki peran penting dalam Web3 karena bertindak sebagai antarmuka yang menghubungkan pengguna dengan aplikasi terdesentralisasi. Selain itu, Yan et al. (2024:4) menjelaskan bahwa e-*wallet* seperti MetaMask juga digunakan dalam proses autentikasi Web3, melalui permintaan tanda tangan digital (*sigital signature*) maupun persetujuan transaksi oleh pengguna.

Berdasarkan kajian tersebut, MetaMask dapat dipahami sebagai komponen penting dalam ekosistem *blockchain* karena tidak hanya berfungsi sebagai penyimpan aset digital, tetapi juga sebagai sarana autentikasi dan otorisasi ketika pengguna berinteraksi dengan aplikasi terdesentralisasi. Dalam penelitian ini, MetaMask digunakan untuk menghubungkan akun pengguna dengan sistem, melakukan autentikasi melalui *digital signature*, serta memberikan persetujuan terhadap transaksi *blockchain*. Oleh karena itu, aspek keamanan penggunaan MetaMask tetap perlu diperhatikan, terutama dalam setiap proses koneksi, permintaan tanda tangan digital, maupun persetujuan transaksi.

15. *Digital Signature*

Digital signature merupakan mekanisme kriptografi yang digunakan untuk membuktikan keaslian, integritas, dan persetujuan terhadap suatu data atau transaksi digital. *National Institute of Standards and Technology (2023:3)* menjelaskan bahwa *digital signature* merupakan hasil transformasi kriptografis terhadap data yang menyediakan mekanisme untuk memverifikasi autentikasi asal data, menjaga integritas data, serta menjamin

non-repudiation atau ketidakmampuan penandatangan untuk menyangkal keterlibatannya. Dalam konteks *blockchain*, Syarif et al. (2026:26) menjelaskan bahwa tanda tangan digital dibangun di atas kriptografi kunci asimetris (*asymmetric cryptography*). Mekanisme ini memastikan bahwa hanya pemilik sah dari pasangan kunci kriptografi yang dapat memberikan persetujuan terhadap suatu transaksi tanpa perlu mengungkapkan *private key* yang dimilikinya.

Berdasarkan berbagai pendapat tersebut, *digital signature* berperan sebagai mekanisme autentikasi yang menjamin identitas pengguna, integritas data, serta *non-repudiation* dalam sistem berbasis *blockchain*. Pada penelitian ini, *digital signature* digunakan untuk memverifikasi identitas pengguna ketika melakukan autentikasi maupun memberikan persetujuan terhadap proses unggah karya digital ke *blockchain*. Dengan demikian, setiap aksi yang dilakukan dapat dipastikan berasal dari pemilik akun yang sah sehingga risiko penyalahgunaan akses maupun pemalsuan identitas dapat diminimalkan.

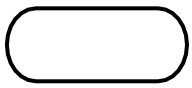
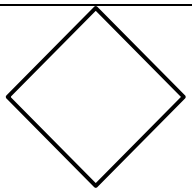
16. *Flowchart*

Flowchart merupakan representasi visual yang digunakan untuk menggambarkan urutan langkah atau proses dalam suatu sistem. Setiap tahapan direpresentasikan menggunakan simbol-simbol grafis yang dihubungkan oleh garis atau panah sehingga alur proses dapat dipahami secara sistematis Khairunisa et al. (2023:28). Pendapat lain mengatakan bahwa *flowchart* adalah bentuk visualisasi algoritma yang digunakan untuk menggambarkan penyelesaian suatu permasalahan secara terstruktur. Selain

mempermudah pemahaman terhadap logika proses, *flowchart* juga berfungsi sebagai media komunikasi antar anggota tim pengembang sehingga implementasi sistem dapat dilakukan sesuai dengan rancangan yang telah ditetapkan. (Susanto & Syukron, 2020:10).

Berdasarkan berbagai pendapat tersebut, *flowchart* dapat dipahami sebagai diagram yang merepresentasikan alur kerja atau algoritma suatu sistem menggunakan simbol-simbol standar. Penyajian proses dalam bentuk visual membantu pengembang memahami urutan aktivitas secara lebih jelas, memudahkan proses analisis, serta mengurangi kemungkinan terjadinya kesalahan pada tahap implementasi.

Tabel 2.2 Simbol – Simbol *Flowchart*

Simbol	Nama	Fungsi
	Flowline	Simbol berbentuk panah yang menunjukkan arah perpindahan proses atau aliran data dari satu langkah ke langkah berikutnya.
	Terminal / *Terminator*	Simbol yang digunakan untuk menandai titik awal (*start*) maupun akhir (*end*) dari suatu proses atau algoritma.
	Proses /*Process*	Simbol yang menunjukkan aktivitas, operasi, atau tahapan pengolahan data yang dilakukan oleh sistem.
	Keputusan / *Decision*	Simbol yang digunakan untuk menggambarkan proses pengambilan keputusan berdasarkan suatu kondisi tertentu sehingga menghasilkan percabangan alur.

Simbol	Nama	Fungsi
	Dokumen	Simbol yang menunjukkan adanya dokumen atau informasi yang digunakan, diproses, maupun dihasilkan selama proses berlangsung.
	*Input* / *Output*	Simbol yang merepresentasikan proses penerimaan masukan (*input*) atau penyajian keluaran (*output*) dari suatu sistem.

Sumber: Khairunisa et al. (2023:33-36)

17. *Unified Model Language* (UML)

Unified Modeling Language (UML) merupakan bahasa pemodelan visual yang digunakan untuk merancang, memvisualisasikan, mendokumentasikan, serta menggambarkan struktur dan perilaku suatu sistem perangkat lunak. Gunawan et al. (2023:51) menjelaskan bahwa UML membantu pengembang dalam memahami rancangan sistem sebelum tahap implementasi dilakukan sehingga proses pengembangan menjadi lebih terarah.

Pendapat serupa disampaikan oleh Sumirat et al. (2023:73) yang mengatakan bahwa UML adalah bahasa berbasis grafis yang digunakan untuk memvisualisasikan, menspesifikasikan, membangun, dan mendokumentasikan perangkat lunak berorientasi objek. UML tidak hanya berfungsi sebagai alat dokumentasi, tetapi juga dapat diimplementasikan dengan berbagai bahasa pemrograman seperti Java, C++, *Visual Basic*, maupun basis data berorientasi objek.

Mengacu pada berbagai pendapat tersebut, UML dapat diartikan sebagai pemodelan visual yang digunakan untuk menggambarkan struktur,

fungsi, serta interaksi antar komponen dalam suatu sistem perangkat lunak. Penggunaan UML membantu pengembang maupun pemangku kepentingan memperoleh pemahaman yang sama terhadap rancangan sistem sebelum proses pengembangan dan implementasi dilakukan. Dalam penelitian ini, UML digunakan sebagai alat pemodelan untuk menggambarkan kebutuhan fungsional, alur aktivitas, interaksi antar komponen, serta struktur sistem yang dikembangkan. Adapun jenis UML yang digunakan penulis meliputi:

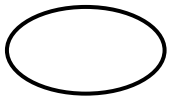
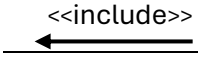
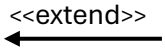
a. *Use Case* Diagram

Menurut Gunawan et al. (2023:55) *Use Case Diagram* merupakan salah satu diagram UML yang memodelkan interaksi antara pengguna dan sistem dengan tujuan menampilkan kebutuhan fungsional utama yang harus disediakan sistem sesuai perspektif pengguna. Pendapat tersebut sejalan dengan Sumirat et al. (2023:80-81) yang menjelaskan bahwa diagram ini merupakan representasi visual yang menghubungkan interaksi antara aktor eksternal dan sistem dalam pelaksanaan proses bisnis. Selain merepresentasikan fungsi sistem, diagram ini juga berperan dalam proses identifikasi aktor, verifikasi kebutuhan fungsional, spesifikasi antarmuka, serta menjadi media komunikasi antara analis sistem dan pemangku kepentingan.

dapat dikatakan bahwa *Use Case Diagram* adalah diagram UML yang digunakan untuk memodelkan interaksi antara pengguna dan sistem dalam bentuk kebutuhan fungsional. Diagram ini memberi gambaran mengenai fitur yang disediakan sistem, aktor yang terlibat, serta hubungan

di antara keduanya sehingga membantu proses analisis, perancangan, dan komunikasi kebutuhan sistem sebelum tahap implementasi dilakukan.

Tabel 2.3 Simbol *Use Case* Diagram

Simbol	Elemen	Keterangan
	Aktor	Simbol yang merepresentasikan pengguna atau entitas eksternal yang berinteraksi dengan sistem.
	*Use Case*	Simbol yang menggambarkan fitur yang disediakan sistem sebagai respons terhadap interaksi yang dilakukan oleh aktor.
	Assocation	Simbol yang menunjukkan komunikasi antara aktor dengan *use case* yang dijalankan dalam sistem.
	*Include*	Simbol yang menunjukkan bahwa *use case* memanggil fungsionalitas *use case* lain sebagai bagian dari prosesnya.
	*Extend*	Simbol yang menunjukkan adanya fungsionalitas tambahan yang dijalankan apabila kondisi tertentu terpenuhi pada *use case* utama.

Sumber: Gunawan et al. (2023:56)

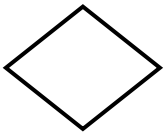
b. *Activity* Diagram

Activity Diagram merupakan salah satu diagram UML yang digunakan untuk memodelkan urutan aktivitas atau alur kerja dalam suatu sistem maupun proses bisnis. Hariyanto dalam Khairunisa et al. (2023:175-176) menjelaskan bahwa diagram ini menggambarkan rangkaian aktivitas yang terjadi selama suatu proses berlangsung sehingga alur kerja sistem dapat dipahami secara lebih jelas. Gunawan et al. (2023:64) juga menyatakan bahwa *Activity Diagram* berfungsi untuk memvisualisasikan jalannya aktivitas dalam sistem, sekaligus membantu

mengidentifikasi tahapan proses yang masih dapat ditingkatkan guna memperoleh efisiensi yang lebih baik.

Berdasarkan pendapat di atas, dapat disimpulkan bahwa *Activity Diagram* dapat dipahami sebagai diagram UML yang menggambarkan urutan aktivitas beserta aliran proses yang terjadi dalam suatu sistem. Diagram ini memudahkan pengembang dalam memahami logika proses, menganalisis alur kerja, serta mengidentifikasi peluang perbaikan sebelum sistem diimplementasikan.

Tabel 2.4 Simbol *Activity* Diagram

Simbol	Elemen	Keterangan
	*Activity*	Simbol yang menunjukkan aktivitas atau proses yang dilakukan dalam suatu alur kerja sistem.
	*Initial Node*	Simbol yang menandai titik awal dimulainya suatu aktivitas atau proses dalam diagram.
	*Activity Final Node*	Simbol yang menandai titik akhir atau selesai dalam seluruh rangkaian suatu aktivitas.
	*Decision*	Simbol yang digunakan untuk menggambarkan proses pengambilan keputusan dan menghasilkan percabangan alur berdasarkan kondisi.
	Line Connector	Simbol yang menghubungkan satu aktivitas dengan aktivitas lainnya sehingga membentuk alur proses yang berkesinambungan.

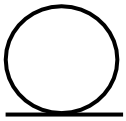
Sumber: Gunawan et al. (2023:65)

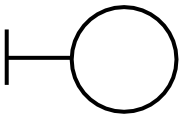
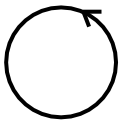
c. *Sequence* Diagram

Menurut Gunawan et al. (2023:60) *Sequence* Diagram adalah diagram yang merepresentasikan interaksi antar objek dalam sistem secara berurutan untuk menggambarkan alur logika suatu fitur atau fungsi. diagram ini menunjukkan aliran pesan (*message*) yang dikirimkan antar objek untuk merepresentasikan logika suatu fitur atau fungsi pada sistem. Sumirat et al. (2023:86) juga berpendapat bahwa *sequence* diagram dapat menggambarkan interaksi secara rinci antar objek, termasuk urutan pengiriman pesan. Objek-objek yang terlibat umumnya disusun dari kiri ke kanan sesuai dengan urutan komunikasi yang terjadi selama proses berlangsung.

Berdasarkan berbagai pendapat tersebut, *Sequence Diagram* dapat dipahami sebagai diagram UML yang memvisualisasikan interaksi antarobjek secara kronologis. Diagram ini membantu menggambarkan alur komunikasi, urutan pemanggilan fungsi, serta keterlibatan setiap objek dalam menjalankan suatu proses di dalam sistem.

Tabel 2.5 Simbol *Sequence* Diagram

Simbol	Elemen	Keterangan
	Aktor	Simbol yang merepresentasikan pengguna atau entitas eksternal yang berinteraksi dengan sistem.
	*Entity Class*	Simbol yang menggambarkan objek yang bertugas menyimpan atau mengelola data dalam sistem.

Simbol	Elemen	Keterangan
	*Boundary Class*	Simbol yang Menunjukkan komponen antarmuka yang menjadi penghubung antara pengguna dengan sistem.
	*Control Class*	Simbol yang mengatur alur logika proses serta mengoordinasikan komunikasi antara *boundary class* dan *entity class*.
	*A Focus of Control & A Life Line*	Simbol yang menunjukkan periode ketika suatu objek menjalankan proses atau menerima kendali selama interaksi berlangsung.
	A Message	Simbol yang menggambarkan pengiriman informasi, perintah, atau pemanggilan fungsi dari satu objek ke objek lainnya.

Sumber: Gunawan et al. (2023:60-61)

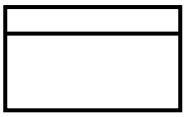
d. *Class* Diagram

Class diagram merupakan jenis diagram struktur dalam UML yang digunakan untuk menampilkan susunan kelas, atribut, metode, dan hubungan antar objek. Diagram ini bersifat statis karena hanya fokus pada representasi struktur sistem, bukan pada interaksi atau alur proses yang terjadi, Sumirat et al. (2023:88-89). Pendapat lain dari Gunawan et al. (2023:57) juga menyatakan bahwa *Class* diagram digunakan untuk memodelkan struktur kelas beserta hubungan antar kelas sehingga memberikan gambaran menyeluruh mengenai objek-objek yang membentuk sistem dan keterkaitannya.

Berdasarkan pendapat di atas, dapat disimpulkan bahwa *Class* diagram adalah diagram UML yang digunakan untuk merepresentasikan struktur kelas dalam suatu sistem, meliputi atribut, metode, serta hubungan

antar kelas atau objek. Diagram ini membantu pengembang memahami organisasi komponen sistem dan menjadi acuan dalam proses implementasi perangkat lunak berorientasi objek.

Tabel 2.6 Simbol *Class* Diagram

Simbol	Elemen	Keterangan
	Association	Simbol ini menunjukkan hubungan atau keterkaitan antara dua kelas yang saling berinteraksi di dalam sistem.
	*Class*	Simbol ini merepresentasikan *blueprint* suatu objek yang berisi atribut dan metode sebagai dasar pembentukan objek.

Sumber: Gunawan et al. (2023:57-58)

18. Pengujian Sistem

Pengujian sistem merupakan tahapan yang dilakukan untuk memastikan bahwa sistem yang dikembangkan dapat berjalan sesuai dengan rancangan yang telah dibuat. Pada penelitian ini, pengujian sistem dilakukan menggunakan *black box testing* dan *white box testing*. *Black box testing* digunakan untuk menguji kesesuaian fungsi sistem dari sisi pengguna, sedangkan *white box testing* digunakan untuk menguji beberapa logika internal program, alur kode, serta percabangan pada fungsi-fungsi utama pada sistem. Dengan menggunakan kedua metode tersebut, pengujian diharapkan dapat memastikan bahwa sistem berjalan dengan baik dari sisi fungsionalitas maupun dari sisi struktur logika program.

a. *Black Box Testing*

Black box testing merupakan metode pengujian yang berfokus pada evaluasi fungsional tanpa memerlukan akses terhadap struktur kode program. Corradini et al. (2023:1) menjelaskan bahwa pendekatan *black box* banyak digunakan dalam pengujian otomatis karena tidak memerlukan akses terhadap *source code* dan dapat menguji sistem secara independen dari arsitektur internalnya.

Berdasarkan uraian tersebut, dapat disimpulkan bahwa *black box testing* sesuai digunakan untuk menguji fitur sistem dari sisi pengguna. Dalam penelitian ini, *black box testing* digunakan untuk menguji fitur utama pada sistem ini. Setiap fitur diuji berdasarkan skenario tertentu untuk memastikan bahwa input yang diberikan menghasilkan output sesuai dengan yang ditentukan.

b. *White Box Testing*

Shiddiq (2022:2) menjelaskan bahwa *white box testing* dikembangkan berdasarkan kode program, sehingga penguji perlu memahami aliran kontrol dan aliran data pada program. Derakhshanfar et al. (2022:2) juga menjelaskan bahwa pengujian *white box* dapat memanfaatkan *control flow graph* (CFG), yaitu representasi kode program dalam bentuk *node* dan *edge* untuk menggambarkan kemungkinan alur eksekusi.

Salah satu teknik *white box testing* adalah basis *path testing*. Teknik ini dilakukan dengan membuat CFG, *menghitung cyclomatic*

complexity, menentukan *independent path*, dan menyusun *test case* berdasarkan jalur tersebut. Shiddiq (2022:1-2) menjelaskan bahwa *cyclomatic complexity* digunakan untuk mengukur kompleksitas program dan menentukan jalur independen yang akan digunakan dalam skenario pengujian, dengan rumus 2.1 sebagai berikut.

$$V(G) = E - N + 2 \quad (2.1)$$

Berdasarkan uraian tersebut, dapat disimpulkan bahwa *white box testing* sesuai digunakan untuk menguji logika internal program, terutama pada fungsi yang memiliki percabangan. Dalam penelitian ini, *white box testing* difokuskan pada empat fungsi utama yang berhubungan langsung dengan tujuan penelitian, yaitu proses *upload file* ke IPFS, pencatatan data *file* ke *blockchain*, validasi keberadaan *file* pada *blockchain*, dan verifikasi transaksi *blockchain*. Dengan melalui tahapan pembuatan CFG, perhitungan *cyclomatic complexity*, penentuan *independent path*, dan penyusunan *test case*.

B. Kajian Empiris

Penelitian yang dilakukan oleh Tai & Thanh (2024) mengembangkan sistem *Digital Rights Management* (DRM) dengan mengintegrasikan *blockchain*, *InterPlanetary File System* (IPFS), dan *watermarking* untuk melindungi hak cipta konten digital. *Blockchain* dimanfaatkan untuk pencatatan kepemilikan dan lisensi, sedangkan IPFS digunakan sebagai media penyimpanan terdesentralisasi. Sejalan dengan itu, Liu et al. (2024) menerapkan *blockchain* dan IPFS juga pada sistem perlindungan hak cipta digital dengan

dukungan *digital signature* untuk verifikasi kepemilikan. Kedua penelitian tersebut menunjukkan bahwa integrasi *blockchain* dan IPFS mampu meningkatkan keamanan, integritas, dan perlindungan kepemilikan karya digital.

Selanjutnya, Makhtar et al. (2024) menjelaskan pemanfaatan *smart contract* Ethereum dan IPFS pada sistem manajemen *e-book*, di mana *file* disimpan pada IPFS sedangkan metadata dicatat pada *blockchain*. Hasil serupa juga dikemukakan oleh Saif et al. (2024) yang menyatakan bahwa *blockchain* lebih sesuai digunakan untuk menyimpan *hash* atau *fingerprint* data, sedangkan data utama disimpan secara *off-chain* melalui IPFS. Kedua penelitian tersebut memperkuat konsep pemisahan penyimpanan *file* dan metadata pada sistem berbasis *blockchain*.

Selain itu, Madapati & Pradhan (2025) memanfaatkan *blockchain* dan IPFS untuk perlindungan hak cipta video melalui pencatatan kepemilikan menggunakan *smart contract* Ethereum. Di sisi lain, Ahmad A. P. (2024) menunjukkan bahwa kombinasi *blockchain* dan IPFS dapat digunakan sebagai penyimpanan terdistribusi untuk mengurangi risiko *single point of failure* pada sistem terpusat.

Berdasarkan penelitian-penelitian tersebut, *blockchain* dan IPFS terbukti berpotensi mendukung pengelolaan *file* digital yang lebih aman, transparan, dan terdesentralisasi. Oleh karena itu, penelitian ini mengimplementasikan *blockchain* Ethereum dan IPFS pada sistem *file sharing* karya digital berbasis website dengan mengintegrasikan pencatatan kepemilikan,

penyimpanan *file* terdesentralisasi, pengelolaan hak akses, dan verifikasi integritas *file*.

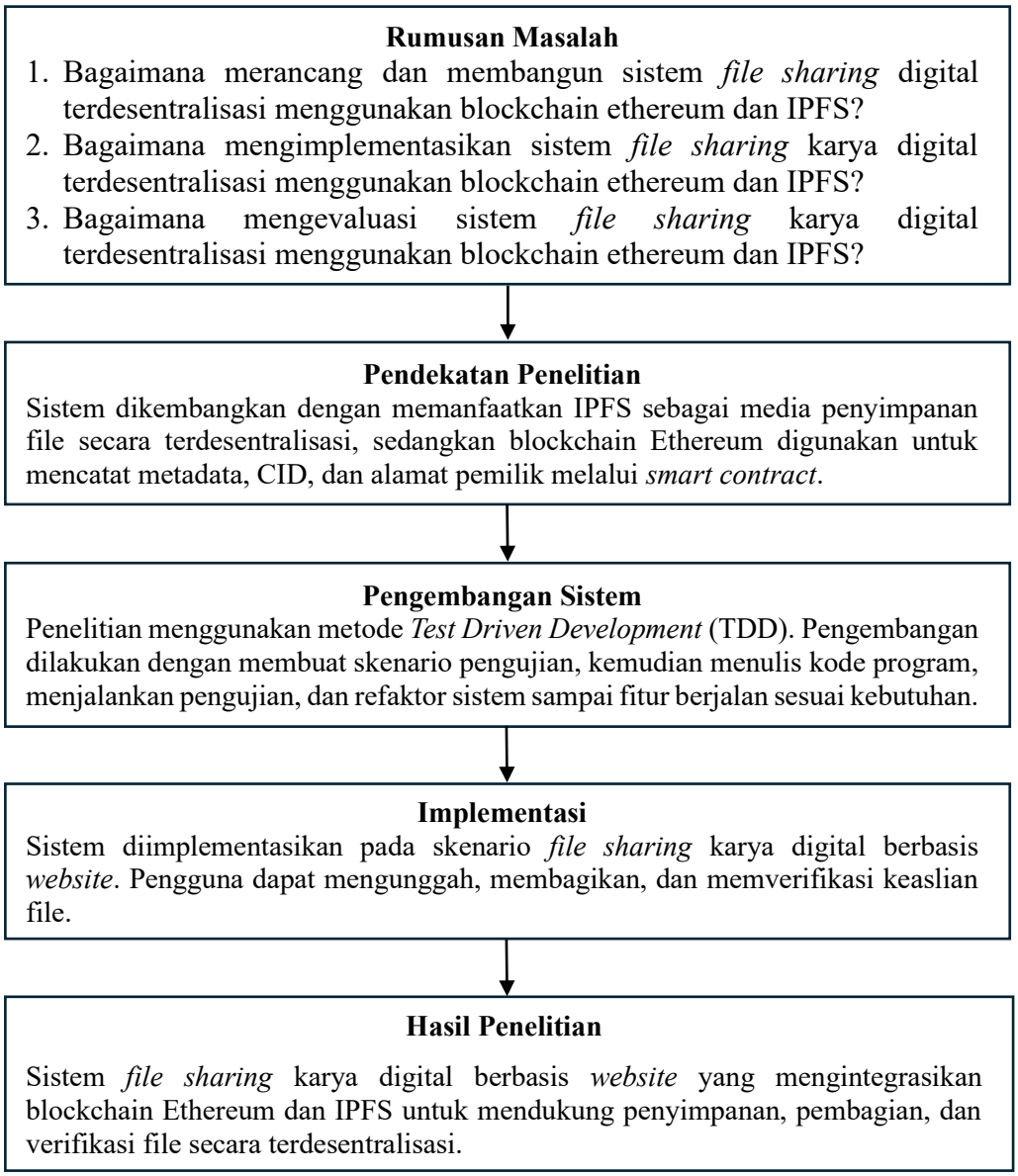
C. Kerangka Berpikir

Permasalahan dalam penelitian ini adalah sistem *file sharing* karya digital yang masih banyak menggunakan penyimpanan terpusat (*centralized storage*). Model penyimpanan tersebut memiliki kelemahan, seperti risiko *single point of failure*, manipulasi data, serta penyalahgunaan karya digital yang dapat mengakibatkan perubahan kepemilikan maupun distribusi tanpa izin. Oleh karena itu, diperlukan sistem *file sharing* yang mampu menjamin keamanan, transparansi, dan integritas data.

Untuk mengatasi permasalahan tersebut, penelitian ini memanfaatkan teknologi *blockchain* Ethereum dan *InterPlanetary File System* (IPFS). *Blockchain* Ethereum digunakan untuk mencatat metadata, *hash* SHA-256, dan informasi kepemilikan *file* melalui *smart contract*, sedangkan IPFS digunakan sebagai media penyimpanan *file* secara terdesentralisasi. Integrasi kedua teknologi tersebut diharapkan mampu meningkatkan keamanan penyimpanan dan menjaga integritas data.

Penelitian ini mengembangkan sistem *file sharing* karya digital terdesentralisasi berbasis website. Pada sistem yang dikembangkan, pengguna mengunggah *file* melalui website, kemudian *file* disimpan pada IPFS sehingga menghasilkan *Content Identifier* (CID). Selanjutnya, CID, *hash* SHA-256, dan metadata *file* dicatat pada *blockchain* Ethereum melalui *smart contract* sehingga dapat digunakan untuk proses verifikasi integritas dan kepemilikan *file*.

Hasil yang diharapkan dari penelitian ini adalah sistem *file sharing* karya digital terdesentralisasi yang mampu mendukung penyimpanan, pengelolaan, dan pembagian karya digital secara lebih aman, transparan, dan terintegrasi. Berdasarkan alur pemikiran tersebut, kerangka berpikir penelitian ini ditunjukkan pada Gambar 2.4.



Gambar 2.4 Kerangka Berpikir