

BAB II

TINJAUAN PUSTAKA

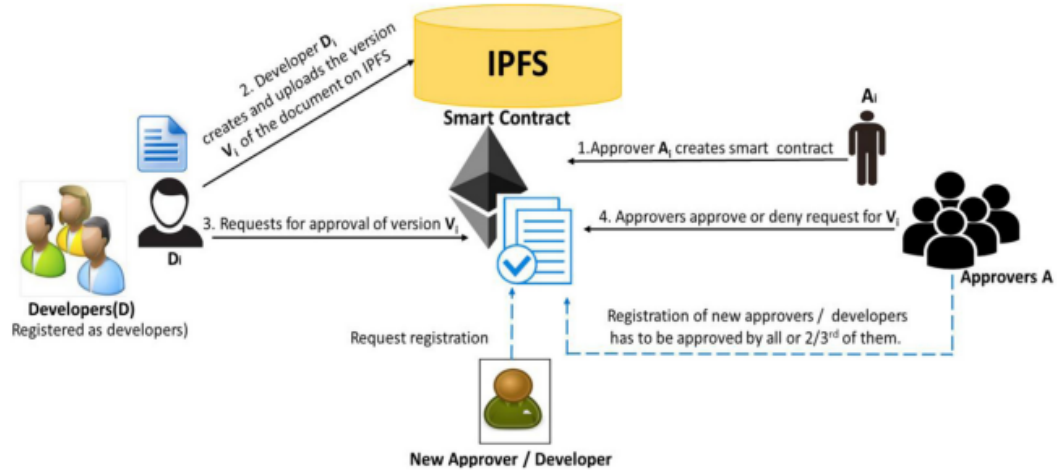
2.1 Penelitian Terkait

Terdapat beberapa penelitian terkait yang dijadikan acuan teori bagi penulis dalam menyusun tugas akhir ini. Teori-teori yang ada pada bab ini terkait *blockchain*, *InterPlanetary File System (IPFS)*, dan *remote update* yakni sebagai berikut:

2.1.1 *Decentralized document version control using ethereum blockchain and IPFS* [3]

Penelitian dengan judul “*Decentralized document version control using ethereum blockchain and IPFS*” yang dilakukan oleh N. Nizamuddin, K. Salah, M. Ajmal Azad, J. Arshad, dan M.H. Rehman ini merancang sebuah sistem untuk mengatur versi suatu dokumen dengan memanfaatkan teknologi *blockchain* dan IPFS. Lewat sistem yang dirancang diharapkan dapat menyelesaikan permasalahan pada sistem berbasis *centralized* yang saat ini banyak digunakan. Penggunaan *blockchain* dan IPFS dapat menyelesaikan permasalahan *single point of failure*, *tampered* pada riwayat dokumen, dan *time consumption* yang lebih lama ketika *approver* harus me-*review* dokumen yang ditambahkan atau diubah yang semuanya dapat dialami pada sistem *centralized*. Sistem *decentralized* yang dibuat menggunakan *smart contract* yang di *deploy* pada jaringan *blockchain* Ethereum dan menyimpan dokumen pada tempat penyimpanan IPFS. Cara kerja dari sistem ini adalah *approver master* membuat *smart contract* dan melakukan *deploy* ke jaringan Ethereum. Kemudian, orang dengan *address* yang teregistrasi sebagai *developer* yang akan melakukan perubahan/ pembuatan versi baru atas dokumen yang ada akan mengunggah dokumen ke IPFS lalu melakukan *request* kepada *smart contract* untuk memperoleh *approval*. Saat memanggil fungsi *request*

approval pada *smart contract* disertai juga IPFS hash dari *initial document* sebagai parameter.



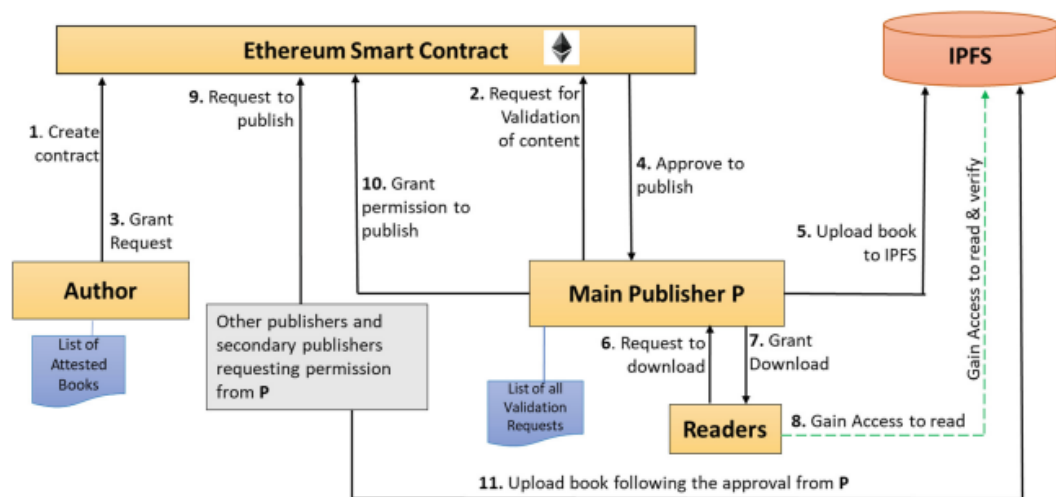
Gambar 2.1 Cara Kerja Sistem

Dari penelitian ini dapat diperoleh beberapa kesimpulan sebagai berikut:

- *Smart contract* yang di *deploy* pada jaringan *blockchain* dapat menjadi sebuah validator otomatis tanpa melibatkan pihak eksternal karena *smart contract* telah memiliki beberapa syarat-syarat yang harus dipatuhi oleh kedua belah pihak dalam penelitian ini kemudian kita sebut sebagai *developer* dan *approval* dalam bentuk kode yang hanya dapat dieksekusi apabila syarat-syarat tersebut terpenuhi.
- *Validator* atau dalam penelitian ini disebut sebagai *approver* adalah pihak-pihak (*node*) yang berada di dalam jaringan *blockchain* yang akan melakukan tugas untuk *mining* untuk memverifikasi sehingga terbentuk sebuah blok yang baru.

2.1.2 *IPFS-blockchain-based authenticity of online publications* [4]

Penelitian dengan judul “*IPFS-blockchain-based authenticity of online publications*” yang disusun oleh Nishara Nizamuddin, Haya R. Hasan, dan Khaled Salah ini membahas tentang rancangan sistem untuk melakukan autentikasi terhadap publikasi dari suatu karya tulis atau buku yang dimana setiap karya tulis disimpan ke dalam IPFS dan memperoleh IPFS hash. Dalam rancangan ini terdapat beberapa pihak yang terlibat, yaitu *Author*, *Main Publisher*, *Second/ Other Publisher*, dan *Readers*. *Author* adalah pihak yang membuat *smart contract* yang berarti menetapkan syarat-syarat yang harus dipenuhi oleh pihak lainnya ketika berkomunikasi dengan *smart contract* (saat ingin melakukan sesuatu terhadap karya tulis buatannya). Ketika *Main Publisher* memperoleh akses dari *smart contract* untuk mengunggah karya tulis yang dibuat oleh *Author*, maka karya tulis akan di unggah ke IPFS. Setelah itu apabila ada *Other Publisher* yang ingin membuat versi lain (yang sudah diterjemahkan ke bahasa lain) dari karya tulis tersebut, maka *Other Publisher* harus mengajukan izin kepada *Author* dan *Main Publisher* lewat *smart contract* untuk boleh mengunggah versi baru ke IPFS. Dari sisi *Readers* juga dapat melakukan pengajuan akses terhadap karya tulis yang ingin dibaca/ diunduh dari IPFS. Pembaca dapat mengakses riwayat dan keaslian karya tulis berkat penerapan teknologi *blockchain* yang menyimpan IPFS *hash* sehingga setiap data di IPFS bersifat *immutable*.



Gambar 2.2 Diagram komunikasi antar entitas

Dari penelitian ini dapat diperoleh beberapa kesimpulan sebagai berikut:

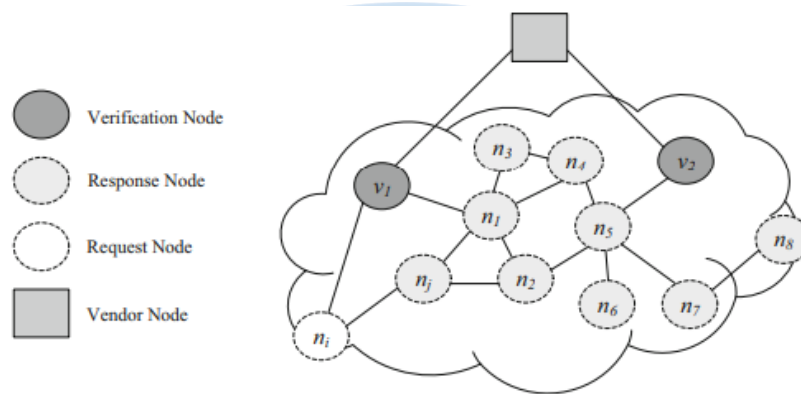
- *Smart contract* dapat menunjang pemberian akses dengan baik karena melalui *smart contract* kita dapat mengecek identitas dari pengirim *request* ke dalam jaringan *blockchain* yang bersifat *decentralized* dan *immutable* sehingga tahan terhadap pemalsuan data.
- *IPFS hash* atau *content identifier* yang terbentuk dari konten yang disimpan dalam *blockchain* sangat cocok untuk menjaga integritas dan originalitas dari konten yang disimpan karena apabila konten diubah maka *content identifier* pun akan berubah.
- Integrasi antara *Decentralized Applications* (DApps) dan *smart contract* diperlukan agar penggunaan dapat lebih mudah dilakukan karena dalam penelitian ini belum dikembangkan dari sisi *frontend*.

2.1.3 *Blockchain-based secure firmware update for embedded devices in an Internet of Things environment* [5]

Penelitian dengan judul “*Blockchain-based secure firmware update for embedded devices in an Internet of Things environment*” yang ditulis oleh Jong Hyouk Lee dan Boohyung Lee ini berisikan rancangan dari sebuah sistem *secure firmware update scheme* yang melibatkan 3 jenis entitas yang berbeda. Entitas yang dimaksud adalah :

- *Blockchain node*: *node* yang berada di dalam jaringan *blockchain*.
- *Normal node*: *embedded device* yang akan melakukan *update firmware* di dalam jaringan *blockchain*.
- *Verification node*: *node* yang mengatur informasi *firmware* seperti validator dan data *firmware* untuk setiap model *device*.
- *Vendor node*: *node* yang berada diluar jaringan *blockchain* dan terhubung secara aman dengan *verification node* untuk menyediakan data dari *latest firmware*.

Embedded device nantinya akan melakukan *request* untuk melakukan *firmware update* ke *blockchain nodes* dan akan menerima *response* apakah *firmware* yang saat ini digunakan *up-to-date* atau tidak. Jika *firmware* tidak *up-to-date* maka akan terjadi pengunduhan data *firmware* terbaru yang disediakan secara *peer-to-peer firmware sharing network* yang terdiri dari *blockchain node*. Sedangkan jika *firmware* yang dimiliki sekarang *up-to-date* maka *blockchain node* akan mengirimkan *firmware version check request*.



Gambar 2.3 Arsitektur sistem

Dari penelitian ini dapat ditarik beberapa kesimpulan sebagai berikut:

- Proses *firmware update* dilakukan dari *device* sehingga proses *update* yang terjadi secara *by request*.
- Penempatan *embedded device* sebagai *node* yang berada di dalam jaringan *blockchain* membuat komunikasi semakin aman untuk dilakukan namun muncul keterbatasan dari sisi spesifikasi *embedded device* yang digunakan karena *microcontroller* yang kurang *powerfull* seperti ESP32 belum kompatibel untuk menjadi sebuah *node* yang terhubung langsung dengan jaringan *blockchain*.

2.2 Tinjauan Teori

Dalam proses pembuatan penelitian ini, penulis juga meninjau beberapa teori ataupun teknologi yang digunakan. Berikut beberapa teori yang ditinjau adalah:

2.2.1 Blockchain

Kehadiran blockchain telah membuka pintu bagi revolusi teknologi yang kian hari semakin canggih dan luas penggunaannya. [6] Sejak awal

dipopulerkannya teknologi blockchain oleh seseorang dengan username Satoshi Nakamoto pada 2008 dengan menciptakan sebuah jaringan Bitcoin *blockchain* untuk sistem pembayaran elektronik, *blockchain* telah menjadi salah satu teknologi yang penggunaannya telah dilakukan secara masif dan ke berbagai bidang yang ada. *Blockchain* merupakan buku besar *digital* terdesentralisasi yang berisi rangkaian data transaksi dan terdistribusi sehingga dikelola oleh banyak pihak yang berpartisipasi pada suatu jaringan *blockchain*. [7] Penamaan *blockchain* sendiri diambil dari cara bagaimana teknologi ini menyimpan data transaksi, yaitu di dalam blok-blok (*block*) yang terhubung satu sama lain hingga membentuk sebuah rantai (*chain*). Semakin banyak transaksi yang dilakukan maka semakin panjang pula rantai *blockchain* dalam suatu jaringan yang bekerja sesuai dengan aturan yang telah disepakati bersama oleh seluruh partisipan jaringan. Mekanisme kerja dari *blockchain* sendiri tidak melibatkan otoritas sentral layaknya bank, melainkan melibatkan jaringan komputer *peer-to-peer* untuk proses verifikasi. Hal ini membuat teknologi ini menjadi terdesentralisasi yakni data dibagikan kepada seluruh komputer yang berpartisipasi.

Proses verifikasi pada *blockchain* disebut sebagai *signature* atau penandatanganan secara *digital* menggunakan kriptografi. Kriptografi sendiri merupakan proses enkripsi yang menjadi salah satu dasar dari teknologi *blockchain* karena banyak komponen *blockchain* yang menggunakan kriptografi, seperti *wallet address*, *previous block hash*, *private key*, *phrase seed*, dan lainnya.

Dalam teknologi *blockchain* juga dikenal sebuah algoritma konsensus. Konsensus adalah mekanisme yang disepakati bersama dalam suatu jaringan yang akan digunakan seluruh entitas di dalam *blockchain* untuk menyetujui penambahan data baru di dalam jaringan. [8] Di dalam pengaplikasian *blockchain* sendiri, terdapat dua permasalahan yang harus dipecahkan, yaitu *double spending* dan *Byzantine Generals Problem*. *Double spending* berarti masalah dimana terjadi penggunaan kembali sebuah nominal di dalam dua transaksi yang berbeda di satu waktu yang

bersamaan. Sedangkan, *Byzantine Generals Problem* adalah masalah yang terdapat pada *distributed system* dimana data dikirimkan ke antar *node* yang berbeda melalui *peer-to-peer communication* dan ada kemungkinan suatu atau beberapa *node* diserang dan mengalami perubahan isi data maka *node* harus bisa membedakan mana data yang sudah diubah dan memperoleh data yang asli dari *node* yang tidak terserang. Permasalahan ini dapat terpecahkan dengan menggunakan algoritma konsensus yang tepat sehingga fundamental dari *blockchain* dapat terlaksana dengan baik.

Hingga saat ini, sudah terdapat beberapa jenis konsensus namun hanya beberapa konsensus saja yang umum digunakan seperti:

- *Proof of Work* (PoW)

PoW adalah konsensus yang paling umum digunakan karena menjadi konsensus yang digunakan pada jaringan *blockchain* terbesar pertama yaitu jaringan *Bitcoin*. Konsensus ini sendiri bekerja dengan menerapkan kompetisi kepada banyak *node* dengan berbekal informasi *previous block* untuk memecahkan sebuah persoalan matematika yang nantinya akan menjadi *hash* untuk blok yang baru atau sering kita sebut sebagai *mining*. *Node* atau *miner* yang berhasil pertama kali memecahkannya akan memperoleh hadiah *cryptocurrency Bitcoin*. [9] Hingga saat ini konsensus ini masih menjadi salah satu yang teraman karena hanya dapat dilakukan *tampering* apabila ada seseorang yang mengontrol 50% lebih dari *hashing power* di dunia dan memastikan dirinya lah yang akan berhasil menjadi orang pertama yang memecahkan persoalan matematika dan men-*generate* blok baru tetapi hal ini sangat sulit dilakukan karena biaya untuk melakukan hal itu sangat besar. Walaupun demikian, konsensus PoW memiliki kekurangan dimana sangat tidak efisien baik secara waktu maupun energi. Karena energi yang diperlukan untuk melakukan *mining* sangatlah besar. Maka dari itu, ditemukanlah beberapa konsensus yang lebih efektif dibandingkan PoW namun dengan standar keamanan yang tetap tinggi.

- *Proof of Stake (PoS)*

Konsensus PoS hadir untuk menyelesaikan persoalan konsumsi energi yang sangat berlebihan pada PoW. Kompetisi pada konsensus PoW sudah tidak ada lagi di PoS melainkan digantikan dengan pemilihan *stakeholders* secara acak di dalam jaringan untuk menjadi *validators*. *Stakeholder* yang nantinya terpilih sebagai *validator* akan memiliki kewenangan dan kesempatan untuk memvalidasi suatu transaksi yang ada kemudian akan memperoleh bayaran yang sering disebut sebagai *gas fee*. [10] Implementasi konsensus ini pada jaringan *blockchain* besar rencananya dilakukan pada jaringan *main net Ethereum 2.0* pada bulan Agustus 2022 setelah diundur dari jadwal rencana sebelumnya pada tahun 2019.

Pada jaringan *Ethereum 2.0* nantinya, *validator* akan dipilih berdasarkan seberapa banyak kepemilikannya terhadap *cryptocurrency ethereum*. Hal ini akan membuat perubahan besar pada salah satu jaringan *blockchain* terbesar saat ini karena praktik *mining* sudah tidak memungkinkan lagi untuk dilakukan berdampak pada perubahan konsensus yang ada. Namun, serangan 51% yang identik dengan serangan pada jaringan *blockchain* masih memungkinkan terjadi walaupun sudah menggunakan konsensus PoS. Tetapi hal ini tetap sangat sulit dilakukan karena untuk menguasai 51% jaringan sama artinya dengan memiliki jumlah *ethereum* sebanyak 51% dari total yang beredar yang berarti sangat mahal sekali.

Dalam proses berjalannya suatu jaringan *blockchain* terdapat beberapa jenis *node* yang terlibat secara *on chain* yang memiliki karakteristik dan fungsi yang berbeda-beda, yaitu :

- *Full node*: merupakan *nodes* yang menyimpan seluruh informasi di dalam jaringan *blockchain*, sehingga apabila ada data baru yang dimasukkan ke dalam jaringan maka *full node* akan memperoleh data tersebut juga. Dengan

keberadaan *full node* maka dapat dipastikan bahwa ketersediaan atau *availability* terhadap data-data yang disimpan di dalam *blockchain* sangat tinggi karena data tersebar ke jumlah *node* yang sangat banyak.

- *Lightweight node* atau *Simplify Payment Verification (SPV)*: merupakan *nodes* yang memiliki rangkuman transaksi terhadap suatu *address* tertentu sehingga data transaksi yang diperoleh sudah disaring terlebih dahulu sesuai dengan *address* yang dibuka di dalam suatu *wallet* tertentu.
- *Mining node*: merupakan *nodes* yang memiliki fungsi untuk menerima dan menyebarkan transaksi ke seluruh jaringan *blockchain* pada jaringan yang menggunakan konsensus *Proof-of-Work (PoW)*. *Nodes* tipe ini akan memecahkan persoalan matematika yang berarti berhasil melakukan *hashing block* terhadap suatu transaksi.

2.2.2 *Smart Contract*

[12] *Smart contract* merupakan program komputer yang memiliki *self-verifying*, *self-executing*, dan *tamper-resistant*. Konsep awal dari *smart contract* sendiri diperkenalkan oleh Nick Szabo pada tahun 1994. Kemunculan *blockchain* bukan hanya sekedar menjadi buku besar yang mencatat transaksi yang terjadi dan menyebarkannya ke seluruh jaringan namun perkembangannya telah membawa kemungkinan untuk menjalankan suatu program di atas jaringan *blockchain*. Kehadiran jaringan *Ethereum* sebagai jaringan terbesar kedua setelah *Bitcoin* telah memungkinkan sebuah kontrak yang berupa program komputer untuk disimpan di dalam jaringan. Dengan men-*deploy smart contract*, maka kita dapat melakukan kustomisasi terhadap sebuah persetujuan yang harus disepakati kedua belah pihak dalam proses transaksi, bahkan transaksi

yang dikirimkan bukan lagi berbentuk nilai mata uang melainkan data-data.

Saat ini ada aplikasi yang menggunakan *smart contract* sebagai pusat logika dalam proses kontrol atau yang sering kita sebut sebagai *backend* pada konsep lingkungan *centralized*. Aplikasi yang dinamai sebagai *Decentralized Applications* (DApps) ini berinteraksi dengan *smart contract* melalui ABI (*Application Binary Interface*) yakni layaknya API untuk *machine language level*. Hal ini disebabkan karena *smart contract* hidup dan berjalan di atas *blockchain* yang komunikasinya masih berada di *low-level language*. *Smart contract* akan menjadi *interface* antara *ByteCode* dan JSON RPC dimana *ByteCode* hasil kompilasi *smart contract* akan di simpan di jaringan *blockchain* dan protokol JSON RPC digunakan saat *frontend* memanggil *function* dari *smart contract*.

Terdapat beberapa opsi pilihan bahasa yang dapat digunakan untuk membuat *smart contract* namun salah satu bahasa yang paling umum dan banyak digunakan adalah *Solidity*. Selain *Solidity* ada juga bahasa lain seperti *Vyper* dan *Rust*. Dari segi *framework* dan *blockchain tools* terdapat beberapa yang umum digunakan seperti *Hardhat*, *Truffle*, dan *OpenZepelline*.

2.2.3 *InterPlanetary File System* (IPFS)

IPFS merupakan protokol dan jaringan *peer-to-peer* yang memungkinkan untuk menyimpan dan menyebarkan sebuah data ke dalam suatu *distributed file system*. Penggunaan IPFS dapat menjadi salah satu solusi untuk penyimpanan data di *blockchain* yang sangat terbatas dari sisi kapasitas karena pada umumnya hanya data berukuran kecil yang disimpan ke dalam *blockchain*. Cara kerja dari IPFS sendiri adalah dengan memotong-motong sebuah data menjadi data yang lebih kecil kemudian menyebarkannya ke *nodes* yang lainnya. Hal ini membuat *availability level* terhadap data yang disimpan di IPFS cukup tinggi karena apabila ada

salah satu *node* yang mati maka data dapat disediakan oleh *node* lainnya yang masih memiliki data tersebut.

[11] IPFS berjalan di atas *network layer* Libp2p yang merupakan sistem modular untuk mengatur protokol, spesifikasi, dan *library* yang digunakan dalam *development* jaringan *peer-to-peer*. Libp2p bekerja dengan menggunakan *multi address* yang akan berisi beberapa informasi *addressing* yang diperlukan seperti *IP address*, *IP version*, *transport layer*, dan *port*. Selain itu, Libp2p juga menggunakan *peer identity* yang menjadi identitas suatu *peer* di dalam jaringan yang disertai dengan *private key* dan *public key*.

Dari sisi penyimpanan, IPFS menerapkan *content addressing* sehingga tidak selayaknya tempat penyimpanan *centralized* yang ketika ingin diakses kita harus memberikan spesifikasi lokasi dari suatu data (*location addressing*) yang membuat apabila lokasi penyimpanan *down* maka data tidak dapat diakses atau sering disebut sebagai *single point of failure*. Dengan kekuatan *content addressing* yang dimiliki IPFS data dicari berdasarkan identitas dari data tersebut tanpa memperdulikan dari mana data tersebut diambil, asalkan identitas data *valid* maka data akan diberikan. Identitas dari data tersebut disebut dengan *Content Identifier* (CID) yang merupakan *hash* dari konten/ data yang disimpan dan setiap ada perubahan terhadap konten maka akan diproduksi CID yang baru sehingga CID ini juga berperan sebagai *versioning* di dalam IPFS. Pada umumnya CID di *hash* dengan algoritma SHA-256.

2.3 Summary

Berdasarkan hasil studi dan tinjauan terhadap teori *blockchain* dan IPFS serta penelitian terdahulu yang berhubungan dengan *remote update firmware* maka terdapat beberapa kesimpulan yang bisa penulis ambil, yang mendasari perancangan penelitian ini:

- Jaringan *blockchain* yang digunakan adalah jaringan *Ethereum*. Pemilihan jaringan ini didasarkan pada kemampuannya untuk menjalankan *smart contract* serta sudah memiliki banyak

blockchain framework yang mendukung proses *develop* seperti Geth dan Hardhat. Referensi dari penggunaan jaringan *ethereum* juga sudah banyak dan umum digunakan dalam penelitian.

- Sistem akan menggunakan IPFS sebagai tempat penyimpanan *file firmware*. Hal ini didasarkan pada kapabilitas *blockchain* yang buruk dalam menyimpan data karena tidak mampu menyimpan data dengan ukuran yang besar, sehingga penyimpanan *off-chain* atau di luar jaringan menjadi opsi pada penelitian ini. Selain itu, sifat *anti-tampering* pada data yang disimpan pada IPFS sama dengan sifat dari *blockchain*. IPFS memiliki *content identifier* (CID) yang menjadi identitas data yang kemudian CID ini disimpan pada jaringan *blockchain* agar identitas data dapat bergabung ke dalam jaringan *blockchain*.
- Rancangan akan memiliki *interface* berupa aplikasi web agar sistem memiliki *user interface* dalam penggunaannya. Hal ini pun didukung dengan adanya *application binary interface* (ABI) hasil dari kompilasi *smart contract* sehingga *frontend* dapat berkomunikasi dengan *smart contract* via ABI.
- *Device* yang digunakan adalah Raspberry Pi. Pemilihan jenis *device* ini dikarenakan sudah *powerful* dan mampu menjadi IPFS *node* sehingga proses integrasi dengan IPFS dan *blockchain* akan lebih mudah dilakukan.
- *Blockchain* memiliki 2 jenis *address* yang dapat digunakan untuk melakukan *transaction*, yaitu ***Externally Owned Account (EOA)*** dan ***Smart Contract Address (SCA)***. EOA merupakan *address* yang mewakili sebuah *wallet* dan SCA mewakili *smart contract*. Transaksi yang dapat dilakukan adalah EOA dengan EOA, EOA dengan SCA, dan EOA dengan EOA.