BAB II

LANDASAN TEORI

2.1 Teori Umum

2.1.1 Arsip

Arsip adalah rekaman atau catatan dari setiap aktivitas yang dilakukan oleh suatu organisasi. Catatan ini, yang dikenal sebagai dokumen atau informasi terekam, bisa berupa tulisan, gambar, maupun suara (Setianingsih, 2022). Menurut Undang-Undang Republik Indonesia Nomor 43 Tahun 2009 Tentang Kearsipan menerangkan bahwa arsip adalah rekaman kegiatan atau peristiwa dalam berbagai bentuk dan media sesuai dengan perkembangan teknologi informasi dan komunikasi yang dibuat dan diterima oleh lembaga negara, pemerintahan daerah, lembaga pendidikan, perusahaan, organisasi politik, organisasi kemasyarakatan, dan perseorangan dalam pelaksanaan kehidupan bermasyarakat, berbangsa, dan bernegara.

2.1.2 Manajemen Arsip

Odgers mendefinisikan manajemen arsip sebagai sebuah proses yang mencakup pengawasan, penyimpanan, dan pengamanan dokumen serta arsip. Manajemen ini memastikan bahwa dokumen dan arsip terlindungi dan dapat diakses sesuai kebutuhan (Salehah, 2020). Menurut Aryani (2019) dalam Aprilia dkk. (2020) manajemen arsip selalu mengikuti perubahan dan perkembangan teknologi yang terjadi di masyarakat. Berikut perbedaan manajemen kearsipan tradisional (manual) dengan manajemen kearsipan elektronik.

Comparison	Traditional archive	Electronic Archive
Document Recovery	minutes to hours	Seconds
Missing Documents	Sometimes	Not at all
Document Saving	Minutes to hours	Seconds
Document Sending	Long time and sometimes resent back. The fax machine gives unclear copies	Easy and faster to be sent resent back. The fax machine gives unclear copies.
Document Sharing	A lot of copies	Just taking the authority
Dedicated places	Large and expensive	Small and cheap
Customer Services	Takes a time	Less time

Gambar 2.1 Perbedaan Kearsipan Tradisional dan Kearsipan Elektronik

(Sumber: Laarfi, 2020)

Gambar 2.1 membandingkan kearsipan tradisional dan elektronik. Kearsipan tradisional memakan waktu dalam pemulihan dan pengiriman dokumen, berisiko kehilangan arsip, membutuhkan ruang besar, serta layanan pelanggan yang lambat. Sebaliknya, kearsipan elektronik memungkinkan pencarian dan pengiriman dokumen dalam hitungan detik, minim risiko kehilangan, efisien dalam penyimpanan, dan mendukung layanan pelanggan yang cepat.

2.1.3 Manajemen Arsip Elektronik

Mulyantono (2021) dalam Harky dkk. (2022) menyebutkan istilah manajemen arsip digital sebagai *Electronic Records Management System* (ERMS). *Electronic records management system* merupakan bagian dari *Business Information System* (BIS) yang memiliki tujuan utama untuk menangkap dan mengelola arsip digital. Sistem ini dirancang khusus untuk mengelola proses penciptaan, penggunaan, pemeliharaan, dan pemusnahan arsip digital dengan tujuan menyediakan bukti dari aktivitas bisnis.

2.1.4 *Blockchain*

2.1.4.1 Konsep *Blockchain*

Laurence (2017) dalam Putri (2021) menjelaskan konsep *blockchain*. *Blockchain* terdiri dari blok dan rantai. Blok-blok tersebut adalah data atau informasi digital yang saling terhubung satu sama lain dan disimpan dalam basis data publik yang disebut rantai. *Blockchain* juga dikenal sebagai buku besar bersama yang terdistribusi, yang bersifat permanen, di mana data atau informasi yang telah disimpan dalam suatu blok tidak dapat diubah oleh siapapun. Permanen dalam konteks *blockchain* berarti bahwa jika ingin melakukan perubahan pada data yang telah disimpan, harus mendapatkan persetujuan mayoritas pengguna jaringan *blockchain* terlebih dahulu. Setelah itu, akan dibuat blok baru yang berisi data perubahan tersebut, sehingga blok yang sudah ada sebelumnya tidak dihapus.

2.1.4.2 Karakteristik *Blockchain*

Menurut Luca dkk. (2022) dalam Sari dan Gelar (2024) menerangkan bahwa terdapat enam karakteristik utama dari *blockchain* sebagai berikut:

1. Desentralisasi

   Sistem *blockchain* dapat diakses dan dipelihara oleh semua *node*, memungkinkan setiap transaksi atau pertukaran data divalidasi oleh semua pihak tanpa campur tangan pihak ketiga.

2. Imutabilitas

   Data pada *blockchain* tidak bisa diubah karena setiap blok terhubung dengan blok sebelumnya. Dengan demikian, perubahan kecil pada satu blok dapat membuat blok lain menjadi tidak valid.

3. Keterlacakan

   Pengguna dapat memverifikasi dan melihat riwayat data yang tersimpan di *blockchain*.

4. Non-repudiasi

   Transaksi yang sudah ditandatangani secara digital oleh seorang pengguna dan disimpan di *blockchain* tidak dapat disangkal. Ini karena tanda tangan digital memerlukan kunci privat yang hanya dimiliki oleh pengguna tersebut.

5. Transparansi

   Setiap pengguna dapat mengakses dan berinteraksi dengan sistem *blockchain* sesuai dengan hak akses yang mereka miliki.

6. Pseudonimitas dan anonimitas

   Blockchain memiliki kemampuan untuk mempertahankan anonimitas pengguna pada tingkat tertentu.

2.1.4.3 Struktur *Blockchain*

Noor (2020) menjelaskan komponen *blockchain* yang terdiri dari tiga komponen utama yaitu blok, rantai, dan jaringan.

1. Blok (*Block*)

   Blok adalah kumpulan transaksi yang tercatat dalam periode tertentu, menyimpan data secara permanen dan tidak bisa diubah.

2. Rantai (*Chain*)

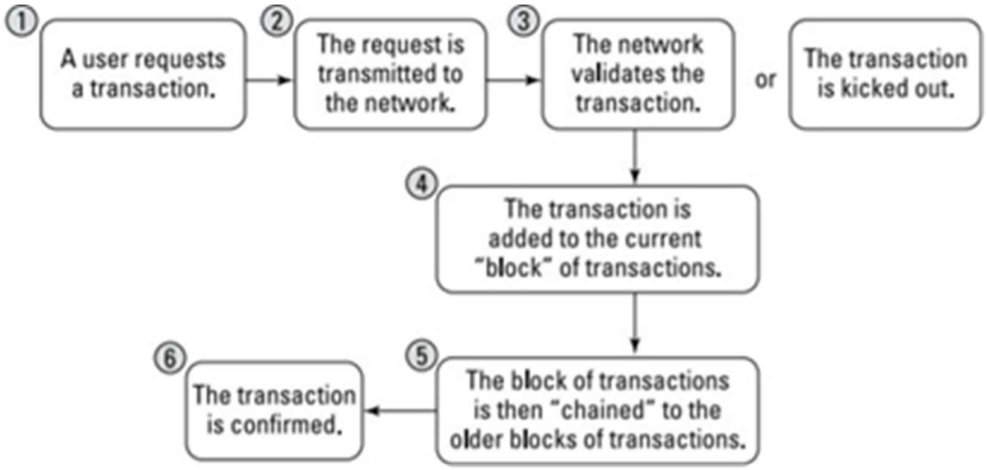
   Rantai menghubungkan blok menggunakan *hash* untuk menjaga integritas dan keaslian data.

3. Jaringan (*Network*)

   Jaringan terdiri dari *node* yang menyimpan salinan lengkap data dan menyetujui perubahan bersama.

2.1.4.4 Cara Kerja *Blockchain*

Cara kerja *blockchain* mirip dengan *spreadsheet* yang berisi transaksi-transaksi dan diduplikasi ratusan kali dalam jaringan komputer. Jaringan ini secara rutin memperbarui *spreadsheet* tersebut. Dari analogi ini, dapat disimpulkan bahwa *blockchain* tidak memerlukan perantara, setiap orang dapat menulis transaksi pada *spreadsheet* mereka sendiri. Karena itu, *blockchain* menggunakan algoritma khusus untuk mencapai konsensus dalam jaringan pada setiap entri yang ditambahkan. Berikut ini adalah gambaran bagaimana *blockchain* bekerja menurut Laurence (2017) dalam Putri (2021).



Gambar 2.2 Cara Kerja Blockchain

(Sumber: Laurence, 2017 dalam Putri, 2021)

Cara kerja *blockchain* dapat dijelaskan dengan mengikuti langkah-langkah berikut, seperti yang ditunjukkan pada gambar:

1. Pengguna Mengajukan Transaksi

   Seorang pengguna mengajukan permintaan transaksi, misalnya untuk menambah atau memperbarui arsip digital.

2. Permintaan Ditransmisikan ke Jaringan

   Permintaan transaksi dikirim ke jaringan *blockchain* yang terdiri dari banyak *node* pengelola arsip digital secara terdistribusi.

3. Validasi Transaksi oleh Jaringan

   Jaringan memvalidasi transaksi untuk memastikan keabsahan dan integritas data, mencegah manipulasi.

4. Transaksi Ditambahkan ke Blok Saat Ini

   Setelah validasi, transaksi dicatat dalam blok transaksi yang sedang berjalan.

5. Blok Transaksi Dirangkai dengan Blok Lama

   Blok transaksi baru dihubungkan dengan blok sebelumnya, membentuk rantai yang menjaga integritas dan urutan arsip.

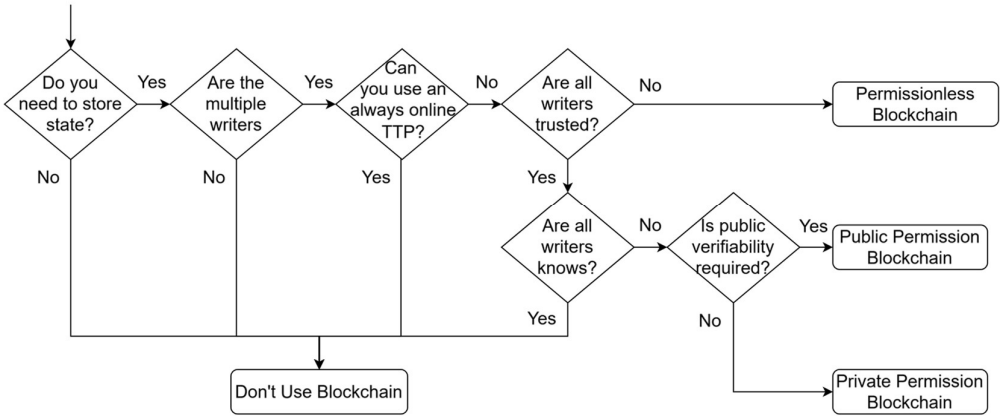
6. Transaksi Dikonfirmasi

   Setelah blok ditambahkan ke rantai, transaksi dikonfirmasi dan tercatat permanen, menjamin keamanan dan keandalan arsip.

Dengan mengikuti proses ini, *blockchain* memastikan bahwa setiap transaksi atau perubahan yang dilakukan pada arsip digital dapat dilacak, diverifikasi, dan diamankan, menjadikannya solusi yang sangat cocok untuk manajemen kearsipan elektronik.

2.1.4.5 Analisis Kebutuhan Penggunaan Teknologi *Blockchain*

Analisis ini sangatlah penting agar penggunaan teknologi *blockchain* bukan hanya digunakan karena mengikuti tren teknologi yang sedang populer dan canggih saja, melainkan benar-benar digunakan untuk kebutuhan sistem dan dapat mengatasi permasalahan yang ada pada sistem sebelumnya sistem. Analisis kebutuhan dapat dilakukan dengan mengikuti flowchart pada Gambar 2.3.



Gambar 2.3 Flowchart Analisis Kebutuhan Penggunaan Blockchain

(Sumber: Wust & Gervais, 2028 dalam Putri, 2021)

Berikut ini hal-hal yang perlu diperhatikan dalam tahap analisis akan kebutuhan penggunaan *blockchain*:

1. Hal pertama yang harus diperhatikan yaitu apakah perlu adanya data untuk disimpan? Jika tidak ada, maka sistem tidak membutuhkan *blockchain*.
2. Kedua, siapa saja yang dapat menginput data untuk disimpan? Jika hanya satu orang saja, maka sistem tidak membutuhkan *blockchain*.
3. Ketiga, apakah terdapat pihak ketiga dalam sistem seperti admin yang akan selalu aktif dan bisa dipercaya? Jika ya, maka sistem tidak membutuhkan *blockchain*.

4. Keempat, apakah semua pengguna sistem yang dapat menyimpan atau menginput data dapat diketahui? Jika tidak diketahui, maka bisa menggunakan *permissionless blockchain*.
5. Kelima, Jika semua pengguna sistem yang dapat menyimpan atau menginput data dapat diketahui, apakah semuanya dapat dipercaya? Jika iya, maka sistem tidak membutuhkan *blockchain*.
6. Jika tidak semua pengguna sistem yang dapat menyimpan atau menginput data dapat dipercaya, hal yang harus dipertimbangkan adalah apakah verifikasi publik dibutuhkan? Jika iya, maka kita bisa menggunakan *public permission blockchain*. Dan jika tidak, maka kita bisa menggunakan *private permission blockchain*.

2.2 Teori Khusus

2.2.1 Kriptografi

Menurut Cristy dan Riandari (2021) dalam Hidayatulloh dkk. (2023) kriptografi berasal dari 2 kata dalam bahasa Yunani, yaitu kata *crypto* memiliki arti rahasia dan kata *graphia* yang berarti tulisan. Secara definisi kriptografi adalah ilmu yang mempelajari seni untuk mengamankan isi pesan atau data dan informasi dengan tujuan supaya isi dari pesan tersebut hanya dapat diketahui oleh pengirim dan penerima (Sitepu dkk., 2022 dalam Hidayatulloh dkk., 2023). Tujuan utama kriptografi adalah menjaga kerahasiaan, integritas, dan otentikasi data. Prinsip Dasar dari kriptografi yaitu:

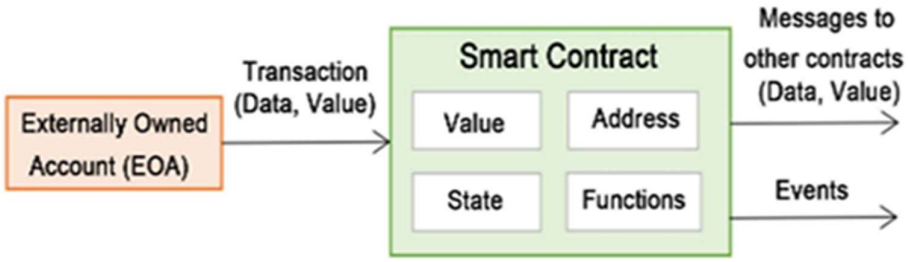
1. Enkripsi merupakan transformasi pesan asli (*plaintext*) menjadi bentuk tersamar (*ciphertext*) menggunakan algoritma kriptografis dan kunci.

2. Dekripsi merupakan proses mengembalikan *ciphertext* ke *plaintext* menggunakan kunci yang sesuai.

2.2.2 *Smart Contract*

2.2.2.1 Cara Kerja *Smart Contract*

Mark Giancaspro (2017) dalam Mansula (2023) menjelaskan *smart contract* sebagai program komputer yang memverifikasi dan menjalankan ketentuannya setelah terjadinya peristiwa yang telah ditentukan sebelumnya. Setelah dikodekan dan dimasukkan ke dalam *blockchain*, kontrak tidak dapat diubah dan beroperasi sesuai dengan instruksi yang diprogram.



Gambar 2.4 Struktur Smart Contract

(Sumber: Alsulbi dkk., 2021)

Alsulbi dkk. (2021) mendeskripsikan bahwa *smart contract*s terdiri dari nilai, alamat, status dan fungsi seperti yang ditunjukkan pada Gambar 2.4. Gambar 2.4 menunjukkan struktur dasar sebuah *smart contract*. Dimulai dari *Externally Owned Account* (EOA) yang mengirimkan transaksi berisi data dan nilai ke *smart contract* yang terdiri dari nilai, alamat, status dan fungsi. *Smart contract* memproses transaksi sesuai instruksi yang diprogramkan, menghasilkan pesan ke kontrak lain atau memicu peristiwa berisi data dan nilai. *Smart contract* ini beroperasi otomatis dan tidak bisa diubah setelah masuk ke dalam *blockchain*.

Menurut Mohanta dkk. (2018) dalam Fernando (2022) menerangkan bahwa sebuah *smart contract* memiliki karakteristik berikut:

1. *Smart contract* merupakan serangkaian kode yang dapat dibaca oleh mesin yang berjalan pada platform *blockchain*.
2. *Smart contract* adalah bagian dari satu program aplikasi.
3. *Smart contract* adalah program yang digerakkan oleh peristiwa.
4. *Smart contract* bersifat otonom setelah dibuat, tidak perlu dipantau.
5. *Smart contract* dapat didistribusikan.

2.2.2.2 Peran Spesifik *Smart Contract* dalam Penelitian

Dalam konteks penelitian ini, *smart contract* tidak hanya bertindak sebagai kontrak digital pasif, melainkan diimplementasikan sebagai tulang punggung logika bisnis (*business logic*) yang terdesentralisasi dan otonom. Perannya dirancang secara spesifik untuk menangani dan mengotomatiskan fungsional utama dalam sistem manajemen arsip yang dikembangkan.

Smart contract berfungsi sebagai gerbang otorisasi sistem. Ia mengelola proses pendaftaran pengguna baru dan menetapkan peran mereka secara *on-chain* (*user, archivist,* atau *signer*). Setiap peran memiliki hak akses yang telah didefinisikan secara programatik di dalam kode contract, sehingga memastikan bahwa hanya pengguna dengan otorisasi yang tepat yang dapat melakukan tindakan tertentu.

Setiap kali sebuah dokumen diunggah ke sistem, *smart contract* secara otomatis menerima, memvalidasi, dan mencatat metadata dokumen beserta *hash* kriptografisnya ke dalam *blockchain*. Proses ini menciptakan jejak audit yang

immutable dan dapat diverifikasi secara publik, yang secara fundamental menjamin integritas dan keaslian setiap arsip dari upaya manipulasi.

Smart contract menerapkan logika kontrol akses yang kompleks, baik untuk dokumen privat maupun publik. Untuk dokumen privat, ia mengelola daftar alamat (*whitelist*) yang diizinkan dan secara otomatis memvalidasi setiap permintaan akses terhadap daftar tersebut. Karena aturan ini dieksekusi oleh jaringan terdistribusi, bukan oleh server terpusat, maka risiko adanya satu titik kegagalan (*single point of failure*) dapat dieliminasi.

Seluruh proses yang terkait permintaan dan validasi tanda tangan digital sepenuhnya diotomatiskan oleh *smart contract*. Ia mengelola alur pengajuan permintaan dari pengarsip ke penandatangan, mencatat setiap perubahan status (*Pending, Approved, Rejected*), dan merekam tanda tangan digital yang valid sebagai bukti transaksi yang sah dan tidak dapat disangkal di *blockchain*.

Dengan demikian, implementasi *smart contract* dalam penelitian ini secara efektif mengubah aturan-aturan kearsipan dari sekadar prosedur operasional manual menjadi kode program yang transparan, dapat diaudit, dan dieksekusi secara otonom. Hal ini secara fundamental meningkatkan integritas, keamanan, dan akuntabilitas dalam pengelolaan arsip digital.

2.2.3 DApps (*Decentralized Application*)

DApps merupakan aplikasi terdesentralisasi yang berfungsi seperti aplikasi web tradisional, tetapi menggunakan *wallet* untuk berinteraksi dengan *blockchain* melalui *smart contract* (Cagigas dkk., 2021. Wu (2019) dalam Fernando (2022) menjelaskan bahwa DApps dapat memiliki klien, aplikasi seluler, dan server seperti

aplikasi tradisional, tetapi data dan operasinya disimpan dalam *smart contract* di jaringan *blockchain*. Menurut Cai dkk. (2019) dalam Fernando (2022) DApps dapat dikenali melalui empat karakteristik berikut:

1. DApps harus bersifat *open source* agar dapat diaudit oleh pihak ketiga.
2. DApps memerlukan sistem mata uang internal atau token untuk mendukung transaksi dan perhitungan kredit dalam ekosistemnya.
3. Konsensus dalam DApps dijalankan secara terdesentralisasi untuk memastikan transparansi.
4. Dengan sistem terdesentralisasi, DApps tidak memiliki titik pusat kegagalan karena seluruh komponen dijalankan di *blockchain*.

2.2.4 IPFS (*Inter Planetary File System*)

Trautwein dkk. (2022) menjelaskan bahwa *Inter Planetary File System* (IPFS) adalah sebuah sistem penyimpanan dan distribusi data berbasis *peer-to-peer* yang dirancang untuk mendukung web terdesentralisasi. IPFS memungkinkan data disimpan dan diakses secara terdistribusi tanpa bergantung pada penyedia layanan cloud besar, sehingga meningkatkan keandalan dan desentralisasi web. Berikut prinsip-prinsip IPFS:

1. IPFS memakai *content identifier* (CID), *hash* unik yang memisahkan konten dari lokasi penyimpanannya, sehingga data bisa diakses dari *peer* mana saja secara desentralisasi.
2. IPFS mengindeks objek secara terdesentralisasi menggunakan jaringan *peer-to-peer* dan *Distributed Hash Table* (DHT), memungkinkan data ditemukan dari berbagai lokasi dan meningkatkan ketahanan sistem.

3. Data di IPFS bersifat *immutable* dan dapat diverifikasi sendiri melalui *hash*ing, setiap perubahan menghasilkan CID baru untuk menjaga integritas tanpa otoritas pusat.

2.3 Teori Perancangan

2.4.1 Ethereum

Ethereum adalah platform *blockchain* yang dirancang untuk memungkinkan pengembangan aplikasi terdesentralisasi (DApps) dan menjalankan *smart contract*. Ethereum membuka peluang bagi pengembang untuk menciptakan berbagai aplikasi yang dapat beroperasi tanpa perantara (Raharjo, 2022). *Blockchain* ethereum berfungsi sebagai buku besar publik yang terdesentralisasi, dimana setiap transaksi dan kontrak dapat diverifikasi oleh semua pengguna jaringan. Hal ini menciptakan ekosistem yang lebih adil dan terbuka, dimana siapapun dapat berpartisipasi tanpa harus bergantung pada pihak ketiga (Raharjo, 2022).

Dengan kemampuan untuk menciptakan Organisasi Desentralisasi Otonomi, ethereum juga memungkinkan pembentukan entitas yang dapat beroperasi tanpa manajemen tradisional, dengan keputusan yang diambil secara kolektif oleh anggotanya. Ethereum dapat membuka jalan bagi inovasi dan model bisnis baru yang lebih inklusif dan efisien (Faqir-Rhazoui dkk., 2021).

2.4.2 *Wallet* MetaMask

MetaMask merupakan *wallet* kripto berbasis ethereum yang berperan penting dalam memfasilitasi akses ke dalam jaringan *blockchain*. Fungsi MetaMask meliputi pembuatan akun di berbagai jaringan ethereum, pengelolaan kunci pribadi, transfer dana antar akun, dan penyimpanan token. Selain itu, MetaMask

memungkinkan pengguna untuk bertransaksi di beragam jaringan ethereum, meninjau detail transaksi melalui Etherscan, serta melakukan pertukaran token pada *Decentralized Exchanges* (DEX). Selain itu MetaMask juga menyuntikkan pustaka JavaScript web3JS, yang memungkinkan interaksi dengan *blockchain* ethereum dan mendukung fungsi pemrograman *smart contract* (Lee, 2023).

Gas adalah satuan harga internal untuk menjalankan transaksi atau kontrak dalam ethereum. Alih-alih menggunakan Ether sebagai biaya transaksi, ethereum sengaja menggunakan konsep gas untuk menilai transaksi, karena gas lebih tepat menggambarkan kompleksitas perhitungan yang terlibat, sementara harga Ether berfluktuasi karena pengaruh pasar. Harga gas ditentukan oleh permintaan pasar terhadap transaksi, di mana penambang atau validator cenderung memprioritaskan transaksi berdasarkan harga gas yang dibayarkan, semakin tinggi harga gas yang dibayar, semakin cepat transaksi tersebut dikonfirmasi (Lee, 2023).

2.4.3 Solidity

Solidity adalah bahasa pemrograman yang berjenis *statically-typed* dan menggunakan tanda kurung kurawal, dirancang khusus untuk mengembangkan *smart contract* yang berjalan di *ethereum virtual machine* (EVM). Bahasa ini memungkinkan pengembang untuk mendefinisikan aturan dan perilaku aplikasi terdesentralisasi DApps (*decentralized applications*) melalui *smart contract* yang dieksekusi di dalam jaringan *peer-to-peer*. Dalam eksekusinya, tidak ada pihak yang memiliki otoritas khusus, sehingga *smart contract* dapat diterapkan untuk berbagai logika seperti tokenisasi nilai, kepemilikan, pemungutan suara, dan lainnya (Solidity, 2024).

2.4.4 JavaScript

Saragih (2016) dalam Fernando (2022) menjelaskan bahwa JavaScript adalah pengembangan dari bahasa C++ dengan sintaks yang lebih mudah dipahami. Interpreter untuk bahasa ini sudah tersedia di ASP dan Internet Explorer. Salah satu keunggulan JavaScript adalah kemampuannya untuk berinteraksi dengan HTML, yang memudahkan pengembang dalam menambahkan konten dinamis, mengganti warna latar belakang, mengubah banner, menciptakan efek mouse, menu interaktif, dan lainnya.

2.4.5 NodeJS

NodeJS adalah platform perangkat lunak untuk aplikasi jaringan dan sisi server. NodeJS ditulis dalam bahasa JavaScript dan dapat dijalankan di sistem operasi Windows, Mac OS, dan Linux tanpa perlu memodifikasi kode program (Supardi, 2020 dalam Al Fakhirin, 2022).

NodeJS menggunakan mesin V8 yang dikembangkan oleh Google, dengan mesin V8 dari Google, NodeJS menawarkan performa tinggi melalui model *event-driven,* yang cocok untuk aplikasi web *real-time* seperti chatting dan game online. Selain itu, npm (*Node Package Manager*) menyediakan banyak modul untuk mempermudah pengembangan aplikasi.

2.4.6 ReactJS

ReactJS adalah sebuah *library* JavaScript yang dikembangkan oleh Facebook. React bukan merupakan framework MVC, melainkan pustaka yang bersifat komposabel untuk antarmuka pengguna, yang memungkinkan kita untuk

membuat berbagai antarmuka yang dapat dibagi menjadi beberapa komponen (Musa, 2018 dalam Rohmawati & Zakaria, 2023).

ReactJS sering digunakan bersama NodeJS, yang berfungsi sebagai lingkungan server untuk membangun aplikasi web yang dinamis dan *real-time*. Kombinasi keduanya memungkinkan pengembang untuk menciptakan aplikasi web full-stack yang efisien, dengan React menangani tampilan antarmuka pengguna di sisi klien dan NodeJS mengelola logika serta data di sisi server.

2.4.7 API Web3JS

API Web3JS merupakan API yang berfungsi untuk menghubungkan front-end dengan *blockchain* sehingga dapat berinteraksi dengan *smart contract*. Putri (2021) menjelaskan Web3JS adalah kumpulan *library* yang memungkinkan kita untuk dapat berinteraksi dengan *node* ethereum lokal maupun remote, menggunakan HTTP, WebSocket, atau IPC. API Web3JS memuat modul berikut:

1. Web3-eth berfungsi untuk *blockchain* ethereum dan *smart contract*.
2. Web3-shh berfungsi untuk *whisper* protokol untuk berkomunikasi p2p dan broadcast.
3. Web3-bzz berfungsi untuk *swarm* protokol, penyimpanan file terdesentralisasi.
4. Web3-utils berfungsi untuk pembantu yang berguna untuk pengembangan DApp.

2.4.8 PostgreSQL

PostgreSQL adalah salah satu sistem basis data besar yang menawarkan kemampuan skalabilitas, fleksibilitas, dan performa tinggi. Basis data ini banyak digunakan di berbagai platform dan kompatibel dengan berbagai bahasa pemrograman.

PostgreSQL merupakan sistem manajemen basis data objek-relasional (ORDBMS) yang dikembangkan di Departemen Ilmu Komputer Universitas California, Berkeley. Sebagai pelopor untuk banyak perangkat lunak DBMS komersial, PostgreSQL dilisensikan di bawah GPL (*General Public License*), sehingga penggunaannya, modifikasi, dan distribusi dapat dilakukan secara gratis untuk keperluan pribadi maupun komersial (Wahyuningsih & Junianto, 2020).

2.4.9 Hardhat

Jain (2023) menjelaskan bahwa hardhat adalah framework pengembangan untuk ethereum yang digunakan untuk membangun, menguji, dan menyebarkan *smart contract* pada jaringan *blockchain* ethereum dan kompatibel dengan ethereum lainnya. Hardhat mempermudah pengembang dalam mengelola proses pengembangan *smart contract* dengan menyediakan berbagai alat dan fitur.

Beberapa fitur utama Hardhat antara lain:

1. Mendukung kompilasi otomatis *smart contract* yang ditulis menggunakan Solidity dan Vyper, serta mendukung berbagai versi Solidity.
2. Menyediakan jaringan lokal yang memungkinkan pengembang untuk menguji kontrak mereka secara langsung di dalam lingkungan lokal sebelum dilakukan *deploy* ke jaringan ethereum utama atau testnet.

3. Menyediakan integrasi dengan framework pengujian seperti Mocha dan Chai untuk melakukan unit *testing* pada *smart contract* secara efisien.
4. Memiliki fitur debugging yang kuat, yang memungkinkan pengembang untuk melakukan pemecahan masalah *smart contract*, dengan tampilan yang jelas dari log transaksi dan *stack trace*.
5. Mendukung berbagai plugin eksternal yang memperluas fungsionalitasnya, seperti plugin untuk *deploy*, verifikasi kontrak di Etherscan, atau integrasi dengan layanan lainnya.
6. Integrasi dengan EtherJS dan Web3JS. Hardhat bekerja dengan pustaka JavaScript populer seperti EtherJS dan Web3JS untuk memfasilitasi interaksi dengan *smart contract* dari aplikasi front-end.

2.4.10 Metode *Rapid Application Development* (RAD)

Putri (2021) menjelaskan bahwa metode pengembangan *Rapid Application Development* (RAD) adalah metode yang mengadaptasi metode waterfall yang dilakukan dalam waktu singkat. Di dalam metode RAD terdapat 3 (tiga) tahapan yaitu:

Gambar 2.5 Tahapan Metode Rapid Application Development (RAD)

(Sumber: Putri, 2021)

1. Pada tahap rencana kebutuhan, kebutuhan pengguna dan pemangku kepentingan dikumpulkan dan dianalisis. Tim pengembang bekerja sama untuk menentukan spesifikasi sistem. Hasilnya berupa dokumen kebutuhan yang jelas untuk desain berikutnya.
2. Pada tahap proses desain, desain sistem dan antarmuka pengguna dikembangkan berdasarkan dokumen kebutuhan. Prototipe dibuat untuk mendapatkan feedback dari pengguna.
3. Pada tahap implementasi, kode sumber dikembangkan dan diuji untuk memastikan fungsionalitasnya. Pengguna dilibatkan dalam pengujian untuk memverifikasi sistem. Mencakup dukungan pasca-implementasi untuk mengatasi masalah yang mungkin muncul.

Pemilihan metodologi pengembangan harus disesuaikan dengan sifat dan tujuan proyek. Untuk penelitian yang bersifat investigatif dan melibatkan teknologi baru seperti *blockchain*, di mana terdapat ketidakpastian teknis dan potensi perubahan kebutuhan, diperlukan pendekatan yang fleksibel dan adaptif.

Pengembangan sistem yang melibatkan paradigma interaksi yang sepenuhnya baru, seperti pada arsitektur berbasis *blockchain*, menghadapi tantangan di mana spesifikasi kebutuhan tradisional seringkali tidak memadai. Konsep-konsep seperti penggunaan dompet digital (*digital wallet*) sebagai identitas, keharusan menyetujui setiap transaksi yang mengubah *state*, dan pemahaman terhadap data yang bersifat *immutable* (tidak dapat diubah), merupakan alur kerja yang asing bagi sebagian besar pengguna. Menjelaskan mekanisme ini hanya melalui dokumen teks atau diagram statis sangatlah sulit. Oleh karena itu,

prototipe yang dihasilkan melalui siklus RAD tidak hanya berfungsi sebagai model kerja, tetapi juga sebagai instrumen validasi konsep. Prototipe memungkinkan para pemangku kepentingan untuk secara langsung mengalami dan berinteraksi dengan alur kerja yang baru, sehingga mampu memberikan umpan balik yang jauh lebih akurat dan kontekstual. Pendekatan ini terbukti efektif untuk menjembatani kesenjangan pemahaman ketika memperkenalkan teknologi inovatif kepada pengguna (Chrismanto dkk., 2019).

Selain itu, tantangan rekayasa perangkat lunak *blockchain* yang paling signifikan adalah sifat *immutable* dari *smart contract* setelah diterapkan (*deployed*) di jaringan utama. Kesalahan logika atau celah keamanan dalam kode *smart contract* tidak dapat diperbaiki dengan mudah seperti pada aplikasi terpusat; kesalahan tersebut bersifat permanen dan dapat berakibat fatal. Risiko ini menjadikan metodologi tangkas (*agile*) seperti RAD lebih sesuai dibandingkan pendekatan tradisional, terutama untuk proyek dengan ketidakpastian teknologi dan kebutuhan yang sering berubah seperti pada pengembangan *blockchain* (Bahar dkk., 2025).

Metodologi RAD, dengan penekanannya pada *workshop design* dan siklus pembangunan prototipe berulang, secara langsung memitigasi risiko ini. Logika inti dari *smart contract* dapat dirancang, diimplementasikan dalam lingkungan pengujian yang terkontrol (seperti *Hardhat*), dan diuji secara iteratif bersama pengguna. Setiap siklus memungkinkan pengembang untuk memvalidasi bahwa aturan bisnis yang tertanam dalam kontrak, seperti hak akses (RBAC), alur persetujuan, dan logika pembagian dokumen, berfungsi sesuai harapan sebelum

dikunci secara permanen di *blockchain*. Proses ini secara drastis mengurangi risiko kegagalan sistemik pada tahap implementasi akhir.

Dengan demikian, pemilihan RAD bukanlah didasarkan pada kecepatan pengembangan semata, melainkan pada kemampuannya untuk secara sistematis mengatasi risiko-risiko unik yang muncul dari persinggungan antara teknologi *blockchain* yang kompleks dan kebutuhan pengguna akhir.

2.4 Teori Pengujian

2.4.1 Pengujian Black Box

Pressman dan Maxim (2019) menjelaskan black box *testing* adalah metode pengujian perangkat lunak yang berfokus pada aspek fungsional dari sistem tanpa mempertimbangkan struktur internal dari kode program. Dalam *Software Engineering: A Practitioner's Approach*, dijelaskan bahwa pendekatan ini dilakukan dengan memberikan input pada perangkat lunak dan mengamati output yang dihasilkan, untuk memastikan apakah perangkat lunak berperilaku sesuai dengan spesifikasi yang telah ditentukan.

Pengujian ini bertujuan untuk mengevaluasi fungsi perangkat lunak dari perspektif pengguna, sehingga penguji tidak perlu mengetahui bagaimana sistem bekerja secara internal. Black box *testing* digunakan untuk menemukan kesalahan dalam fungsi, kesalahan antarmuka, kesalahan dalam struktur data eksternal, serta kesalahan dalam performa atau inisialisasi sistem. Dengan kata lain, pendekatan ini menekankan pada apa yang dilakukan oleh perangkat lunak, bukan bagaimana cara kerjanya di dalam.

2.4.2 Pengujian White Box

Pressman dan Maxim (2019) menjelaskan white box *testing* adalah metode pengujian perangkat lunak yang mengevaluasi struktur internal, logika, dan alur kontrol dari program. Dalam *Software Engineering: A Practitioner's Approach*, dijelaskan bahwa pengujian ini dilakukan dengan memeriksa kode sumber secara langsung untuk memastikan bahwa setiap jalur logika dalam program telah diuji secara menyeluruh. Penguji perlu memahami bagaimana kode bekerja, termasuk struktur percabangan, perulangan, serta interaksi antar fungsi atau modul.

Tujuan utama dari white box *testing* adalah untuk menjamin bahwa semua bagian dalam program bekerja sesuai dengan yang diharapkan, termasuk memastikan bahwa tidak ada jalur program yang tidak teruji atau tidak dapat dijalankan. Metode ini sangat efektif dalam menemukan kesalahan tersembunyi dalam logika program yang mungkin tidak terdeteksi melalui pengujian fungsional.

2.4.3 Pengujian Komparatif

Dalam buku "*Design and Analysis of Experiments*" karya Douglas C. Montgomery menjelaskan bahwa eksperimen komparatif dilakukan untuk mengetahui apakah ada perbedaan nyata antara dua perlakuan atau lebih terhadap suatu variabel tertentu.

Montgomery (2019) menekankan pentingnya pengacakan dalam penempatan perlakuan pada unit eksperimen agar hasil yang diperoleh tidak dipengaruhi oleh faktor luar yang tidak dikendalikan. Selain itu, penggunaan replikasi diperlukan agar data yang dihasilkan lebih dapat diandalkan dan tidak hanya bergantung pada satu pengamatan saja.

Fokus penelitian pada aspek Integritas dan Transparansi data bukanlah sebuah pilihan arbitrer, melainkan hasil dari sintesis antara kebutuhan mendesak dari domain masalah dengan kapabilitas inti dari teknologi yang diusulkan. Dari sisi domain kearsipan, fungsi sebuah arsip sebagai alat bukti yang sah menuntut adanya jaminan keaslian dan keutuhan.

Hal ini secara eksplisit diamanatkan oleh Undang-Undang Republik Indonesia Nomor 43 Tahun 2009 tentang Kearsipan dan diperkuat oleh standar internasional dari *International Council on Archives* (ICA) yang mendefinisikan arsip sebagai rekaman yang harus terjaga otentisitas, reliabilitas, dan integritasnya. Di sisi lain, dari perspektif teknologi, *National Institute of Standards and Technology* (NIST) melalui laporannya NISTIR 8202 menegaskan bahwa karakteristik inheren *blockchain* adalah sifatnya yang *tamperevident* dan *immutable*, yang merupakan fondasi teknis untuk menjamin integritas dan transparansi (Yaga dkk., 2019). Dengan demikian, kedua aspek ini menjadi jembatan konseptual antara masalah yang ada dengan solusi yang ditawarkan.

Dalam konteks penelitian ini, pengujian komparatif diadaptasi untuk mengevaluasi dua arsitektur sistem yang berbeda, yaitu:

1. Sistem berbasis *Blockchain* (arsitektur terdesentralisasi).
2. Sistem berbasis arsitektur terpusat (arsitektur terpusat menggunakan API dan database).

Untuk memfokuskan evaluasi, kerangka pengujian komparatif dalam penelitian ini distrukturkan ke dalam dua pilar utama yang diturunkan langsung dari judul dan tujuan penelitian:

2.4.3.1 Pengujian Integritas Data

Pilar ini berfokus pada kemampuan sistem untuk memastikan keutuhan, keaslian, dan perlindungan data dari modifikasi yang tidak sah. Pengujian integritas dalam konteks ini sejalan dengan konsep yang dibahas oleh Stančić dan Bralić (2021), yang menyoroti pentingnya imutabilitas data untuk mengatasi keterbatasan pada arsip digital tradisional. Metrik-metrik yang diuji di bawah pilar ini akan mensimulasikan berbagai skenario serangan atau pelanggaran aturan, seperti upaya manipulasi data dan penegakan kontrol akses berbasis peran (RBAC).

2.4.3.2 Pengujian Transparansi

Pilar ini mengevaluasi kemampuan sistem untuk menyediakan *audit trail* yang lengkap, dapat diandalkan, dan tidak dapat disangkal. Hal ini penting untuk akuntabilitas dan penelusuran riwayat arsip. Seperti yang dijelaskan oleh Thazhath dkk. (2022), log audit yang *immutable* dan terjaga privasinya adalah komponen kunci dalam sistem yang aman. Oleh karena itu, metrik ini akan mengukur imutabilitas dari riwayat revisi dan finalitas dari proses persetujuan digital.

Selain kedua pilar utama tersebut, analisis komparatif yang komprehensif juga perlu mempertimbangkan biaya kinerja (*performance cost*) dari setiap arsitektur. Pengukuran kinerja, seperti yang dijelaskan oleh Hyperledger (2018), mencakup metrik latensi dan *throughput*. Dalam penelitian ini, analisis kinerja tidak digunakan sebagai penentu utama "optimalisasi", melainkan sebagai cara untuk mengkuantifikasi *trade-off* yang terjadi. Analisis ini memberikan konteks seimbang mengenai "biaya" komputasi yang harus dibayar untuk mendapatkan jaminan integritas dan transparansi yang lebih tinggi dari teknologi *blockchain*.

2.5 Penelitian Terkait dan Kebaruan Penelitian

Berikut ini adalah penelitian-penelitian terkait yang dijadikan referensi dan penjelasan mengenai aspek kebaruan dari penelitian yang akan dilakukan.

Tabel 2.1 Penelitian Terkait dan Kebaruan Penelitian.

No	Peneliti/Tahun	Judul	Teknologi yang Digunakan	Keterkaitan	Kesenjangan
1	Anggraini & Nurbaiti (2023)	Strategi Implementasi Pengarsipan Digital Dokumen Kontrak Berbasis Web dalam Manajemen Operasional (Studi Kasus di PT Industri Nabati Lestari)	Sistem Manajemen Arsip Elektronik	Penerapan digitalisasi di bidang kearsipan.	Tidak mengimplementasikan teknologi *blockchain* pada manajemen arsip
2	Aprilia dkk. (2020)	Efektivitas Pengelolaan Arsip Elektronik di Indonesia	Sistem Manajemen Arsip Elektronik	Penerapan digitalisasi di bidang kearsipan.	Tidak mengimplementasikan teknologi *blockchain* pada manajemen arsip
3	Chairina & Candrasa (2022)	Peran Manajemen Arsip dalam Pengamanan Data Base	Sistem Manajemen Arsip Elektronik	Penerapan digitalisasi di	Tidak mengimplementasikan

				bidang kearsipan.	teknologi *blockchain* pada manajemen arsip
4	Amalia (2022)	Efektivitas Digitalisasi Arsip Surat Melalui Pembuatan Aplikasi Document Management System (DMS) Pada Subbagian Tata Usaha Kantor Kementerian Agama Kota Lhokseumawe	Sistem Manajemen Arsip Elektronik	Penerapan digitalisasi di bidang kearsipan.	Tidak mengimplementasikan teknologi *blockchain* pada manajemen arsip
5	Putri (2021)	Perancangan Dan Implementasi Teknologi *Blockchain* Pada Transaksi Wakaf Produktif	Sistem *Blockchain*	Penerapan *blockchain*	Tidak menunjukan nilai optimalisasi implementasi teknologi *blockchain* pada manajemen arsip
6	Saputra dkk. (2023)	*AniraBlock: A Leap Towards Dynamic Smart contracts in Agriculture Using Blockchain Based Key-Value Format Framework*	Sistem Manajemen pertanian berbasis *Smart contract*s pada *Blockchain*	Penggunaan *blockchain* di bidang kearsipan.	Tidak menunjukan nilai optimalisasi implementasi teknologi *blockchain* pada manajemen arsip
7	Noor (2020)	Implementasi *Blockchain* Di Dunia Kearsipan: Peluang, Tantangan, Solusi, Atau Masalah Baru?	Sistem *Blockchain*	Penggunaan *blockchain* di	Tidak menunjukan nilai optimalisasi implementasi

				bidang kearsipan.	teknologi *blockchain* pada manajemen arsip
8	Alsulbi dkk. (2021)	*A Proposed Framework for Secure Data Storage in a Big Data Environment Based on Blockchain and Mobile Agent*	Sistem Manajemen Dokumen berbasis *Blockchain*	Penggunaan *blockchain* di bidang kearsipan.	Tidak menunjukan nilai optimalisasi implementasi teknologi *blockchain* pada manajemen arsip
9	Permatasari dkk. (2020)	*Blockchain Implementation to Verify Archives Integrity on Cilegon E-Archive*	Sistem Manajemen Dokumen berbasis *Blockchain*	Penggunaan *blockchain* di bidang kearsipan.	Tidak menunjukan nilai optimalisasi implementasi teknologi *blockchain* pada manajemen arsip
10	Cagigas dkk. (2021)	*Blockchain for Public Services: A Systematic Literature Review*	Sistem Manajemen Dokumen berbasis *Blockchain*	Penggunaan *blockchain* di bidang kearsipan.	Tidak menunjukan nilai optimalisasi implementasi teknologi *blockchain* pada manajemen arsip

Tabel 2.1 merinci penelitian-penelitian terkait dan menunjukkan kebaruan teknologi dalam manajemen arsip. Berdasarkan informasi tersebut, peneliti bermaksud mengimplementasikan teknologi *blockchain* dengan *smart contract* dalam manajemen arsip di Indonesia. Penelitian ini bertujuan untuk mengatasi dampak yang ditimbulkan dari penggunaan manajemen arsip digital yang semakin komplek sehingga diharapkan mampu memberikan optimalisasi integrasi dan transparansi pada manajemen arsip digital.