

**PENERAPAN *BLOCKCHAIN* DENGAN INTEGRASI *SMART CONTRACT*
PADA SISTEM *CROWDFUNDING***



TUGAS AKHIR

*Disusun dalam rangka memenuhi salah satu persyaratan
Untuk menyelesaikan program Strata-1 Departemen Informatika
Fakultas Teknik Universitas Hasanuddin
Makassar*

Disusun Oleh :

<u>FIQAR ARPIALIM</u>
D42115304

**DEPARTEMEN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS HASANUDDIN
MAKASSAR
2020**

LEMBAR PENGESAHAN

“PENERAPAN BLOCKCHAIN DENGAN INTEGRASI SMART
CONTRACT PADA SISTEM CROWDFUNDING”

Disusun Oleh :

FIQAR APRIALIM
D421 15 304

Skripsi ini telah dipertahankan pada Ujian Akhir Sarjana tanggal 2 Desember 2020. Diterima dan disahkan sebagai salah satu syarat memperoleh gelar Sarjana Teknik (S.T) pada Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin.

Gowa, 2 Desember 2020

Disetujui Oleh :

Pembimbing I,



Adnan, S.T., M.T., Ph.D.
NIP. 19740426 200501 1 002

Pembimbing II,



Dr. Eng. Ady Wahyudi Paundu, S.T., M.T.
NIP. 19750313 200912 1 003

Diterima dan disahkan oleh:



PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini :

Nama : FIQAR APRIALIM

NIM : D421 15 304

Departemen : SI Teknik Informatika

Menyatakan dengan sebenar-benarnya bahwa skripsi yang berjudul :

PENERAPAN *BLOCKCHAIN* DENGAN INTEGRASI *SMART CONTRACT* PADA SISTEM *CROWDFUNDING*

Adalah karya ilmiah saya sendiri dan sepanjang pengetahuan saya di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan/ditulis/diterbitkan sebelumnya. Kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila dikemudian hari ternyata di dalam naskah skripsi ini terdapat unsur-unsur jiplakan, saya bersedia menerima sanksi atas perbuatan tersebut dan diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2000, pasal 25 ayat 2 dan pasal 70)

Gowa, 2 Desember 2020

Yang Membuat Pernyataan



FIQAR APRIALIM

ABSTRAK

Popularitas *crowdfunding* yang semakin berkembang menyebabkan peningkatan persaingan antara berbagai *platform crowdfunding* dalam menyediakan sistem yang baik dan efisien. Sebuah proyek galang dana umumnya melibatkan transaksi keuangan yang jumlahnya tidak sedikit. Dengan demikian, keamanan data dan transparansi dari transaksi keuangan yang terjadi merupakan hal utama yang perlu ditingkatkan. Selain itu, perhitungan biaya pemrosesan proyek galang dana yang umumnya diterapkan pada sistem *crowdfunding* masih terbilang kurang optimal karena ditetapkan dengan tarif berbasis persentase berdasarkan jumlah dana yang diterima oleh setiap proyek. Teknologi *blockchain* kemudian hadir untuk memberikan keamanan dan transparansi pada sistem transaksi keuangan. Walaupun demikian, penerapan teknologi *blockchain* pada sistem *crowdfunding* belum cukup karena proses yang terjadi dalam sistem tidak hanya sekedar proses transaksi keuangan dasar, tetapi terdapat protokol penggalangan dana yang perlu diterapkan dalam proses transaksi tersebut. *Smart contract*, yaitu kontrak berbentuk perangkat lunak yang terotomatisasi, kemudian mulai diintegrasikan ke teknologi *blockchain* untuk dapat memenuhi kebutuhan suatu sistem dalam melakukan berbagai macam bentuk pemrosesan dengan protokol *blockchain*. Integrasi *smart contract* memungkinkan implementasi *blockchain* pada sistem *crowdfunding* dapat dilakukan. Penerapan kedua teknologi ini dilakukan dengan menggunakan *platform blockchain Ethereum*. Sistem *crowdfunding* yang menggunakan arsitektur *blockchain* dapat memberikan transparansi pada setiap kegiatan pemberian dana dan keamanan data transaksi karena aksesibilitas data yang bersifat publik dan sifat terdesentralisasi *blockchain*. Sistem *crowdfunding* yang dibangun dalam studi ini mampu mengoptimalkan biaya pemrosesan proyek galang dana karena setiap pemrosesan dilakukan secara otomatis menggunakan *smart contract* dengan biaya yang ditetapkan berdasarkan tarif sama rata. Hasil perbandingan menunjukkan pengoptimalan biaya pemrosesan pembentukan proyek galang dana tercapai untuk total dana yang besarnya sekitar Rp2.000.000,00 atau lebih dan pengoptimalan biaya pemrosesan pemberian dana tercapai untuk jumlah pemberian dana yang besarnya sekitar Rp500.000,00 atau lebih.

Kata kunci: *crowdfunding, blockchain, smart contract, Ethereum, transaksi.*

KATA PENGANTAR

Puji dan syukur penulis panjatkan atas kehadiran Allah SWT karena berkat rahmat dan karunia-Nya sehingga tugas akhir yang berjudul “*Penerapan Blockchain Dengan Integrasi Smart Contract Pada Sistem Crowdfunding*” ini dapat diselesaikan sebagai salah satu syarat dalam menyelesaikan jenjang Strata-1 (S-1) pada Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin.

Penulis menyadari bahwa dalam penyusunan dan penulisan laporan tugas akhir ini tidak lepas dari bantuan, bimbingan serta dukungan dari berbagai pihak, dari masa perkuliahan sampai dengan masa penyusunan tugas akhir. Oleh karena itu, penulis dengan senang hati menyampaikan terima kasih kepada:

1. Orang tua penulis, Bapak Dr. dr. Arifin Seweng, MPH dan Ibu Dra. Nurbaeti, M.Kes. serta Saudara kandung penulis, Fadil Apriawan dan Nurfina Yuniar, yang selalu memberikan dukungan, doa, semangat dan kekuatan kepada penulis dalam menjalani masa perkuliahan, terlebih pada saat penggerjaan tugas akhir;
2. Bapak Adnan, S.T., M.T., Ph.D. selaku dosen pembimbing I dan Bapak Dr.Eng. Ady Wahyudi Paundu, S.T., M.T. selaku dosen pembimbing II yang selalu menyediakan waktu, tenaga, pikiran dan perhatian yang besar untuk mengarahkan penulis dalam penyusunan tugas akhir;
3. Bapak Dr. Amil Ahmad Ilham, S.T., M.IT. dan Bapak Iqra Aswad, S.T., M.T. selaku dosen penguji yang telah memberikan saran sehingga laporan tugas akhir ini menjadi lebih baik;

4. Bapak Dr. Amil Ahmad Ilham, S.T., M.IT. selaku Ketua Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin atas bimbingan yang diberikan selama masa perkuliahan penulis;
5. Segenap Staf Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin yang telah membantu segala urusan administrasi yang diperlukan selama masa perkuliahan penulis;
6. Teman-teman dan kakak-kakak Laboratorium UBICON, yang telah memberikan dukungan dan semangat;
7. Teman-teman Laboratorium IOT, yang telah memberikan dukungan dan semangat;
8. Teman-teman HYPERV15OR atas dukungan dan semangat yang telah diberikan.
9. Seluruh pihak yang tidak sempat penulis sebutkan satu persatu, yang telah meluangkan waktu, tenaga dan pikiran selama penyusunan laporan tugas akhir ini.

Akhir kata, penulis berharap semoga Allah SWT berkenan membala segala kebaikan dari semua pihak yang telah banyak membantu. Semoga tugas akhir ini dapat bermanfaat bagi para pembacanya.

Makassar, Juli 2020

Penulis

DAFTAR ISI

PENERAPAN <i>BLOCKCHAIN</i> DENGAN INTEGRASI <i>SMART CONTRACT</i> PADA SISTEM <i>CROWDFUNDING</i>	i
LEMBAR PENGESAHAN	ii
PERNYATAAN KEASLIAN.....	iii
ABSTRAK	iv
KATA PENGANTAR.....	v
DAFTAR ISI.....	vii
DAFTAR TABEL	x
DAFTAR GAMBAR.....	xi
BAB I PENDAHULUAN.....	1
1.1. Latar Belakang	1
1.2. Rumusan Masalah	4
1.3. Tujuan Penelitian.....	4
1.4. Manfaat Penelitian.....	5
1.5. Batasan Masalah.....	5
1.6. Sistematika Penulisan.....	5
BAB II TINJAUAN PUSTAKA	7
2.1. <i>Crowdfunding</i>	7
2.2. <i>Blockchain</i>	9
2.3. <i>Smart Contract</i>	20
2.4. <i>Ethereum</i>	22
2.4.1. <i>Account</i>	24

2.4.2.	Transaksi.....	25
2.4.3.	<i>Block</i>	27
2.4.4.	Eksekusi Transaksi	30
2.4.5.	Arsitektur <i>Blockchain</i>	31
	BAB III METODOLOGI PENELITIAN	33
3.1.	Waktu dan Lokasi Penelitian.....	33
3.2.	Instrumen Penelitian.....	33
3.3.	Gambaran Umum Sistem	35
3.3.1.	<i>Activity Diagram</i> Sistem.....	36
3.3.2.	Desain <i>Smart Contract</i>	45
3.3.3.	Detail Perancangan Sistem	47
3.4.	Skenario Analisis dan Pengujian.....	49
3.4.1.	Pengujian Fungsionalitas Sistem.....	49
3.4.2.	Analisis Keamanan & Transparansi Sistem	50
3.4.3.	Pengujian Besar Biaya Transaksi Sistem	50
	BAB IV HASIL DAN PEMBAHASAN	53
4.1.	<i>Cryptocurrency</i>	53
4.2.	Pengujian Fungsionalitas Sistem.....	54
4.2.1.	Pembentukan Proyek.....	54
4.2.2.	Pemberian Dana pada Proyek.....	56
4.2.3.	Pengambilan Dana Proyek	57
4.2.4.	Pengembalian Dana Proyek.....	58
4.3.	Analisis Keamanan dan Transparansi Sistem	59

4.3.1.	Dasar Teoritis	59
4.3.2.	Pengujian <i>Blockchain Ethereum</i>	64
4.3.2.1.	Pengujian Mayoritas Node Jujur (Kondisi Pertama).....	66
4.3.2.2.	Pengujian <i>Double Spending</i> (Kondisi Kedua).....	68
4.3.2.3.	Pengujian Rantai <i>Block</i> Tidak Valid (Kondisi Ketiga).....	71
4.3.3.	Analisis Terhadap Serangan Digital Lainnya	74
4.4.	Pengujian Besar Biaya Transaksi	77
4.4.1.	Pemrosesan Pembentukan Proyek	77
4.4.2.	Pemrosesan Pemberian Dana pada Proyek.....	80
4.4.3.	Evaluasi Hasil Pengujian Besar Biaya Pemrosesan	84
BAB V PENUTUP	90
5.1.	Kesimpulan.....	90
5.2.	Saran.....	91
DAFTAR PUSTAKA	92
LAMPIRAN	94

DAFTAR TABEL

Tabel 2. 1 Jenis <i>Blockchain</i>	15
Tabel 2. 2 Bidang Aplikasi Blockchain	18
Tabel 2. 3 Komponen Transaksi (Kasireddy 2017)	26
Tabel 3. 1 <i>Input</i> dan <i>Output</i> Pembentukan Proyek	38
Tabel 3. 2 <i>Input</i> dan <i>Output</i> Pemberian Dana pada Proyek.....	41
Tabel 3. 3 <i>Input</i> dan <i>Output</i> Pengambilan Dana Proyek	42
Tabel 3. 4 <i>Input</i> dan <i>Output</i> Pengembalian Dana Proyek.....	44
Tabel 3. 5 <i>Variable</i> dan <i>Function</i> dari Contract Crowdfunding.....	45
Tabel 3. 6 <i>Variable</i> dan <i>Function</i> dari Contract Project	46
Tabel 4. 1 Hasil <i>Black Box Testing</i> Pembentukan Proyek.....	55
Tabel 4. 2 Hasil <i>Black Box Testing</i> Pemberian Dana pada Proyek	56
Tabel 4. 3 Hasil <i>Black Box Testing</i> Pengambilan Dana Proyek	58
Tabel 4. 4 Hasil <i>Black Box Testing</i> Pengembalian Dana Proyek.....	58
Tabel 4. 5 Jenis Serangan yang Teratas.....	74
Tabel 4. 6 Hasil Pengujian Nilai <i>gasUsed</i> untuk Lima Kondisi Pemrosesan Pembentukan Proyek	78
Tabel 4. 7 Hasil Pengujian Nilai <i>gasUsed</i> untuk Tiga Kondisi Pemrosesan Pemberian Dana pada Proyek	81
Tabel 4. 8 Pembebatan Biaya pada Beberapa Sistem <i>Crowdfunding</i>	84

DAFTAR GAMBAR

Gambar 2. 1 Ilustrasi <i>Blockchain</i>	11
Gambar 2. 2 Detail <i>Block</i> pada <i>Blockchain</i>	12
Gambar 2. 3 Komputasi <i>Block Hash</i>	13
Gambar 2. 4 Arsitektur <i>Decentralized Application</i>	22
Gambar 2. 5 <i>Block Header</i>	30
Gambar 3. 1 Gambaran Umum Sistem	35
Gambar 3. 2 <i>Use Case Diagram</i> Sistem	36
Gambar 3. 3 <i>Activity Diagram</i> Pembentukan Proyek.....	38
Gambar 3. 4 <i>Activity Diagram</i> Pemberian Dana pada Proyek.....	40
Gambar 3. 5 <i>Activity Diagram</i> Pengambilan Dana Proyek	42
Gambar 3. 6 <i>Activity Diagram</i> Pengembalian Dana Proyek.....	43
Gambar 3. 7 Pengembangan Aplikasi <i>Web</i> Sistem <i>Crowdfunding</i>	48
Gambar 3. 8 Pengembangan <i>Smart Contract</i> Sistem <i>Crowdfunding</i>	49
Gambar 4. 1 Prosedur Transaksi Mata Uang Fiat	53
Gambar 4. 2 Prosedur Transaksi <i>Cryptocurrency</i>	53
Gambar 4. 3 Proses Pencatatan Data <i>Blockchain</i>	62
Gambar 4. 4 Struktur Rantai <i>Block</i>	62
Gambar 4. 5 Konfigurasi <i>Genesis Block</i>	65
Gambar 4. 6 Konfigurasi <i>Node</i> Pengujian	66
Gambar 4. 7 Gambaran Pengujian Mayoritas <i>Node</i> Jujur (Kondisi Pertama).....	67
Gambar 4. 8 Rantai <i>Block</i> Setelah Proses <i>Mining</i> (Pengujian Kondisi Pertama). .	67
Gambar 4. 9 Sinkronisasi Rantai <i>Block Node</i> 3 (Pengujian Kondisi Pertama)....	68

Gambar 4. 10 Pembentukan Double Spending (Pengujian Kondisi Kedua)	69
Gambar 4. 11 <i>Block Data</i> Transaksi 1 di Node 2 (Pengujian Kondisi Kedua)	70
Gambar 4. 12 <i>Block Data</i> Transaksi 2 di Node 3 (Pengujian Kondisi Kedua).....	70
Gambar 4. 13 Sinkronisasi Rantai <i>Block</i> (Pengujian Kondisi Kedua).....	71
Gambar 4. 14 <i>Block Data</i> Transaksi <i>Double Spending</i> Setelah Sinkronisasi (Pengujian Kondisi Kedua).....	71
Gambar 4. 15 Penghapusan Data Chaindata Geth (Pengujian Kondisi Ketiga) ...	72
Gambar 4. 16 Jumlah <i>Block Node</i> 2 Setelah Penghapusan Data (Pengujian Kondisi Ketiga).....	73
Gambar 4. 17 <i>Output Command Peer</i> Pada Node 2 (Pengujian Kondisi Ketiga)	73
Gambar 4. 18 Besar Biaya Transaksi (<i>transactionFee</i>) Eksekusi Pemrosesan Pembentukan Proyek dalam <i>Ether</i>	80
Gambar 4. 19 Besar Biaya Transaksi (<i>transactionFee</i>) Eksekusi Pemrosesan Pemberian Dana pada Proyek dalam <i>Ether</i>	83
Gambar 4. 20 Perbandingan Biaya Pemrosesan Pembentukan Proyek Galang Dana	87
Gambar 4. 21 Perbandingan Biaya Pemrosesan Pemberian Dana pada Proyek Galang Dana.....	88

BAB I

PENDAHULUAN

1.1. Latar Belakang

Crowdfunding merupakan suatu metode baru dalam melakukan pengumpulan dana yang diperoleh dari kontribusi masyarakat dalam memenuhi suatu tujuan tertentu. *Crowdfunding* memberikan kesempatan kepada masyarakat dalam menyalurkan bantuan uang ataupun materi untuk suatu kepentingan yang dianggap sedang membutuhkan. Kepentingan ini dapat berupa kepentingan yang bersifat personal dan kepentingan umum yang mencakup banyak orang.

Platform crowdfunding mulai dikembangkan oleh berbagai instansi untuk memudahkan masyarakat dalam melakukan penyaluran donasi untuk suatu kepentingan yang dianggap membutuhkan. Beberapa contoh *platform crowdfunding* yang ada saat ini yaitu Kickstarter, GoFundMe, dan Kitabisa.

Popularitas *crowdfunding* yang semakin berkembang menyebabkan peningkatan persaingan antara berbagai *platform crowdfunding* untuk menyediakan sistem yang baik dan efisien sebagai media yang digunakan oleh masyarakat dalam melakukan kegiatan penggalangan dana. Popularitas *crowdfunding* dapat dilihat pada contoh perkembangan Kitabisa, dimana berdasarkan *Kitabisa Online Giving Report 2018*, tercatat peningkatan penyaluran donasi yang dilakukan masyarakat setiap tahunnya selalu meningkat lebih dari 100%.

Perkembangan *platform crowdfunding* yang dampaknya begitu besar di masyarakat mengharuskan bahwa *platform* tersebut memiliki sistem yang dapat

dipercaya dan aman untuk digunakan oleh masyarakat dalam melakukan kegiatan penggalangan dana. Dalam pengembangan sistem *crowdfunding*, properti utama yang perlu diperhatikan yaitu keamanan data dan transparansi transaksi keuangan yang terjadi pada kegiatan penggalangan dana.

Transparansi pada transaksi keuangan yang terjadi dalam sistem *crowdfunding* dapat memberikan kepercayaan kepada pengguna dalam melakukan aktivitas pemberian dana pada suatu penggalangan dana. Sementara itu, keamanan data memberikan proteksi terhadap proses penggalangan dana yang terjadi dalam sistem *crowdfunding* sehingga berjalan secara sesuai tanpa adanya gangguan dari pihak yang tidak diketahui. Kedua properti ini sangat rentan dan dibutuhkan oleh pengguna sistem *crowdfunding* karena dalam kegiatan penggalangan dana terdapat proses transaksi keuangan yang jumlahnya tidak sedikit. Jika terjadi gangguan dari pihak yang tidak diketahui, maka kerugian yang dapat ditimbulkan akan berdampak besar pada banyak orang.

Properti transparansi dan keamanan pada sistem *crowdfunding* tidak dapat tercapai secara maksimal apabila penerapan dari sistem *crowdfunding* masih dilakukan secara tersentralisasi (*centralized*). Artinya, otoritas dari sistem hanya dipegang oleh pihak tertentu saja. (Kaushik, et al. 2017)

Selain properti transparansi dan keamanan, pengoptimalan biaya pemrosesan juga dibutuhkan pada sistem *crowdfunding* yang ada saat ini. Seluruh proses kontrak penggalangan dana masih dilakukan dengan bantuan dari sistem milik pihak ketiga, yang dalam hal ini adalah *platform crowdfunding*. Sistem *crowdfunding* umumnya menerapkan biaya pemrosesan pada suatu penggalangan

dana sebagai bentuk bayaran atas mengerjakan pemrosesan yang dibutuhkan dalam kegiatan galang dana. Besar biaya pemrosesan ini berkisar antara 3% sampai dengan 5% dari total donasi yang diterima pada suatu penggalangan dana. Model perhitungan biaya pemrosesan yang memiliki tarif berbasis persentase tidak menguntungkan bagi penggalangan dana yang memiliki dana dengan jumlah besar. Semakin besar dana yang diterima oleh penggalangan dana maka akan semakin besar juga biaya pemrosesan yang perlu dibayar.

Pada penelitian yang dilakukan oleh Satoshi Nakamoto (2008) dengan judul *Bitcoin: A Peer-to-Peer Electronic Cash System*, terdapat konsep teknologi yang disebut dengan *blockchain*. *Blockchain* memungkinkan suatu sistem untuk dikembangkan secara terdesentralisasi (*decentralized*) sehingga otoritas dari sistem tidak dipegang oleh satu pihak saja, tetapi semua entitas di dalam sistem memiliki hak otoritas yang sama. Sifat terdesentralisasi yang dimiliki oleh *blockchain* memberikan dampak yang signifikan pada properti keamanan dan transparansi dari suatu sistem.

Pengembangan *blockchain* ini awalnya diberlakukan untuk sistem transaksi mata uang digital, atau biasa disebut dengan *cryptocurrency*. Sistem ini memungkinkan suatu transaksi dapat dilakukan tanpa adanya pihak ketiga yang bertugas untuk melakukan validasi dalam menentukan keabsahan dari suatu transaksi. Validasi dari transaksi dilakukan bersama oleh seluruh *node* berupa entitas-entitas yang terhubung pada jaringan sistem. Pencatatan transaksi ini akan tersimpan di seluruh *node* yang tergabung pada jaringan sistem. Dengan kata lain, penyimpanan data tidak terpusat di satu tempat penyimpanan.

Dalam penerapan *blockchain* saat ini, terdapat berbagai macam pengembangan yang telah diterapkan, salah satunya adalah pengintegrasianya dengan *smart contract*. *Smart contract* adalah perangkat lunak terotomatisasi berisi protokol kesepakatan antara dua pihak atau lebih yang dikelola menggunakan sistem terdesentralisasi.

Berdasarkan permasalahan yang ada pada sistem *crowdfunding* dan pengembangan yang terjadi pada teknologi *blockchain*, maka dalam Tugas Akhir ini akan dibangun sistem *crowdfunding* dengan memanfaatkan penggunaan teknologi *blockchain* untuk meningkatkan properti transparansi kegiatan penggalangan dana dan keamanan data yang ada pada sistem, serta integrasi *smart contract* untuk mengoptimalkan biaya pemrosesan yang umumnya diterapkan pada berbagai sistem *crowdfunding*.

1.2. Rumusan Masalah

Rumusan masalah yang akan diuraikan dalam Tugas Akhir ini adalah sebagai berikut:

1. Rendahnya transparansi dan keamanan yang ada pada sistem *crowdfunding* saat ini.
2. Besarnya biaya pemrosesan penggalangan dana yang umumnya diterapkan pada sistem *crowdfunding* saat ini.

1.3. Tujuan Penelitian

Tujuan yang akan dicapai dalam Tugas Akhir ini adalah sebagai berikut:

1. Membangun sistem *crowdfunding* dengan penerapan teknologi *blockchain* sehingga dapat meningkatkan transparansi dan keamanan.

2. Mengintegrasikan *smart contract* pada sistem *crowdfunding* untuk mengoptimalkan biaya pemrosesan penggalangan dana.

1.4. Manfaat Penelitian

Tugas Akhir ini diharapkan dapat memberikan manfaat sebagai berikut:

1. Dari segi akademis, Tugas Akhir ini dapat digunakan sebagai referensi ilmiah terkait informasi mengenai proses pengembangan dan keuntungan penerapan *blockchain* dengan integrasi *smart contract* pada sistem *crowdfunding*.
2. Dari segi praktis, arsitektur sistem *crowdfunding* yang dibangun dalam Tugas Akhir ini dapat digunakan sebagai acuan dalam membangun suatu *platform crowdfunding* yang optimal.

1.5. Batasan Masalah

Batasan-batasan masalah yang ditetapkan dalam Tugas Akhir ini adalah sebagai berikut:

1. Sistem yang diterapkan menggunakan aplikasi sisi klien berbasis *website*.
2. Sistem dibangun dengan menggunakan *platform blockchain Ethereum*.
3. *Smart contract* sistem dibangun dengan menggunakan bahasa pemrograman Solidity.

1.6. Sistematika Penulisan

Sistematika penulisan yang digunakan dalam Tugas Akhir ini terbagi menjadi beberapa pokok bahasan, yaitu :

BAB I PENDAHULUAN

Bab ini memberikan gambaran terkait latar belakang masalah penelitian, rumusan masalah yang diuraikan, tujuan penelitian yang akan dicapai, manfaat penelitian yang diharapkan, batasan penelitian yang ditetapkan, dan sistematika penulisan penelitian.

BAB II TINJAUAN PUSTAKA

Bab ini memberikan kajian-kajian pustaka yang berkaitan dengan topik penelitian dan teori-teori dari berbagai referensi ilmiah yang digunakan dalam pelaksanaan penelitian.

BAB III METODE PENELITIAN

Bab ini memberikan gambaran terkait perancangan sistem yang akan direalisasikan beserta metode pengujinya.

BAB IV HASIL DAN PEMBAHASAN

Bab ini menguraikan hasil penelitian yang telah dilaksanakan.

BAB V PENUTUP

Bab ini berisi kesimpulan dari hasil penelitian dan saran untuk pengembangan penelitian lebih lanjut.

BAB II

TINJAUAN PUSTAKA

2.1. *Crowdfunding*

Salah satu manfaat utama yang diberikan oleh bank adalah ketersediaannya sebagai salah satu sumber pinjaman dana untuk masyarakat yang sedang membutuhkan. Kebutuhan pendanaan ini dapat berupa berbagai macam hal, salah satunya yaitu untuk mengembangkan usaha yang sedang dijalani oleh peminjam. Manfaat ini tentunya dapat diberikan apabila memenuhi beberapa persyaratan yang ditetapkan oleh bank, dimana syarat tersebut belum tentu dapat dipenuhi oleh individu ataupun kelompok masyarakat yang membutuhkan pendanaan.

Seiring dengan berkembangnya zaman, terbentuk metode baru dalam melakukan pengumpulan dana yang diperoleh dari kontribusi masyarakat dalam memenuhi suatu tujuan tertentu. Metode ini disebut sebagai *crowdfunding* atau bisa diartikan sebagai penggalangan dana. *Crowdfunding* umumnya dilakukan oleh suatu individu atau kelompok dengan menggunakan bantuan media publikasi untuk menyebarluaskan informasi terkait penggalangan dana yang dilakukan ke masyarakat. Media informasi ini dapat berupa media sosial atau media yang khusus digunakan sebagai media *crowdfunding*, seperti Kickstarter, GoFundMe, Kitabisa, dan sebagainya.

Crowdfunding terbagi menjadi dua jenis yang dibedakan berdasarkan bentuk kesepakatan antara penggalang dana dan pemberi dana, yaitu *crowdfunding* yang memiliki upah yang akan diberikan kepada pemberi dana

apabila target total pendanaan yang ditetapkan tercapai dan *crowdfunding* yang tidak memiliki upah sama sekali, dengan kata lain dana yang diberikan tergolong sebagai donasi. (Belleflamme, Lambert and Schwienbacher 2013)

Crowdfunding yang dibentuk umumnya memiliki tujuan pendanaan yang jelas. Tujuan pendanaan ini dapat berdasarkan kebutuhan masyarakat di suatu daerah ataupun kebutuhan personal dari suatu individu dan kelompok. Tujuan pendanaan yang jelas dan sesuai dalam suatu penggalangan dana dapat meningkatkan insentif masyarakat dalam berkontribusi sebagai pemberi dana.

Insentif masyarakat untuk berkontribusi sebagai pemberi dana pada suatu *crowdfunding* merupakan hal utama yang harus ditingkatkan oleh penggalang dana untuk dapat mencapai target pendanaan yang ditetapkan. Selain daripada tujuan pendanaan yang jelas dan sesuai, intensif masyarakat dapat ditingkatkan cara lain seperti menyampaikan seluruh informasi yang berhubungan dengan *crowdfunding* yang diadakan secara jelas, meningkatkan publikasi dari *crowdfunding* yang diadakan sehingga dapat mencakup masyarakat yang lebih luas, dan menjaga kepercayaan masyarakat terkait kesesuaian seluruh kegiatan yang terjadi pada *crowdfunding* yang diadakan.

Media sosial merupakan media yang umum digunakan oleh masyarakat dalam melakukan publikasi *crowdfunding*. Media ini banyak digunakan oleh pihak penggalang dana karena kemudahannya dalam menyebarluaskan informasi dari *crowdfunding* yang diadakan kepada masyarakat. Walaupun demikian, penggunaan media sosial sebagai media publikasi *crowdfunding* masih memiliki kekurangan. Kekurangan ini yaitu media sosial tidak memiliki fitur yang dapat

memberikan jaminan bahwa *crowdfunding* yang diadakan oleh suatu individu atau kelompok bukan merupakan suatu rekayasa penipuan untuk mendapatkan keuntungan personal. Sebagai akibatnya, kepercayaan terhadap kebenaran dari *crowdfunding* yang diadakan tidak dapat terbentuk secara utuh.

Platform khusus *crowdfunding* saat ini telah banyak dikembangkan untuk memenuhi kebutuhan masyarakat dalam melakukan kegiatan penggalangan dana. Selain sebagai media publikasi, *platform* khusus ini menyediakan berbagai macam fitur untuk memudahkan masyarakat dalam membentuk *crowdfunding* secara efisien dan sesuai. Fitur yang disediakan dapat berupa verifikasi kebenaran dan keabsahan dari suatu *crowdfunding*, integrasi *payment gateway* untuk memudahkan kegiatan pemberian dana pada suatu *crowdfunding*, dan sebagainya.

2.2. *Blockchain*

Teknologi *blockchain* awalnya dikembangkan pada Bitcoin (Nakamoto 2008), yaitu sistem pembayaran elektronik pada jaringan *peer-to-peer* yang bersifat terdesentralisasi tanpa adanya institusi finansial yang bertindak sebagai pengatur jalannya transaksi. *Blockchain* ini diterapkan untuk menghilangkan kebutuhan institusi finansial sebagai pihak ketiga dalam pengelolaan suatu proses transaksi.

Blockchain pada dasarnya merupakan basis data transaksi yang terdistribusi pada berbagai *node* yang tergabung dalam suatu jaringan *peer-to-peer*. *Blockchhain* merupakan salah satu bentuk dari *Distributed Ledger Technology* (DLT), dimana teknologi ini bersifat terdesentralisasi dan memiliki protokol konsensus yang digunakan untuk mencapai kesepakatan bersama dalam

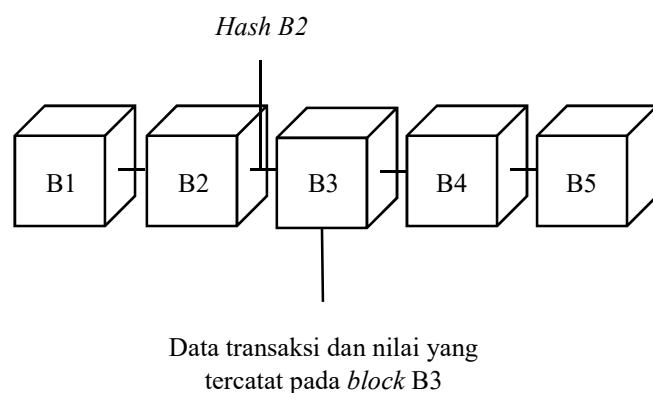
proses pengelolaan basis data yang ada. Akan tetapi, terdapat suatu perbedaan pada *blockchain* jika dibandingkan dengan DLT pada umumnya. Perbedaan ini terletak di struktur basis data yang ada pada *blockchain*, dimana setiap data transaksi yang tercatat akan tergabung ke dalam suatu *block* yang saling terhubung antara satu sama lain dan tidak dapat mengalami perubahan. (Hileman and Rauchs 2017)

Pencatatan data pada teknologi *blockchain* dilakukan melalui beberapa tahap, yaitu :

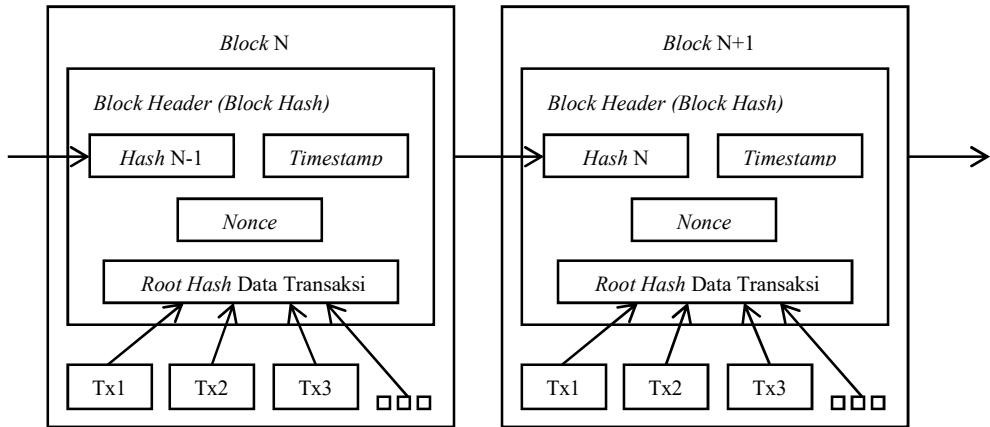
1. *Node* akan melakukan transaksi data menggunakan *digital signature* dan kemudian akan mengumumkannya ke jaringan. *Digital signature* merupakan tanda pengenal yang digunakan oleh suatu *node* dalam jaringan *blockchain*.
2. *Node* lain yang tergabung dalam jaringan akan menerima pengumuman transaksi dan kemudian menggabungkannya ke dalam suatu *block* baru.
3. *Node* penerima akan melakukan eksekusi pembentukan *block* berdasarkan protokol konsensus yang ditetapkan, seperti *Proof of Work*. Pembentukan *block* ini dikenal dengan istilah *mining*.
4. Setelah *node* penerima berhasil membentuk *block* baru berdasarkan protokol yang ditetapkan, selanjutnya *block* baru ini akan diumumkan ke jaringan sehingga dapat ditambahkan ke rantai *block* yang ada.

Pada *blockchain* sistem Bitcoin (Nakamoto 2008), setelah transaksi mata uang digital (*cryptocurrency*) dilakukan oleh suatu *node* menggunakan *digital signature*-nya, *node* tersebut selanjutnya akan mengumumkan transaksi yang

terjadi ke jaringan. *Node* lain kemudian akan menerima pengumuman transaksi-transaksi yang terjadi dan menggabungkannya dengan membentuk *block* menggunakan mekanisme protokol *Proof of Work*. Mekanisme *Proof of Work* akan membentuk suatu *block* baru yang terhubung ke *block* terakhir pada rantai *block* yang ada menggunakan fungsi *hash* kriptografi, seperti SHA-256. *Block* dibentuk dengan cara menghitung nilai *hash*-nya. Nilai *hash* ini biasa disebut sebagai *block hash* atau *block header*. *Block hash* akan didapatkan melalui komputasi fungsi *hash* dari nilai data transaksi yang tergabung dalam *block* dan beberapa nilai khusus, seperti *timestamp*, *nonce*, ataupun *block hash* dari *block* terakhir yang sebelumnya terbentuk pada rantai *block*. Hubungan antara suatu *block* dan *block* lainnya akan terbentuk dengan adanya penggunaan nilai *block hash* terakhir sebagai nilai masukan (*input*) dalam pembentukan nilai *block hash* baru. Ilustrasi *blockchain* dapat dilihat pada Gambar 2.1 dan Gambar 2.2.



Gambar 2. 1 Ilustrasi *Blockchain*



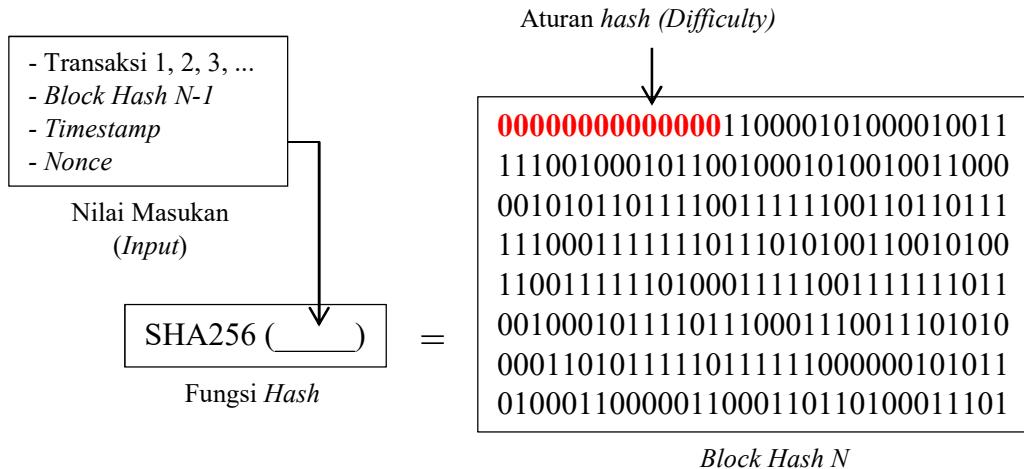
Gambar 2. 2 Detail Block pada Blockchain

Protokol konsensus *Proof of Work* merupakan mekanisme yang digunakan jaringan *blockchain* untuk mencapai kesepakatan bersama dalam pembentukan *block* data yang valid. Kesepakatan didapatkan berdasarkan besar komputasi yang digunakan dalam suatu proses pembentukan *block*. *Proof of Work* ini dapat diartikan sebagai bukti penggunaan komputasi yang besar dalam proses pembentukan suatu *block*.

Dalam mekanisme protokol *Proof of Work*, terdapat aturan yang perlu dipenuhi dalam perhitungan *block hash* dari suatu *block* baru. Aturan ini biasa disebut sebagai *difficulty*. *Difficulty* mengharuskan nilai dari suatu *block hash* memiliki beberapa angka nol pada bit awalnya. Dalam perhitungan nilai *block hash* yang sesuai berdasarkan *difficulty*, nilai masukan *nonce* merupakan kunci utama dalam penyelesaiannya. *Nonce* adalah nilai masukan yang ditentukan oleh *node* dalam melakukan komputasi *block hash*. Penggunaan nilai masukan yang dinamis seperti *nonce* akan dibutuhkan untuk mendapatkan *block hash* yang sesuai berdasarkan *difficulty* yang ditetapkan. Nilai *nonce* dibutuhkan oleh *node* dalam komputasi *block hash* karena nilai masukan lain seperti data utama,

timestamp, ataupun *block hash* dari *block* sebelumnya memiliki nilai yang tetap.

Gambaran komputasi *block hash* dapat dilihat pada Gambar 2.3.



Gambar 2.3 Komputasi *Block Hash*

Fungsi *hash* kriptografi seperti SHA-256 akan menghasilkan nilai *hash* yang sangat tidak terduga dan bersifat satu arah. Dengan kata lain, *hash* yang terbentuk tidak dapat terdekripsi untuk mendapatkan nilai masukan awal yang digunakan pada fungsinya. *Block hash* yang memenuhi kriteria *difficulty* yang ditetapkan hanya dapat dihasilkan dengan cara menjalankan fungsi *hash* secara berulang kali menggunakan *nonce* yang berbeda. *Node* akan saling berkompetisi dalam membentuk suatu *block* dengan cara mencari *nonce* yang sesuai sehingga *block hash* yang didapatkan dapat memenuhi kriteria *difficulty*. Proses ini akan membutuhkan komputasi yang besar. Dengan demikian, didapatkannya nilai *nonce* yang sesuai dapat menjadi bukti bahwa pembentukan *block* telah melalui proses komputasi yang besar. *Block* yang terbentuk dengan nilai *nonce* yang sesuai akan dianggap oleh jaringan sebagai *block* yang valid.

Umumnya, berdasarkan protokol *blockchain* yang ditetapkan, ketika suatu *block* berhasil dibentuk oleh suatu *node*, *node* tersebut akan mendapatkan hadiah berupa *cryptocurrency*. Akan tetapi, karena proses pembentukan *block* ini membutuhkan komputasi yang besar, *node* memiliki pilihan untuk melakukannya atau tidak. *Node* yang melakukan pembentukan *block* dikenal dengan istilah *miner*.

Ketika *miner* telah menemukan *nonce* yang sesuai, selanjutnya *block* yang dibentuk akan diumumkan ke seluruh *node* yang tergabung dalam jaringan. Seluruh *node* akan melakukan verifikasi terkait pembentukan *block* hanya dengan sekali menjalankan fungsi *hash* menggunakan nilai masukan yang telah didapatkan oleh *miner*. Apabila *block* yang terbentuk sesuai berdasarkan protokol yang ditetapkan, maka *block* tersebut akan dimasukkan ke dalam rantai *block* yang sudah ada.

Teknologi *blockchain* terus berkembang dan telah dikenal secara umum sebagai *framework* dalam pengembangan sistem yang bersifat terdesentralisasi (Gao, Hatcher and Yu 2018). Setiap pengembangan sistem terdesentralisasi akan menerapkan *blockchain* dengan berbagai jenis perubahan berdasarkan kebutuhan yang ada. Perubahan ini mencakup jenis protokol konsensus yang digunakan, implementasi konsep teknologi lain, mekanisme pencatatan data, dan sebagainya.

Blockchain memiliki berbagai macam jenis yang dibedakan berdasarkan tiga aspek, yaitu aksesibilitas data, partisipasi *node* dan fungsionalitasnya (Shrivastava and Yeboah 2018). Jenis-jenis *blockchain* dapat dilihat pada Tabel 2.1.

Tabel 2. 1 Jenis *Blockchain*

Jenis	Aspek	Penjelasan
<i>Public Blockchain</i>	Aksesibilitas data	Jenis <i>blockchain</i> ini memperbolehkan setiap individu/kelompok untuk bergabung sebagai <i>node</i> dalam melakukan pembacaan dan pencatatan data. (Lin and Liao 2017)
<i>Consortium Blockchain</i>	Aksesibilitas data	Jenis <i>blockchain</i> ini pada dasarnya bersifat tertutup, tetapi dapat memperbolehkan individu/kelompok tertentu yang tergabung dalam konsorsium untuk berpartisipasi sebagai <i>node</i> dalam melakukan pembacaan dan pencatatan data. (Lin and Liao 2017)
<i>Private Blockchain</i>	Aksesibilitas data	Jenis <i>blockchain</i> ini bersifat tertutup dan hanya ada satu pihak <i>node</i> saja yang dapat melakukan pembacaan dan pencatatan data. (Lin and Liao 2017)

<i>Permissionless Blockchain</i>	Partisipasi <i>node</i>	Jenis <i>blockchain</i> ini tidak memiliki protokol perizinan yang harus dipenuhi oleh suatu pihak untuk berpartisipasi sebagai <i>node</i> . (Rennock, Cohn and Butcher 2018)
<i>Permissioned Blockchain</i>	Partisipasi <i>node</i>	Jenis <i>blockchain</i> ini memiliki protokol perizinan yang harus dipenuhi oleh suatu pihak untuk dapat berpartisipasi sebagai <i>node</i> . (Rennock, Cohn and Butcher 2018)
<i>Stateless Blockchain</i>	Fungsionalitas	Jenis <i>blockchain</i> ini hanya dapat menjalankan logika komputasi sederhana saja seperti pencatatan data transaksi. (Hileman and Rauchs 2017)
<i>Stateful Blockchain</i>	Fungsionalitas	Jenis <i>blockchain</i> ini dapat menjalankan logika komputasi yang lebih kompleks daripada hanya sekedar komputasi pencatatan data. Contoh komputasi kompleks ini seperti

		pemrosesan <i>state</i> berdasarkan logika bisnis yang ada dalam suatu sistem. (Hileman and Rauchs 2017)
--	--	----------------------------------------------------------------------------------------------------------

Dengan penerapan *framework blockchain* pada suatu sistem, berbagai keuntungan dapat tercapai (Sarmah 2018). Keuntungan-keuntungan yang didapatkan antara lain sebagai berikut :

- a. Sistem akan bersifat terdesentralisasi, dimana pengelolaan sistem dilakukan tanpa adanya otoritas yang terpusat.
- b. Pengguna akan memiliki wewenang untuk berpartisipasi dalam melakukan pengelolaan data yang ada.
- c. Data dapat tersedia secara konsisten, lengkap, dan terkini karena tersebar pada setiap *node* yang tergabung dalam jaringan sistem.
- d. Dikarenakan tidak adanya wewenang yang terpusat, pengguna dapat yakin bahwa data akan dieksekusi berdasarkan mekanisme dari protokol yang ada.
- e. *Blockchain* memberikan kekekalan pada setiap datanya sehingga data yang telah tercatat tidak dapat termanipulasi.
- f. Data yang tercatat dapat terlindungi karena terenkripsi menggunakan mekanisme dari protokol *blockchain*.
- g. Sistem memiliki kekebalan pada berbagai jenis serangan (*cyber attack*) karena dibangun menggunakan jaringan *peer-to-peer*. Jaringan sistem

dapat beroperasi secara normal walaupun terdapat *node* yang tidak aktif akibat dari suatu serangan.

Dengan adanya arsitektur yang sangat kompleks dan integrasi berbagai macam teknologi yang diterapkan sehingga dapat menimbulkan bermacam-macam keuntungan, teknologi *blockchain* memiliki bidang penerapan yang luas (Gao, Hatcher and Yu 2018). Penerapan-penerapan ini dapat dilihat pada Tabel 2.2.

Tabel 2. 2 Bidang Aplikasi Blockchain

Bidang Aplikasi	Penjelasan
<i>Internet of Things (IoT)</i>	Sistem <i>IoT</i> dengan skala yang besar dapat dikembangkan secara terdesentralisasi dengan jaringan <i>peer-to-peer</i> menggunakan <i>blockchain</i> . Sifat terdesentralisasi dan dengan adanya protokol <i>blockchain</i> yang diterapkan dapat memberikan proteksi pada sistem <i>IoT</i> dari serangan yang umumnya terjadi pada sistem yang terpusat.
<i>Big Data</i>	<i>Blockchain</i> dianggap mampu memberikan solusi pada permasalahan manajemen data dalam sistem <i>big data</i> seperti proteksi data personal ataupun properti digital. Keandalan dan keamanan yang lebih dalam proses pencatatan data merupakan manfaat yang dapat diberikan

	oleh <i>blockchain</i> pada sistem <i>big data</i> .
<i>Cloud & Edge Computing</i>	Proteksi data merupakan alasan utama penerapan <i>blockchain</i> dalam sistem <i>cloud</i> ataupun <i>edge computing</i> . Data yang tercatat menggunakan mekanisme <i>blockchain</i> tidak dapat termanipulasi sehingga kredibilitas data dalam sistem <i>cloud/edge computing</i> dapat terjaga.
Manajemen Identitas	Dalam bidang ini, <i>blockchain</i> memberikan kemampuan dalam membentuk suatu identitas menggunakan <i>digital signature</i> . <i>Digital signature</i> akan digunakan sebagai tanda untuk memverifikasi kebenaran suatu identitas.
<i>Finansial & Cryptocurrency</i>	Teknologi <i>blockchain</i> awalnya dikembangkan pada bidang finansial. <i>Blockchain</i> memberikan kemungkinan dalam membentuk suatu sistem transaksi yang terdesentralisasi, dimana kebutuhan akan pihak ketiga sebagai pengelola transaksi dapat dihilangkan.
<i>Smart Contract & Otomatisasi</i>	<i>Smart contract</i> pada dasarnya merupakan kode-kode yang dibentuk untuk menjalankan suatu logika bisnis. <i>Smart contract</i> dapat dikembangkan menggunakan <i>blockchain</i> sehingga logika bisnis yang ada dapat dijalankan

	secara otomatis dengan menggunakan jaringan yang terdesentralisasi.
Rantai Suplai	Pada rantai suplai, terdapat berbagai entitas tergabung dalam prosesnya sehingga permasalahan kepercayaan menjadi hal utama di dalamnya. <i>Blockchain</i> memberikan solusi dalam memastikan integritas data pada sistem rantai suplai.
Informasi Medis	Dikarenakan informasi medis bersifat pribadi dan sensitif, <i>blockchain</i> dapat diimplementasikan dalam mengelola aksesibilitas data sehingga pihak yang memiliki wewenang saja yang dapat melihat dan melakukan penyimpanan.
Komunikasi & Jaringan	<i>Blockchain</i> memberikan proteksi dalam proses komunikasi yang terjadi pada suatu jaringan menggunakan <i>digital signature</i> sebagai metode dalam melakukan verifikasi kebenaran identitas.

2.3. *Smart Contract*

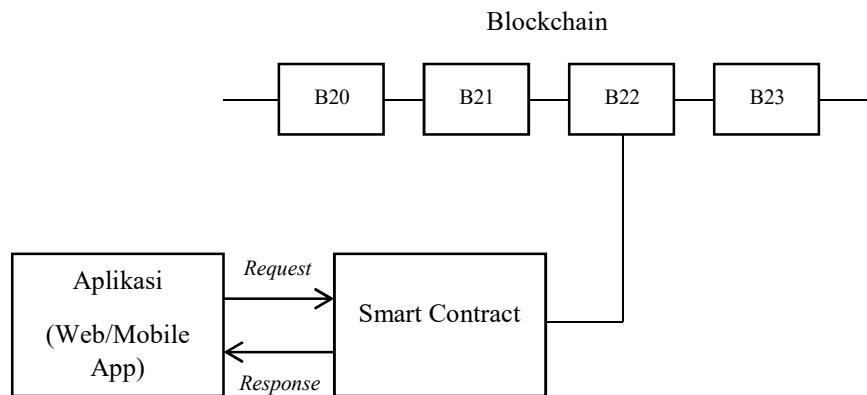
Kontrak pada dasarnya merupakan cara untuk membentuk kesepakatan persetujuan terhadap suatu hal (Szabo 1996). Kontrak secara umum digunakan untuk keperluan dalam membangun protokol persetujuan terhadap suatu hubungan yang dibentuk oleh dua pihak individu/kelompok atau lebih. Kontrak akan dibentuk oleh dua pihak atau lebih dengan menggunakan bantuan supervisi

dari pihak ketiga yang dianggap terpercaya. Supervisi ini sangat penting dimiliki untuk menghindari adanya manipulasi oleh salah satu pihak terhadap kontrak yang dibentuk.

Seiring dengan perkembangan teknologi, terbentuk suatu konsep kontrak baru yang dinamakan dengan *smart contract*. *Smart contract* pada dasarnya merupakan perangkat lunak berisi protokol kesepakatan dan hubungan antara dua pihak atau lebih yang dikelola menggunakan sistem terdesentralisasi. Pengawasan terhadap kesepakatan dan hubungan yang terbentuk akan dilakukan oleh semua pihak yang tergabung dalam jaringan berdasarkan protokol konsensus sistem sehingga kebutuhan supervisi dari suatu pihak ketiga tidak diperlukan.

Smart contract memungkinkan adanya pengembangan yang lebih pada teknologi *blockchain* karena kedua teknologi pada dasarnya diterapkan pada ekosistem yang sama, yaitu dibangun dengan menggunakan jaringan terdesentralisasi. *Blockchain* yang pada awalnya hanya digunakan untuk melakukan proses komputasi sederhana, seperti pencatatan data transaksi, telah dapat dikembangkan untuk melakukan proses komputasi yang lebih kompleks dengan integrasi *smart contract* (Jani 2020).

Kombinasi dari teknologi *blockchain* dan *smart contract* ini dinamakan *decentralized application*. Arsitektur *decentralized application* dapat dilihat pada Gambar 2.4.



Gambar 2. 4 Arsitektur *Decentralized Application*

Decentralized application, bekerja dengan tiga komponen utama, yaitu aplikasi klien, *smart contract*, dan *blockchain*. Aplikasi klien dapat berupa aplikasi *web* ataupun *mobile* yang memiliki kemampuan untuk berkomunikasi dengan *smart contract* melalui *application programming interface* (API). *Smart contract* sendiri akan bekerja layaknya seperti aplikasi *server-side (back-end)*. *Smart contract* tersimpan pada *blockchain* yang tersebar di setiap *node* yang tergabung dalam jaringan terdesentralisasi. *Smart contract* akan dieksekusi berdasarkan permintaan yang dikirim oleh aplikasi klien dan hasilnya akan dikembalikan dalam bentuk respon data. (Sayeed, Marco-Gisbert and Caira 2020)

2.4. Ethereum

Ethereum merupakan salah satu bentuk pengembangan dari teknologi *blockchain* yang awalnya diterapkan pada Bitcoin. Ethereum dikembangkan untuk memungkinkan penggerjaan komputasi yang lebih kompleks pada *framework blockchain* daripada hanya sekedar komputasi pencatatan data transaksi.

Sama halnya dengan Bitcoin, Ethereum pada dasarnya merupakan sistem pembayaran mata uang digital (*cryptocurrency*) yang terdesentralisasi. Perbedaan

utama yang ada pada Ethereum yaitu sistemnya dibangun dengan menggunakan bahasa pemrograman *turing-complete* (Ethereum Community 2020) sehingga memungkinkan pengeraan komputasi yang lebih kompleks, seperti *smart contract*, dilakukan dengan mekanisme *blockchain*. Ethereum telah dikenal luas sebagai *framework* dalam mengembangkan *decentralized application*.

Blockchain Ethereum pada dasarnya merupakan *state machine* berbasis transaksi. *State machine* sendiri mengacu pada proses pengelolaan suatu susunan *input* untuk mengubah *state* yang tersimpan. *State machine* pada Ethereum disebut sebagai Ethereum Virtual Machine (EVM). Pengubahan suatu *state* dilakukan oleh suatu *node* dengan mengirim transaksi yang berisi *input* untuk melakukan proses pengubahan *state*. (Kasireddy 2017)

State pada Ethereum merepresentasikan seluruh transaksi yang terjadi. Sama halnya dengan Bitcoin, transaksi-transaksi ini tergabung pada suatu *block*, dimana setiap *block* ini saling terhubung dengan *block* yang telah terbentuk sebelumnya. *Block* akan dibentuk dengan menggunakan protokol konsensus yang dinamakan GHOST (*Greedy Heavies Observed Subtree*). (Kasireddy 2017)

Protokol GHOST pada dasarnya merupakan protokol *Proof of Work*, tetapi dengan beberapa penyempurnaan. Protokol GHOST menyelesaikan permasalahan terkait *stale block*, yaitu *block* lain yang terbentuk secara bersamaan dengan *block* yang tervalidasi. Dalam protokol GHOST, *miner* tetap akan menerima hadiah apabila membentuk *stale block*. Hal tersebut diterapkan karena pembentukan *block* pada Ethereum tergolong lebih cepat jika dibandingkan dengan Bitcoin.

Sebagai akibatnya, kemungkinan terbentuknya suatu *stale block* pada Ethereum lebih besar dibandingkan pada Bitcoin.

2.4.1. *Account*

Account merupakan tanda pengenal (identitas) dari suatu entitas, seperti pengguna, *node*, ataupun *smart contract*, yang tergabung dalam Ethereum. *Account* pada Ethereum terbagi menjadi dua jenis, yaitu *externally owned account* dan *contract account* (Kasireddy 2017). *Externally owned account* pada dasarnya merupakan akun yang dimiliki oleh pengguna ataupun *node*. Sementara *contract account* merupakan tanda pengenal dari suatu *smart contract* yang tercatat dalam Ethereum. Pada dasarnya, hanya *externally owned account* yang dapat memulai mengirim pesan ke *account* lainnya dengan cara membuat dan menandatangani transaksi menggunakan *digital signature* berupa *private key*, sementara *contract account* hanya bisa melakukan transaksi sebagai bentuk respon apabila telah menerima suatu transaksi dari *account* lain.

Account merupakan obyek yang membentuk *state* pada Ethereum. Setiap *account* memiliki alamat (*address*) dan terdiri dari empat komponen *state* (Wood 2019), berupa :

- *nonce*, merupakan nilai yang merepresentasikan jumlah transaksi yang telah dilakukan oleh suatu *account*.
- *balance*, merupakan jumlah *cryptocurrency* Ethereum (Ether) yang dimiliki oleh suatu *account*.
- *storageRoot*, merupakan *root hash* dari konten-konten milik suatu *account* yang tersimpan. Secara *default*, *storageRoot* memiliki nilai kosong.

- *codeHash*, merupakan *hash* dari kode yang dimiliki oleh suatu *account* yang akan dieksekusi oleh Ethereum Virtual Machine (EVM). Pada *contract account*, *codeHash* merupakan *hash* dari kode *smart contract* yang dibentuk. Sementara pada *externally owned account*, *codeHash* merupakan *hash* dari *string* yang bernilai kosong.

2.4.2. Transaksi

Pada dasarnya, transaksi merupakan suatu instruksi yang dihasilkan oleh *externally owned account*. Transaksi akan dihasilkan dengan menggunakan *digital signature* yang berbasis *public/private key* milik *externally owned account*. Setiap transaksi yang dihasilkan akan tercatat ke dalam *blockchain* Ethereum.

Transaksi dalam Ethereum terbagi menjadi dua jenis, yaitu *message call* dan *contract creation* (Kasireddy 2017). *Message call* merupakan transaksi yang dilakukan dalam melakukan suatu perubahan *state*. *Contract creation* merupakan transaksi yang dilakukan untuk membentuk *contract* baru.

Setiap proses komputasi yang dibutuhkan untuk memproses suatu transaksi dalam Ethereum akan dieksekusi oleh *node miner*. *Miner* akan membutuhkan sebuah bayaran sebagai bentuk hadiah dalam mengeksekusi proses komputasi dari suatu transaksi. Biaya pemrosesan ini dibayar oleh *account* selaku pengirim transaksi berdasarkan unit khusus dalam Ethereum yang dinamakan *gas* (Kasireddy 2017). *Gas* pada dasarnya merupakan satuan unit yang digunakan untuk mengukur besar komputasi yang dibutuhkan dari suatu transaksi.

Penentuan biaya transaksi akan diukur dengan menggunakan dua variabel yaitu *gasPrice* dan *gasLimit*. *gasPrice* merupakan jumlah Ether yang pengirim

transaksi bersedia bayar untuk penggunaan satu *gas* dalam proses komputasi transaksi. Sementara *gasLimit* merupakan jumlah maksimum *gas* yang akan dipakai dalam proses komputasi transaksi. Kedua variabel ini akan ditentukan oleh pengirim transaksi. Sebagai contoh, pengirim transaksi menentukan *gasLimit* sebesar 50000 dan *gasPrice* sebesar 20 Gwei (1 Ether = 1000000000 Gwei) sehingga menghasilkan biaya transaksi sebesar 50000×20 Gwei = 100000 Gwei = 0.0001 Ether.

Adapun komponen-komponen yang terdapat dalam transaksi dapat dilihat pada Tabel 2.3.

Tabel 2.3 Komponen Transaksi (Kasireddy 2017)

Komponen	Penjelasan
<i>nonce</i>	Nomor transaksi yang ditentukan berdasarkan jumlah transaksi yang telah dilakukan oleh <i>account</i> pengirim transaksi.
<i>gasPrice</i>	Jumlah Ether yang ditentukan oleh pengirim transaksi untuk membayar setiap unit <i>gas</i> yang digunakan dalam proses eksekusi transaksi.
<i>gasLimit</i>	Jumlah maksimum unit <i>gas</i> yang dapat digunakan dalam proses eksekusi transaksi. <i>gasLimit</i> akan ditentukan oleh pengirim transaksi.
<i>to</i>	<i>Address</i> dari <i>account</i> penerima transaksi.

<i>value</i>	Jumlah Ether yang akan ditransfer dari pengirim transaksi ke penerima transaksi.
<i>v, r, s</i>	Digunakan untuk menghasilkan <i>digital signature</i> sebagai pengidentifikasi pengirim transaksi.
<i>init</i>	Sebuah bagian kode Ethereum Virtual Machine (EVM) yang digunakan untuk menginisialisasi <i>contract account</i> baru. Nilai ini hanya ada pada transaksi jenis <i>contract creation</i> .
<i>data</i>	<i>Input</i> yang akan digunakan untuk mengubah suatu <i>state</i> pada Ethereum. Nilai ini hanya ada pada transaksi jenis <i>message call</i> .

2.4.3. *Block*

Block pada Ethereum terdiri dari tiga bagian, yaitu *block header*, informasi transaksi-transaksi yang tergabung dalam *block*, dan kumpulan *block header* dari *ommer*, yaitu *stale block* yang terbentuk secara bersamaan dengan *block* yang tervalidasi. (Kasireddy 2017)

Pada Ethereum, *ommer* dipertimbangkan sebagai salah satu penunjang mekanisme *blockchain* agar dapat berjalan dengan baik. Hal tersebut dikarenakan Ethereum memiliki waktu pembentukan *block* yang lebih cepat (15 detik) dibanding *blockchain* lain seperti Bitcoin (10 menit). Semakin cepat waktu

pembentukan *block* pada suatu *blockchain* maka semakin besar juga persaingan yang ada dalam proses pembentukan *block*.

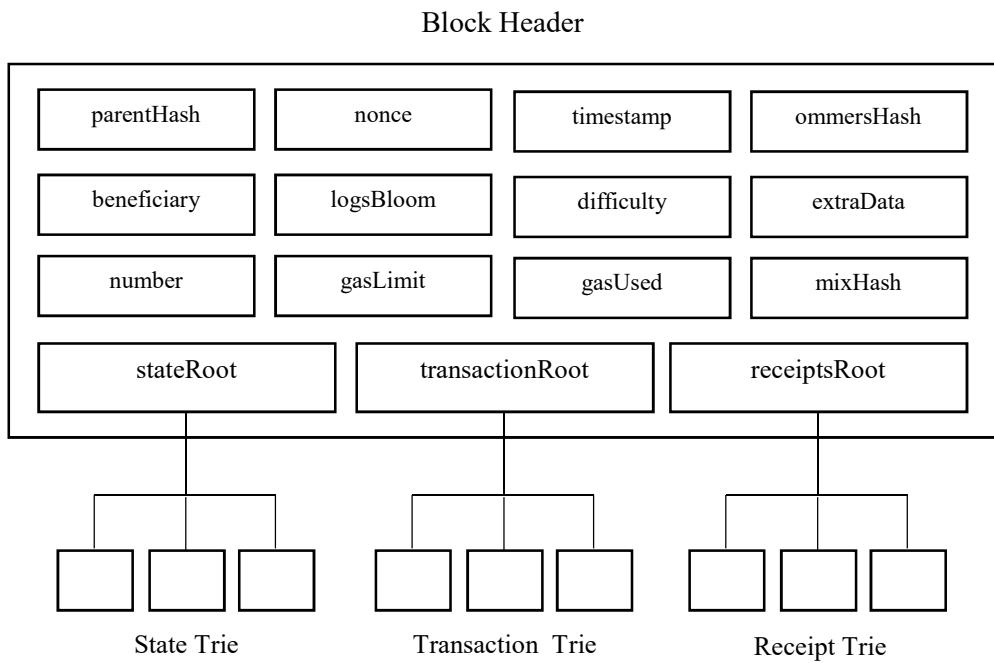
Protokol *blockchain* pada umumnya hanya memberikan hadiah kepada *miner* yang melakukan pembentukan *block* yang tervalidasi. Pada protokol yang diterapkan Ethereum, *miner* yang membentuk *ommer* juga akan mendapatkan hadiah, walaupun jumlahnya tidak sebanyak hadiah *block* tervalidasi. Dengan demikian, insentif *miner* dapat bertambah dalam melakukan pembentukan *block*.

Block dalam Ethereum pada dasarnya mirip seperti *block* dalam Bitcoin, hanya saja berisi tambahan informasi khusus berdasarkan protokol yang ditetapkan. Setiap informasi yang ada akan tergabung dan membentuk *block header*. Informasi-informasi yang terdapat pada *block header* dari suatu *block* yaitu (Wood 2019) :

- *parentHash* : *hash* dari *block header* milik *parent block*, yaitu *block* terakhir sebelum *block* terkait.
- *ommersHash* : *hash* dari daftar *ommer block* terkait.
- *beneficiary* : *address* dari *account* yang menerima biaya pembayaran dalam proses *mining block* terkait.
- *stateRoot* : *hash* dari *state* yang tersimpan pada *block* terkait. *Hash* ini terbentuk menggunakan struktur Merkle Patricia Trie.
- *transactionRoot* : *hash* dari transaksi yang tercatat pada *block* terkait. *Hash* ini terbentuk menggunakan struktur Merkle Patricia Trie.
- *receiptsRoot* : *hash* dari resi transaksi yang tercatat pada *block* terkait. *Hash* ini terbentuk menggunakan struktur Merkle Patricia Trie.

- *logsBloom* : struktur data *bloom filter* yang terdiri atas *log/catatan* informasi.
- *difficulty* : tingkat *difficulty* dari penyelesaian *block* terkait.
- *number* : nilai penjumlahan (*count*) dari *block* terkait.
- *gasLimit* : batas maksimum *gas* yang dapat digunakan dalam pemrosesan komputasi transaksi pada *block* terkait.
- *gasUsed* : jumlah total *gas* yang digunakan pada pemrosesan komputasi transaksi dari *block* terkait.
- *timestamp* : *timestamp* yang berbasis unix dari pembentukan *block* terkait.
- *extraData* : data ekstra yang berhubungan dengan *block* terkait.
- *mixHash* : sebuah *hash*, yang jika dikombinasikan dengan *nonce*, akan membuktikan bahwa *block* terkait telah dibentuk melalui proses komputasi yang cukup.
- *nonce* : sebuah *hash*, yang jika dikombinasikan dengan *mixHash*, akan membuktikan bahwa *block* terkait telah dibentuk melalui proses komputasi yang cukup.

Ilustrasi *block header* dari suatu *block* pada Ethereum dapat dilihat pada Gambar 2.5.



Gambar 2. 5 Block Header

2.4.4. Eksekusi Transaksi

Eksekusi transaksi dalam Ethereum pada dasarnya mengacu pada proses transisi *state* yang ada (Wood 2019). Sebelum transaksi dieksekusi, terdapat beberapa syarat yang harus dipenuhi terlebih dahulu, yaitu :

- Transaksi harus memiliki format yang sesuai. Format *Recursive Length Prefix* (RLP) merupakan format data yang diterima oleh Ethereum.
- *Signature* transaksi yang valid.
- *Nonce* transaksi yang valid.
- *gasLimit* transaksi harus memiliki jumlah yang lebih besar atau sama dengan jumlah *gas* yang terpakai dalam eksekusi komputasi transaksi.

- Jumlah Ether yang dimiliki oleh pengirim transaksi dapat menutupi biaya pemakaian *gas* yang terbentuk berdasarkan perhitungan *gasLimit* dan *gasPrice* yang ditentukan.

Apabila syarat-syarat yang diperlukan telah terpenuhi, selanjutnya eksekusi transaksi dapat dilakukan. Eksekusi transaksi secara umum akan dilakukan melalui tahap-tahap sebagai berikut :

1. Biaya transaksi akan diukur berdasarkan *gasLimit* dan *gasPrice* yang ditentukan. Ether pengirim transaksi akan terpotong berdasarkan hasil perhitungan biaya transaksi tersebut.
2. Menginisialisasi jumlah *gas* yang akan digunakan selama proses komputasi transaksi.
3. Komputasi-komputasi yang diperlukan akan dieksekusi pada Ethereum Virtual Machine (EVM) oleh *miner*.
4. Ketika komputasi yang diperlukan oleh transaksi telah terproses, maka biaya yang terbayar untuk sisa *gas* yang tidak terpakai akan dikembalikan ke pengirim transaksi. Disaat yang bersamaan, biaya *gas* yang terpakai akan dikirim ke *miner*.
5. *State* baru dan *logs* yang berisi catatan transaksi yang terjadi akan terbentuk.

2.4.5. Arsitektur *Blockchain*

Ethereum memiliki arsitektur *blockchain* yang hampir sama dengan *blockchain* Bitcoin. Perbedaannya terletak pada informasi yang tersimpan pada *block* dan algoritma yang dipakai pada pembentukan *Proof of Work*. Pada *block*

Ethereum, selain daripada transaksi, *state* terbaru juga akan disimpan. Sementara itu, algoritma *Proof of Work* yang digunakan pada Ethereum dinamakan *Ethash*.

Algoritma *Ethash* secara formal didefinisikan pada Persamaan 2.1.

$$(m, n) = \text{PoW}(HN, Hn, d) \quad (2.1)$$

Pada Persamaan 2.1, **m** merupakan *mixHash*, **n** adalah *nonce*, **HN** diartikan sebagai *block header* dari *block* baru yang terbentuk (tanpa komponen *nonce* dan *mixHash*), **Hn** sebagai *nonce* dari *block header*, serta **d** yaitu DAG (*data set* besar). (Wood 2019)

Algoritma *Ethash* pada dasarnya akan mencari nilai *mixHash* dan *nonce* yang sesuai berdasarkan *difficulty* yang ditetapkan pada proses pembentukan *block*. Kedua nilai ini merupakan bukti bahwa suatu pembentukan *block* telah melalui proses komputasi yang besar (*Proof of Work*).

BAB III

METODOLOGI PENELITIAN

3.1. Waktu dan Lokasi Penelitian

Pelaksanaan Tugas Akhir ini dilakukan dalam waktu sekitar 12 bulan, dimulai sejak disetujuinya proposal Tugas Akhir pada bulan September 2019 hingga proses pelaporan Tugas Akhir pada bulan September 2020. Lokasi pelaksanaan Tugas Akhir dilakukan di Laboratorium Ubiquitous Computing & Networking Departemen Teknik Informatika, Universitas Hasanuddin.

3.2. Instrumen Penelitian

Instrumen-instrumen yang digunakan dalam pelaksanaan kegiatan penelitian ini yaitu :

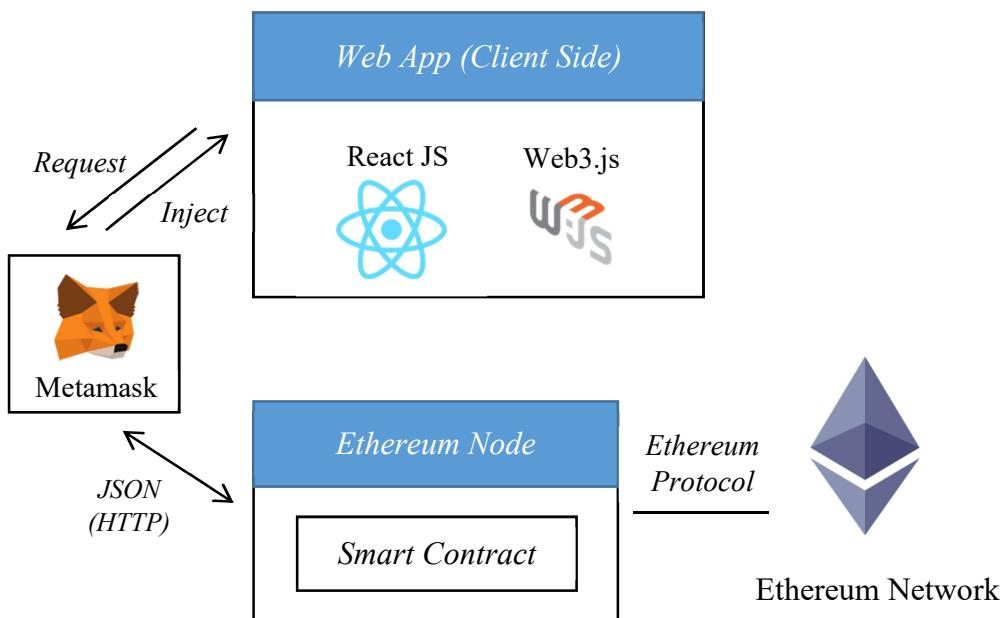
- *Software* :
 - a. Sistem operasi Windows 10 Pro, yaitu sistem operasi yang digunakan untuk menjalankan aplikasi sisi klien sistem.
 - b. *Web browser* Google Chrome, yaitu *web browser* yang digunakan untuk menjalankan aplikasi sisi klien sistem.
 - c. Visual Studio Code, yaitu *code editor* yang digunakan untuk mengembangkan aplikasi sisi klien dan *smart contract* sistem.
 - d. Metamask, yaitu *browser extension* yang digunakan sebagai *crypto wallet* Ethereum dan sekaligus sebagai *gateway* untuk menghubungkan aplikasi sisi klien ke jaringan Ethereum.

- e. Ganache, yaitu *personal blockchain* Ethereum yang digunakan dalam masa pengembangan sistem untuk mengetes kesesuaian fungsionalitas *smart contract* sistem.
 - f. Truffle Framework, yaitu *framework* yang digunakan untuk mengembangkan *smart contract* sistem.
 - g. Remix Ethereum IDE, yaitu *Integrated Development Environment* (IDE) untuk mengembangkan *smart contract* sistem.
 - h. Node Package Manager (NPM), yaitu *package manager* yang digunakan untuk mengelola *package* yang dibutuhkan aplikasi sisi klien dan *smart contract* sistem.
 - i. Node.js, yaitu *JavaScript environment* yang digunakan untuk menjalankan NPM.
 - j. ReactJS, yaitu *JavaScript library* yang digunakan untuk mengembangkan aplikasi sisi klien sistem.
 - k. Web3.js, yaitu *library* yang digunakan pada aplikasi sisi klien sistem untuk menjalankan berbagai fungsi spesifik yang ada pada ekosistem Ethereum.
 - l. Go-Ethereum (geth), yaitu merupakan *command line interface* yang digunakan untuk menjalankan *node* Ethereum.
- *Hardware* berupa *Personal Computer* Sony VAIO VPCSB38GG, CPU Intel Core i7-2640M @ 2.80GHz (4 CPUs), 8192MB RAM, Windows 10 Pro

3.3. Gambaran Umum Sistem

Sistem *crowdfunding* dibangun dengan implementasi *blockchain* dan integrasi *smart contract*. Platform *blockchain* yang digunakan yaitu Ethereum. *Smart contract* ini akan diimplementasikan sebagai aplikasi sisi *server* yang berisi logika bisnis untuk seluruh aktivitas yang terjadi pada sistem *crowdfunding*. Sementara itu, aplikasi berbasis *web* digunakan sebagai aplikasi sisi *klien* untuk proses interaksi langsung antara pengguna dan sistem.

Sistem ini dibangun sebagai *sistem crowdfunding* baru dengan fitur yang lengkap dan memiliki kelebihan dari segi keamanan dan transparansi serta biaya pemrosesan yang lebih optimal dibanding sistem *crowdfunding* yang ada saat ini.

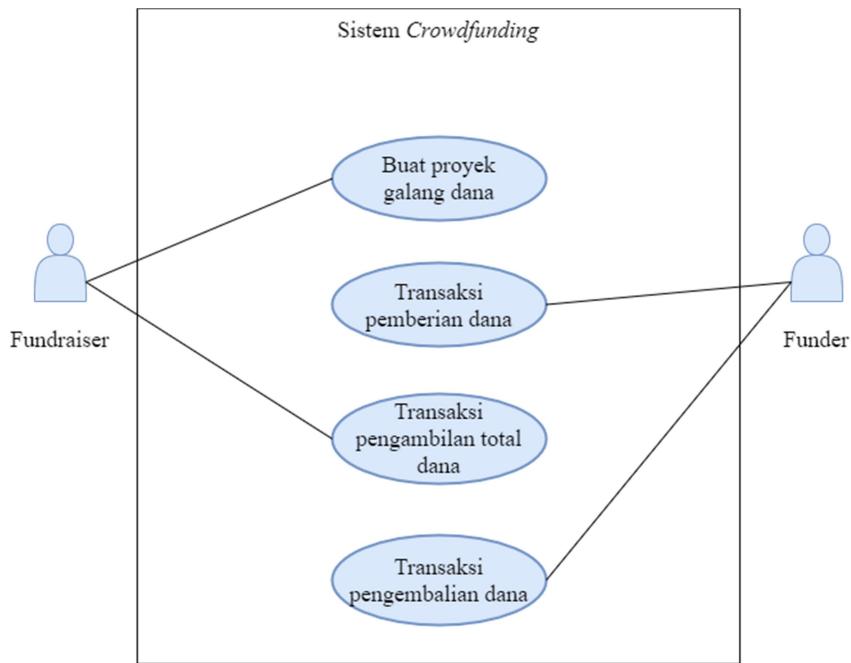


Gambar 3. 1 Gambaran Umum Sistem

Gambaran umum sistem *crowdfunding* yang dibangun dalam studi ini dapat dilihat pada Gambar 3.1. Aplikasi *web* dibentuk sebagai aplikasi dari sisi

klien dimana pengguna berinteraksi secara langsung, dengan logika bisnis yang ditempatkan pada *smart contract*.

Sistem *crowdfunding* pada studi ini akan dibangun dengan dua jenis pengguna, yaitu *fundraiser* sebagai pembentuk proyek galang dana dan *funder* sebagai pemberi dana. *Use case* sistem dapat dilihat pada Gambar 3.2.



Gambar 3. 2 Use Case Diagram Sistem

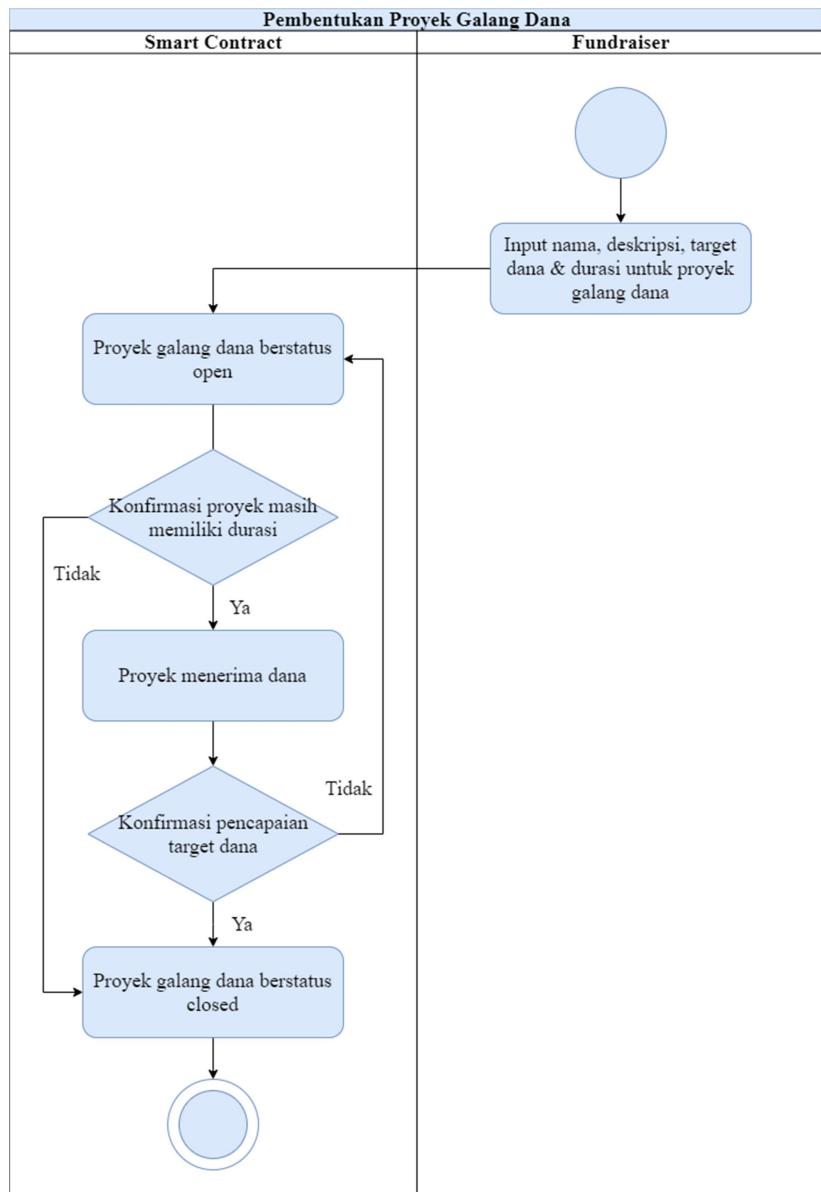
3.3.1. *Activity Diagram* Sistem

Sistem *crowdfunding* pada studi ini dibangun berdasarkan gambaran *activity diagram* untuk masing-masing *use case* sistem. *Activity diagram* pertama untuk *use case* proses pembentukan proyek galang dana dengan interaksi antara *fundraiser* dan *smart contract*. *Activity diagram* kedua untuk *use case* proses pemberian dana pada proyek galang dana dengan interaksi antara *funder* dan *smart contract*. *Activity diagram* ketiga untuk *use case* proses pengambilan total dana proyek galang dana ketika telah mencapai target dana yang ditetapkan

dengan interaksi antara *fundraiser* dan *smart contract*. *Activity diagram* keempat untuk *use case* proses pengembalian dana pada proyek galang dana yang tidak mencapai target dengan interaksi antara *funder* dan *smart contract*.

a. Pembentukan Proyek

Gambar 3.3 merupakan *Activity Diagram* untuk proses pembentukan proyek galang dana, dimana pembentukan proyek dilakukan oleh *fundraiser* dengan mengisi sejumlah data yang kemudian akan diterima oleh *smart contract* untuk diproses dan dikelola berdasarkan logika bisnis yang ditetapkan.



Gambar 3. 3 *Activity Diagram* Pembentukan Proyek

Adapun penjelasan *input* & *output* yang ada pada Gambar 3.3 terkait proses pembentukan proyek galang dana dapat dilihat pada Tabel 3.1.

Tabel 3. 1 *Input* dan *Output* Pembentukan Proyek

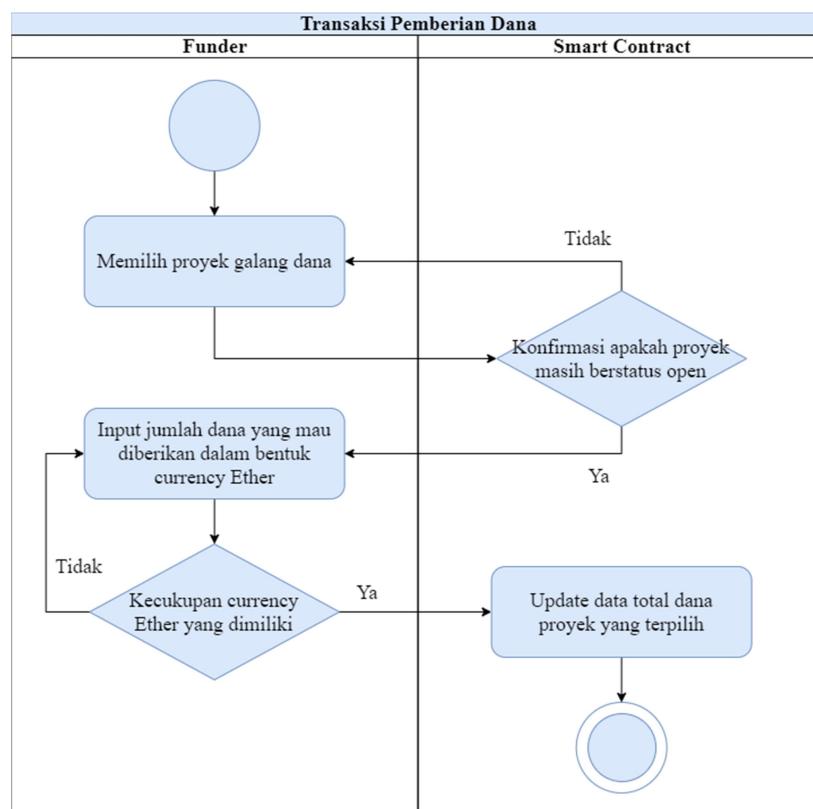
No	Aksi	<i>Input</i>	<i>Output</i>
1.	<i>Fundraiser</i> memberikan	<i>Input</i> berupa nama	Data proyek galang

	<p><i>input</i> data terkait informasi proyek galang dana yang akan dibentuk berupa nama, deskripsi, target dana, dan durasi. <i>Input</i> ini kemudian akan dikirim sebagai data proyek galang dana baru melalui transaksi yang dilakukan oleh <i>fundraiser</i> untuk mengubah <i>state</i> pada <i>smart contract</i> sehingga data proyek galang dana dapat tercatat.</p>	<p>(text), deskripsi (text), target dana (number), dan durasi (number).</p>	<p>dana baru tercatat berdasarkan <i>input</i> yang diterima dan logika bisnis yang berjalan dalam <i>smart contract</i>.</p>
2.	<p><i>Smart Contract</i> proyek galang dana menerima dana dan total dana bertambah. Total dana ini selanjutnya akan diproses untuk penentuan pencapaian target dana.</p>	<p><i>State</i> total dana (<i>default</i> bernilai 0) yang bertambah berdasarkan jumlah dana yang diterima.</p>	<p>Kasus pertama, ketika target dana tercapai, <i>state</i> status proyek galang dana berubah menjadi <i>close</i>. Kasus kedua, ketika target dana belum tercapai, <i>state</i> status</p>

			proyek galang dana masih <i>open</i> .
--	--	--	-------------------------------------------

b. Pemberian Dana pada Proyek

Gambar 3.4 merupakan *Activity Diagram* untuk proses pemberian dana pada proyek galang dana yang telah terbentuk, dimana pemberian dana dilakukan oleh *funder* dengan mengisi *input* jumlah dana yang akan diberikan dan kemudian akan diterima oleh *smart contract* untuk diproses sebagai total dana yang diterima berdasarkan logika bisnis yang ditetapkan.



Gambar 3. 4 *Activity Diagram* Pemberian Dana pada Proyek

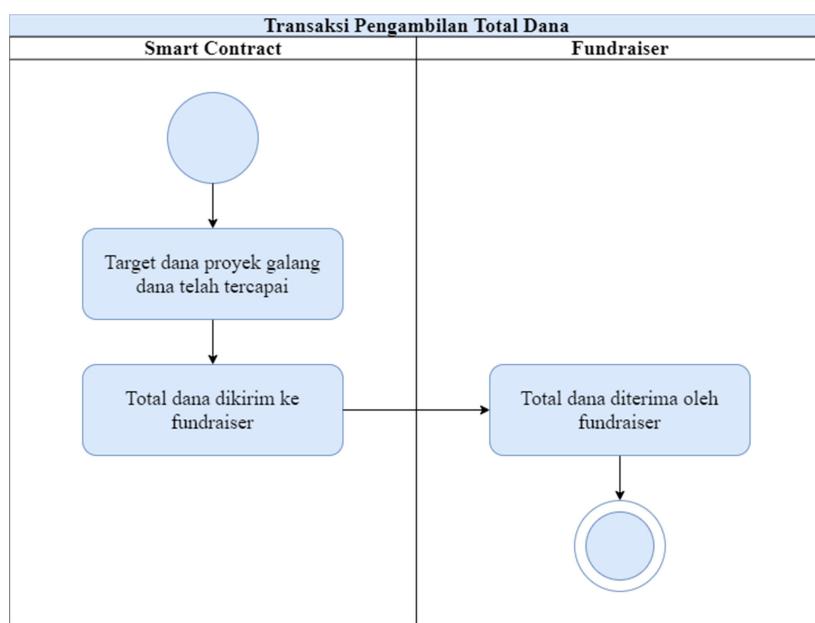
Adapun penjelasan terkait *input* dan *output* yang ada pada Gambar 3.4 terkait proses pemberian dana dapat dilihat pada Tabel 3.2.

Tabel 3. 2 *Input* dan *Output* Pemberian Dana pada Proyek

No	Aksi	Input	Output
1.	Pemilihan proyek galang dana oleh <i>funder</i> dilakukan dengan mengirim <i>request</i> untuk mendapatkan data detail status proyek galang dana.	<i>Input</i> untuk mengirim <i>request</i> data proyek galang dana berupa <i>button click</i> pada tombol detail proyek.	<i>Output</i> berupa respon berisi data detail status proyek galang dana.
2.	<i>Funder</i> memberikan <i>input</i> berupa jumlah dana yang akan diberikan dalam proses pemberian dana ke proyek galang. Input ini kemudian akan divalidasi jumlahnya berdasarkan saldo yang dimiliki oleh <i>funder</i> . Apabila telah tervalidasi, data hasil <i>input</i> tersebut kemudian dikirim dan menyebabkan perubahan total dana yang dimiliki proyek.	<i>Input</i> jumlah dana (<i>integer</i>) yang akan diberikan oleh <i>funder</i> .	<i>Output</i> berupa perubahan data total dana proyek galang dana pada <i>smart contract</i> ketika <i>input</i> telah berhasil melalui proses validasi. Apabila tidak berhasil, maka <i>funder</i> akan diminta kembali untuk memberikan <i>input</i> yang sesuai.

c. Pengambilan Dana Proyek

Proses pengambilan dana pada proyek galang dana akan dilakukan secara otomatis oleh *smart contract*. Ketika target dana proyek telah tercapai, *smart contract* akan mengirim total dana yang ada ke *account* milik *fundraiser*. Saldo *fundraiser* akan bertambah secara otomatis setelah proses pengiriman tersebut telah selesai. *Activity diagram* proses ini dapat dilihat pada Gambar 3.5.



Gambar 3. 5 *Activity Diagram* Pengambilan Dana Proyek

Adapun penjelasan terkait *input* dan *output* yang ada pada Gambar 3.5 terkait proses pengambilan dana proyek dapat dilihat pada Tabel 3.3.

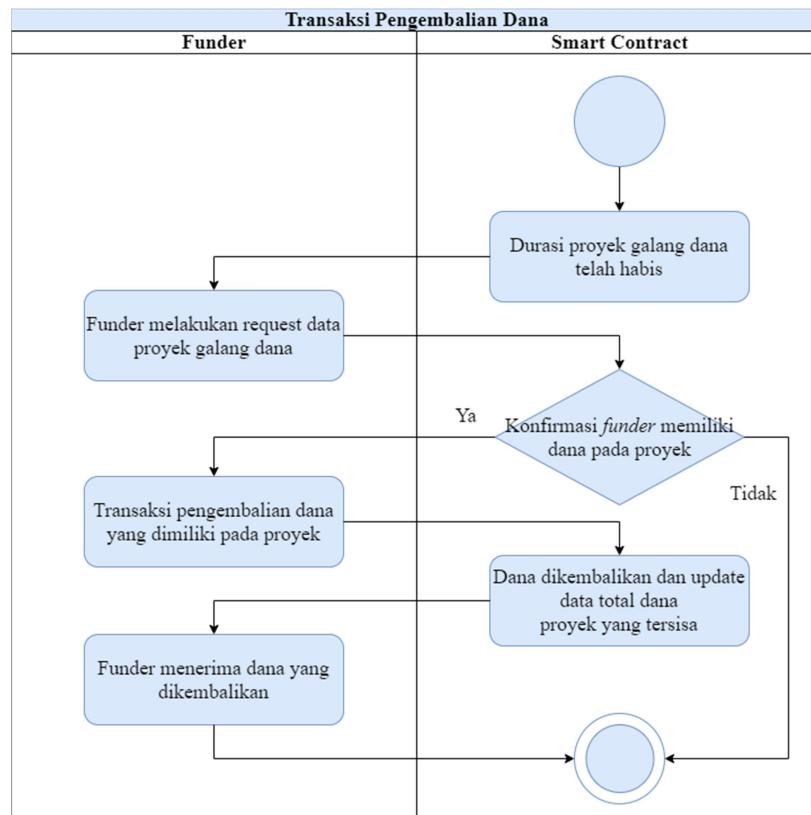
Tabel 3. 3 *Input* dan *Output* Pengambilan Dana Proyek

No	Aksi	Input	Output
1.	<i>Smart contract</i> akan melakukan pengiriman total dana yang ada pada	<i>State target dana yang berubah menjadi tercapai.</i>	Saldo <i>account</i> milik <i>fundraiser</i> bertambah

	proyek ke <i>account</i> milik <i>fundraiser</i> ketika target dana telah tercapai.		berdasarkan total dana yang terkirim.
--	-------------------------------------------------------------------------------------	--	---------------------------------------

d. Pengembalian Dana Proyek

Proses pengembalian dana pada proyek galang dana yang durasinya telah habis dilakukan melalui *request* transaksi oleh *funder* terhadap *smart contract*. Setelah *smart contract* mengonfirmasi bahwa terdapat dana yang dimiliki oleh *funder* pada proyek galang dana, maka *smart contract* akan menerima transaksi yang dilakukan oleh *funder* dan proses pengembalian dana berhasil dilakukan. *Activity diagram* proses ini dapat dilihat pada Gambar 3.6.



Gambar 3. 6 *Activity Diagram* Pengembalian Dana Proyek

Adapun penjelasan terkait *input* dan *output* yang ada pada Gambar 3.6 terkait proses pengembalian dana proyek dapat dilihat pada Tabel 3.4.

Tabel 3. 4 *Input* dan *Output* Pengembalian Dana Proyek

No	Aksi	Input	Output
1.	<i>Funder</i> mengirimkan <i>request</i> data terkait informasi proyek galang dana yang durasinya telah habis untuk mengetahui jumlah dana yang pernah diberikan pada proyek tersebut.	<i>Input request</i> data berupa <i>button click</i> tombol detail proyek.	<i>Output</i> berupa respon data dana yang dimiliki <i>funder</i> . Apabila ada, maka <i>funder</i> dapat melakukan proses pengembalian dana. Jika tidak, maka proses pengembalian dana tidak dapat dilakukan.
2.	<i>Funder</i> melakukan transaksi pada <i>smart contract</i> untuk mendapatkan pengembalian dana berdasarkan jumlah dana yang pernah diberikan pada	<i>Input</i> berupa <i>button click</i> untuk memulai pengembalian dana proyek.	<i>Output</i> berupa perubahan <i>state</i> total dana yang tersisa untuk dikembalikan pada <i>smart contract</i> dan saldo <i>account</i> milik

	proyek galang dana.		<i>funder</i> akan bertambah.
--	---------------------	--	-------------------------------

3.3.2. Desain *Smart Contract*

Pada sistem ini, *smart contract* dikembangkan sebagai *backend service* untuk mengelola logika bisnis yang ada pada sistem *crowdfunding* yang dibangun. *Smart Contract* yang dikembangkan terbagi menjadi dua jenis *smart contract*, yaitu *contract* “*crowdfunding*” dan *contract* “*project*”. *Contract crowdfunding* berisi *variable* dan *function* untuk data dan aksi umum yang dapat dilakukan pada sistem, seperti data jumlah proyek galang dana yang telah terbentuk maupun aksi pembentukan proyek galang dana baru. *Contract* ini akan bekerja sebagai wadah awal untuk inisialisasi *contract project*. Sementara itu, *contract project* berisi *variable* dan *function* untuk data dan aksi yang lebih detail dari suatu proyek galang dana yang ada, seperti data total dana yang ada ataupun aksi pemberian dana.

Variable dan *function* dari *contract crowdfunding* dapat dilihat pada Tabel 3.5.

Tabel 3.5 *Variable* dan *Function* dari Contract Crowdfunding

No.	<i>Variable / Function</i>	Jenis	Penjelasan
1.	<i>projects</i>	<i>Array (var)</i>	<i>Array</i> dari seluruh proyek galang dana yang terbentuk.
2.	<i>createProject</i>	<i>Function</i>	<i>Function</i> untuk membentuk proyek galang dana baru.

3.	<i>getProjects</i>	<i>Function</i>	<i>Function</i> untuk melihat seluruh proyek galang dana yang terbentuk.
----	--------------------	-----------------	--------------------------------------------------------------------------

Adapun *variable* dan *function* dari *contract project* dapat dilihat pada

Tabel 3.6.

Tabel 3. 6 *Variable* dan *Function* dari Contract Project

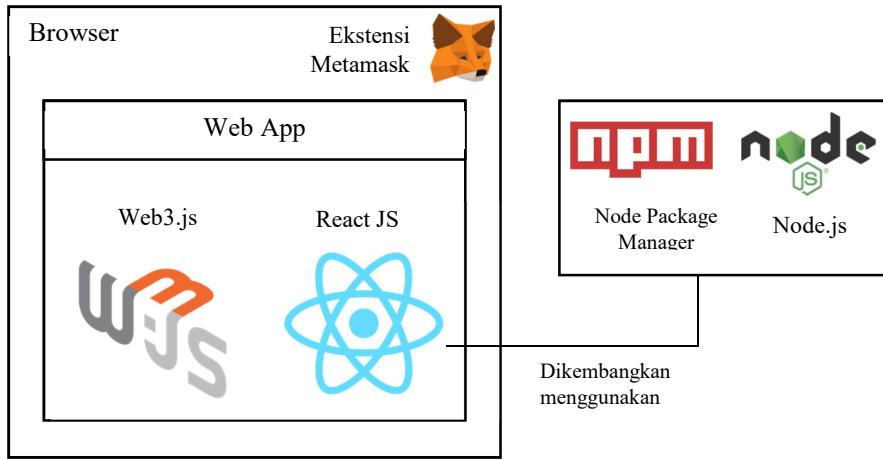
No.	<i>Variable / Function</i>	Jenis	Penjelasan
1.	<i>fundraiser</i>	<i>Address (var)</i>	<i>Address</i> dari <i>account fundraiser</i> proyek galang dana.
2.	<i>fundGoal</i>	<i>Integer (var)</i>	Target dana proyek galang dana.
3.	<i>currentFund</i>	<i>Integer (var)</i>	Total dana proyek galang dana.
4.	<i>duration</i>	<i>Integer (var)</i>	Durasi proyek galang dana.
5.	<i>title</i>	<i>String (var)</i>	Nama proyek galang dana.
6.	<i>description</i>	<i>String (var)</i>	Deksripsi proyek galang dana.
7.	<i>state</i>	<i>Boolean (var)</i>	Status proyek galang dana.
8.	<i>funders</i>	<i>Array (var)</i>	<i>Array</i> dari <i>funder</i> yang telah memberikan dana pada proyek.
9.	<i>funding</i>	<i>Function</i>	<i>Function</i> untuk melakukan pemberian dana.
10.	<i>giveFund</i>	<i>Function</i>	<i>Function</i> untuk melakukan pemberian total dana pada <i>fundraiser</i> jika target telah

			terpenuhi.
11.	<i>getRefund</i>	<i>Function</i>	<i>Function</i> untuk melakukan pengembalian dana ke <i>funder</i> jika durasi proyek telah habis dan target dana belum tercapai.
12.	<i>projectDetails</i>	<i>Function</i>	<i>Function</i> untuk melihat detail informasi dari proyek galang dana.

3.3.3. Detail Perancangan Sistem

Sistem *crowdfunding* pada studi ini terbagi menjadi 2 bagian, yaitu bagian aplikasi sisi klien berupa aplikasi *web* sebagai antarmuka sistem dan bagian *smart contract* sebagai aplikasi sisi server yang berisi logika bisnis sistem.

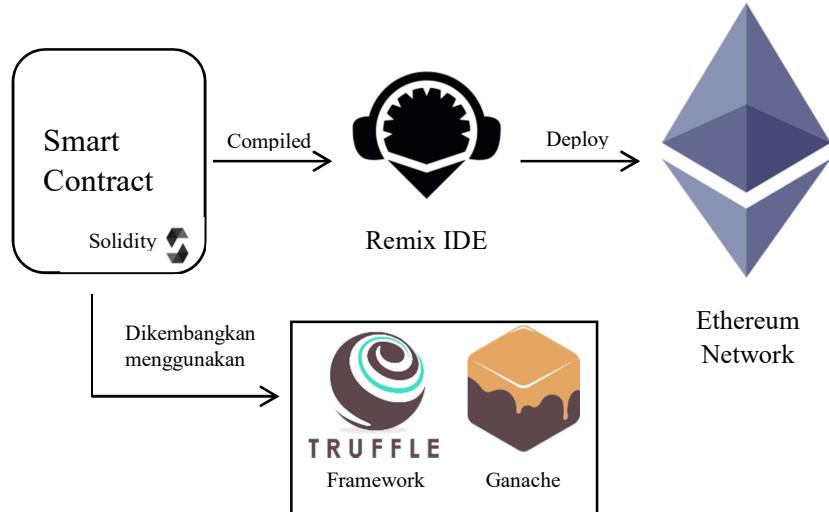
Aplikasi *web* pada sistem dibangun dengan menggunakan *library* ReactJS untuk seluruh komponen antarmuka-nya dan web3.js sebagai *application programming interface* (API) untuk hubungan ke *node Ethereum* dimana *smart contract* sistem terdapat. Aplikasi *web* ini kemudian akan dijalankan melalui *browser* dengan ekstensi yang dinamakan Metamask sehingga dapat terhubung ke jaringan Ethereum. Metamask ini juga digunakan sebagai *wallet Ethereum* untuk proses pengelolaan informasi dan proses transaksi *currency Ether* yang dimiliki oleh *account Ethereum* yang nantinya digunakan pada sistem *crowdfunding*.



Gambar 3.7 Pengembangan Aplikasi Web Sistem *Crowdfunding*

Pada Gambar 3.7 terlihat bahwa aplikasi *web* dikembangkan dengan menggunakan Node.js dan Node package manager (NPM). Node.js dan NPM digunakan untuk kebutuhan implementasi *library-library* untuk kebutuhan lain seperti *library* CSS Materialize UI untuk komponen UI, *library* react *router* untuk *routing* halaman-halaman aplikasi *web*, dan sebagainya.

Sementara itu, *smart contract* pada sistem *crowdfunding* dibangun dengan bahasa Solidity menggunakan *framework* Truffle dan Ganache sebagai *local blockchain environment* untuk tahap pengembangan. Setelah *smart contract* selesai dikembangkan, kemudian dilakukan proses *compile* dan *deploy smart contract* ke jaringan Ethereum menggunakan Remix Ethereum IDE. Proses pengembangan ini dapat dilihat pada Gambar 3.8.



Gambar 3. 8 Pengembangan *Smart Contract* Sistem *Crowdfunding*

3.4. Skenario Analisis dan Pengujian

Pada studi ini, sistem *crowdfunding* yang dibangun dengan implementasi *blockchain* dan integrasi *smart contract* selanjutnya akan dianalisis ataupun diuji berdasarkan beberapa aspek. Aspek ini berupa fungsionalitas sistem, keamanan dan transparansi sistem serta besar biaya transaksi untuk eksekusi pemrosesan yang ada pada sistem.

3.4.1. Pengujian Fungsionalitas Sistem

Pengujian fungsionalitas dilakukan untuk mengetahui apakah sistem *crowdfunding* yang dibangun dapat berjalan baik dan sesuai dengan *use case*-nya. Dalam proses pengujian masing-masing *use-case*, digunakan metode *black box testing*, yaitu pengujian fungsionalitas tanpa melihat struktur internal sistem. *Use case* sistem dapat dilihat pada *use case diagram* sistem Gambar 3.2 dan secara detail dapat dilihat pada *activity diagram* sistem Gambar 3.3, Gambar 3.4, Gambar 3.5 dan Gambar 3.6.

Sistem akan diuji berdasarkan *use case*-nya mulai dari *input* yang diberikan oleh pengguna ke dalam aplikasi sisi klien-nya, proses yang terjadi dalam sistem, sampai pada *output* yang akan terbentuk berupa perubahan *state* pada *smart contract* dan respon yang akan diterima oleh pengguna pada aplikasi sisi klien. Proses yang terjadi dalam sistem akan diuji secara detail, baik yang berjalan pada aplikasi sisi klien ataupun yang berjalan pada *smart contract*.

3.4.2. Analisis Keamanan & Transparansi Sistem

Keamanan dan transparansi sistem *crowdfunding* yang dibangun akan dianalisis berdasarkan arsitektur sistem secara keseluruhan, proses penyimpanan data dan keunggulan yang diberikan dengan implementasi *blockchain* pada sistem. Dasar analisis ini akan berkaitan dengan *platform* Ethereum yang digunakan sebagai ekosistem *blockchain* sistem.

3.4.3. Pengujian Besar Biaya Transaksi Sistem

Penggunaan *platform* Ethereum pada sistem *crowdfunding* yang dibangun memberikan kemungkinan dalam menerapkan pemrosesan logika bisnis yang ada (*smart contract*) dengan implementasi *blockchain* sebagai arsitektur pencatatan data di dalamnya.

Proses-proses yang terjadi berdasarkan logika bisnis yang ada dalam sistem akan dieksekusi oleh *node-node* yang tergabung ke dalam jaringan Ethereum, sebagai ganti dari otoritas utama yang umumnya melakukan eksekusi tersebut pada suatu sistem. Dikarenakan eksekusi ini dilakukan oleh *node-node* Ethereum, maka terdapat beban biaya yang ditetapkan oleh protokol Ethereum

untuk setiap proses komputasinya. Biaya ini nantinya akan diberikan sebagai hadiah kepada *node* yang melakukan eksekusi (*miner*).

Pembentukan biaya ini atas dasar jumlah pemakaian *gas*, yaitu suatu unit pada Ethereum yang digunakan untuk mengukur jumlah usaha komputasi yang dilakukan untuk menjalankan suatu pemrosesan. Biaya pemakaian gas ini pada dasarnya merupakan biaya transaksi. Biaya ini akan diberikan oleh seorang pengirim melalui suatu transaksi yang berisi pemrosesan yang ingin dieksekusi dan ditentukan berdasarkan beberapa parameter. Parameter yang digunakan yaitu :

- *gasPrice*, merupakan harga pemakaian 1 *gas* dalam Gwei ($0.000000001 / 1 \times 10^{-9}$ Ether) untuk melakukan suatu eksekusi transaksi. Parameter ini akan menentukan kecepatan dari eksekusi transaksi sehingga dapat tercatat ke dalam *block*. Semakin besar *gasPrice* maka akan semakin tinggi prioritas transaksi untuk dieksekusi oleh *miner*.
- *gasLimit*, merupakan batas maksimum penggunaan *gas* dalam melakukan eksekusi transaksi. Parameter ini dibentuk untuk menghindari adanya pemakaian *gas* secara terus menerus pada suatu transaksi.

Setelah parameter ini ditentukan, kemudian perhitungan biaya pemakaian *gas* akan dilakukan. Dalam proses perhitungan ini, terlebih dahulu akan dihitung *gasFee* (biaya *gas*) yang akan digunakan dengan mengalikan *gasPrice* dan *gasLimit* yang telah ditentukan oleh pengirim. *Miner* kemudian akan mengeksekusi transaksi ini dan menentukan *gasUsed* (jumlah pemakaian *gas*) yang digunakan dalam proses eksekusinya. Apabila *gasUsed* lebih besar dari *gasLimit* yang ditetapkan, maka proses eksekusi transaksi tidak akan selesai dan

dibatalkan. Jika *gasUsed* sama dengan atau kurang dari *gasLimit* yang ditetapkan, maka transaksi akan berhasil dieksekusi dan sebagian porsi *gasFee* yang telah dibayar akan dikembalikan ke pengirim sesuai dengan jumlah *gas* yang tidak terpakai. Dengan demikian, *transactionFee* (biaya transaksi) didapatkan melalui hasil perkalian *gasUsed* dan *gasPrice* sebagai biaya dari total *gas* yang terpakai dalam eksekusi transaksi.

Parameter utama yang akan mempengaruhi besar biaya transaksi dalam sistem yaitu *gasPrice*. Akan tetapi, nilai dari *gasPrice* tidak menentu. Parameter ini perlu ditentukan berdasarkan keadaan dari Ethereum pada saat transaksi akan dikirim untuk dieksekusi. Ketika terdapat banyak transaksi yang perlu dieksekusi oleh *miner* pada waktu yang sama, maka transaksi yang memiliki nilai *gasPrice* yang lebih besar akan menjadi prioritas utama untuk dieksekusi.

Karena sifat dari penentuan *gasPrice* yang tidak menentu, maka untuk pengujian besar *transactionFee* dalam eksekusi transaksi yang berisi suatu pemrosesan yang ada dalam sistem akan dilakukan dengan menggunakan beberapa nilai *gasPrice*, mulai dari yang terkecil (1 Gwei) sampai yang terbesar (128 Gwei) berdasarkan situs *web Eth Gas Station*.

Pengujian biaya transaksi untuk pemrosesan yang ada pada sistem *crowdfunding* yang dibangun terbagi menjadi dua jenis, yaitu biaya transaksi untuk eksekusi proses pembentukan proyek galang dana dan biaya transaksi untuk eksekusi proses pemberian dana pada proyek galang dana.

BAB IV

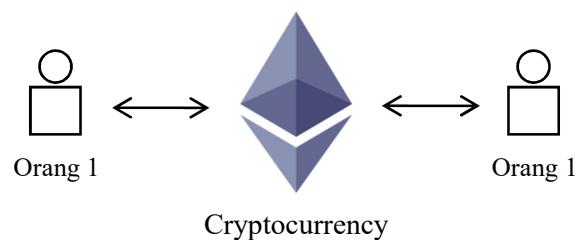
HASIL DAN PEMBAHASAN

4.1. *Cryptocurrency*

Sistem *crowdfunding* yang dibangun dalam tugas akhir ini memanfaatkan adanya teknologi *blockchain* dalam menjalankan proses transaksi keuangan digital tanpa adanya bantuan dari pihak penengah/ketiga. Model keuangan yang ada pada sistem ini umumnya disebut sebagai *cryptocurrency*. Berbeda dengan mata uang fiat, transaksi *cryptocurrency* diproses secara terdesentralisasi dan bersifat global, sehingga dapat menyediakan layanan finansial tanpa menggunakan bantuan dari institusi finansial dan berjalan tanpa adanya ikatan regulasi dari pemerintah. Gambaran perbedaan prosedur transaksi antara *cryptocurrency* dan mata uang fiat dapat dilihat pada Gambar 4.1 dan Gambar 4.2.



Gambar 4. 1 Prosedur Transaksi Mata Uang Fiat



Gambar 4. 2 Prosedur Transaksi *Cryptocurrency*

Kelebihan utama yang diberikan oleh *cryptocurrency* jika dibandingkan dengan mata uang fiat terletak di keamanan proses pengoperasiannya.

Cryptocurrency dioperasikan dengan menggunakan protokol terdesentralisasi sehingga memberikan kepastian terhadap suatu transaksi keuangan yang terjadi bersifat *non-reversible*. Secara umum, kelebihan yang diberikan *cryptocurrency* antara lain sebagai berikut :

- Tidak terikat oleh regulasi sehingga nilai dari *cryptocurrency* hanya akan dipengaruhi berdasarkan *supply & demand*.
- Bersifat independen, artinya tidak dikontrol penuh oleh suatu pihak tertentu.
- Mustahil untuk diimitasi.
- Setiap transaksi yang terjadi akan tersimpan dengan aman karena tercatat menggunakan *cryptography*.
- Bersifat transparan sehingga memberikan kepercayaan kepada pihak yang melakukan transaksi.
- Transaksi internasional dapat diproses dalam waktu yang singkat.
- Biaya untuk transaksi keuangan yang jumlahnya besar terbilang murah.

4.2. Pengujian Fungsionalitas Sistem

Pengujian fungsionalitas sistem dilakukan menggunakan metode *black box testing* untuk masing-masing *use case* berikut.

4.2.1. Pembentukan Proyek

Hasil pengujian proses pembentukan proyek galang dana dengan metode *black box testing* dapat dilihat pada Tabel 4.1.

Tabel 4. 1 Hasil *Black Box Testing* Pembentukan Proyek

No	Aksi	Input & Output	Hasil Pengujian
1.	<p><i>Fundraiser</i> memberikan <i>input</i> data terkait informasi proyek galang dana yang akan dibentuk berupa nama, deskripsi, target dana, dan durasi. <i>Input</i> ini kemudian akan dikirim sebagai data proyek galang dana baru melalui transaksi yang dilakukan oleh <i>fundraiser</i> untuk mengubah <i>state</i> pada <i>smart contract</i> sehingga data proyek galang dana dapat tercatat.</p>	<p><i>Input</i> : nama (<i>text</i>), deskripsi (<i>text</i>), target dana (<i>number</i>), durasi (<i>number</i>). <i>Output</i> : Data proyek galang dana baru tercatat berdasarkan <i>input</i> yang diterima dan logika bisnis yang berjalan dalam <i>smart contract</i>.</p>	Sistem berhasil mencatat data proyek galang dana.
2.	<p><i>Smart Contract</i> proyek galang dana menerima dana sehingga total dana bertambah. Total dana ini selanjutnya akan diproses</p>	<p><i>Input</i> : State total dana (<i>default</i> bernilai 0) yang bertambah berdasarkan jumlah dana yang diterima.</p>	Sistem berhasil mengubah status proyek galang dana menjadi <i>close</i> ketika target dana telah

	<p>untuk penentuan pencapaian target dana dan menyebabkan perubahan pada status proyek apabila telah tercapai.</p>	<p><i>Output</i> : Kasus pertama, ketika target dana tercapai, <i>state</i> status proyek galang dana berubah menjadi <i>close</i>. Kasus kedua, ketika target dana belum tercapai, <i>state</i> status proyek galang dana tidak berubah (<i>open</i>).</p>	tercapai dan apabila belum tercapai maka status proyek galang dana tidak berubah (<i>open</i>).
--	--------------------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------

4.2.2. Pemberian Dana pada Proyek

Hasil pengujian proses pemberian dana pada proyek galang dana dengan metode *black box testing* dapat dilihat pada Tabel 4.2.

Tabel 4. 2 Hasil *Black Box Testing* Pemberian Dana pada Proyek

No	Aksi	<i>Input & Output</i>	Hasil Pengujian
1.	Pemilihan proyek galang dana oleh <i>funder</i> dilakukan dengan mengirim <i>request</i> untuk mendapatkan data detail	<i>Input</i> : <i>request</i> data proyek galang dana menggunakan <i>button click</i> pada tombol detail proyek.	Sistem berhasil mengirimkan respon data detail status proyek galang dana.

	status proyek galang dana.	<i>Output</i> : respon berisi data detail status proyek galang dana.	
2.	<i>Funder</i> memberikan <i>input</i> berupa jumlah dana yang akan diberikan dalam proses pemberian dana ke proyek galang. Input ini kemudian akan divalidasi jumlahnya berdasarkan saldo yang dimiliki oleh <i>funder</i> . Apabila telah tervalidasi, data hasil <i>input</i> tersebut kemudian dikirim dan menyebabkan perubahan total dana yang dimiliki proyek.	<i>Input</i> : jumlah dana (<i>integer</i>) yang akan diberikan oleh <i>funder</i> . <i>Output</i> : perubahan data total dana proyek galang dana pada <i>smart contract</i> ketika <i>input</i> tervalidasi dan proyek galang dana pada <i>smart contract</i> ketika <i>input</i> telah berhasil melalui proses validasi. Apabila tidak berhasil, maka <i>funder</i> akan diminta kembali untuk memberikan <i>input</i> yang sesuai.	Sistem berhasil mengubah data total dana proyek galang dana pada <i>smart contract</i> ketika <i>input</i> tervalidasi dan apabila tidak tervalidasi, sistem meminta <i>funder</i> untuk memberi <i>input</i> baru. Apabila tidak berhasil, maka <i>funder</i> berdasarkan saldo yang dimiliki.

4.2.3. Pengambilan Dana Proyek

Hasil pengujian proses pengambilan dana pada proyek galang dana dengan metode *black box testing* dapat dilihat pada Tabel 4.3.

Tabel 4. 3 Hasil *Black Box Testing* Pengambilan Dana Proyek

No	Aksi	Input & Output	Hasil Pengujian
1.	<i>Smart contract</i> akan melakukan pengiriman total dana yang ada pada proyek ke <i>account</i> milik <i>fundraiser</i> ketika target dana telah tercapai.	<i>Input</i> : State target dana yang berubah menjadi tercapai. <i>Output</i> : Saldo <i>account</i> milik <i>fundraiser</i> bertambah berdasarkan total dana yang terkirim	Sistem berhasil mengirim total dana yang ada pada proyek galang dana ke <i>account</i> milik <i>fundraiser</i> .

4.2.4. Pengembalian Dana Proyek

Hasil pengujian proses pengembalian dana pada proyek galang dana dengan metode *black box testing* dapat dilihat pada Tabel 4.4.

Tabel 4. 4 Hasil *Black Box Testing* Pengembalian Dana Proyek

No	Aksi	Input & Output	Hasil Pengujian
1.	<i>Funder</i> mengirimkan <i>request</i> data terkait informasi proyek galang dana yang durasinya telah habis untuk mengetahui jumlah dana yang pernah diberikan	<i>Input</i> : <i>request</i> data berupa <i>button click</i> tombol detail proyek. <i>Output</i> : respon data dana yang dimiliki <i>funder</i> . Apabila ada, maka <i>funder</i> dapat	Sistem berhasil mengirim respon data dana yang dimiliki oleh <i>funder</i> sehingga proses pengembalian dana dapat dilakukan.

	pada proyek tersebut.	melakukan proses pengembalian dana. Jika tidak, maka proses pengembalian dana tidak dapat dilakukan.	
2.	<i>Funder</i> melakukan transaksi pada <i>smart contract</i> untuk mendapatkan pengembalian dana berdasarkan jumlah dana yang pernah diberikan pada proyek galang dana.	<i>Input</i> : <i>button click</i> untuk memulai pengembalian dana proyek. <i>Output</i> : perubahan <i>state total dana yang tersisa</i> untuk dikembalikan pada <i>smart contract</i> dan saldo <i>account milik funder</i> akan bertambah.	Sistem berhasil mengubah <i>state total dana yang tersisa</i> dan saldo <i>account milik funder</i> bertambah setelah pengembalian dana proyek dilakukan.

4.3. Analisis Keamanan dan Transparansi Sistem

4.3.1. Dasar Teoritis

Metode *blockchain* telah terbukti mampu memberikan kepastian keamanan dalam transaksi keuangan. Saat ini, telah banyak *cryptocurrency* yang berjalan di seluruh dunia yang terbentuk menggunakan basis teknologi

blockchain, termasuk Ethereum. Berikut beberapa karakteristik *blockchain* Ethereum yang dapat menjamin keamanan dan transparansi pada proses transaksi keuangan dalam sistem *crowdfunding* yang dibangun.

a. Terdesentralisasi (*Decentralized*)

Blockchain dikelola secara terdesentralisasi dimana tidak terdapat satu pihak utama yang memiliki hak akses penuh terhadap proses yang ada. Pengelolaan rantai *block* data dapat dilakukan oleh seluruh *node* yang ada dalam sistem *blockchain*. Dengan demikian, setiap *node* memiliki hak akses yang sama terhadap rantai *block* yang terbentuk.

b. Jaringan *Peer-to-Peer* (P2P)

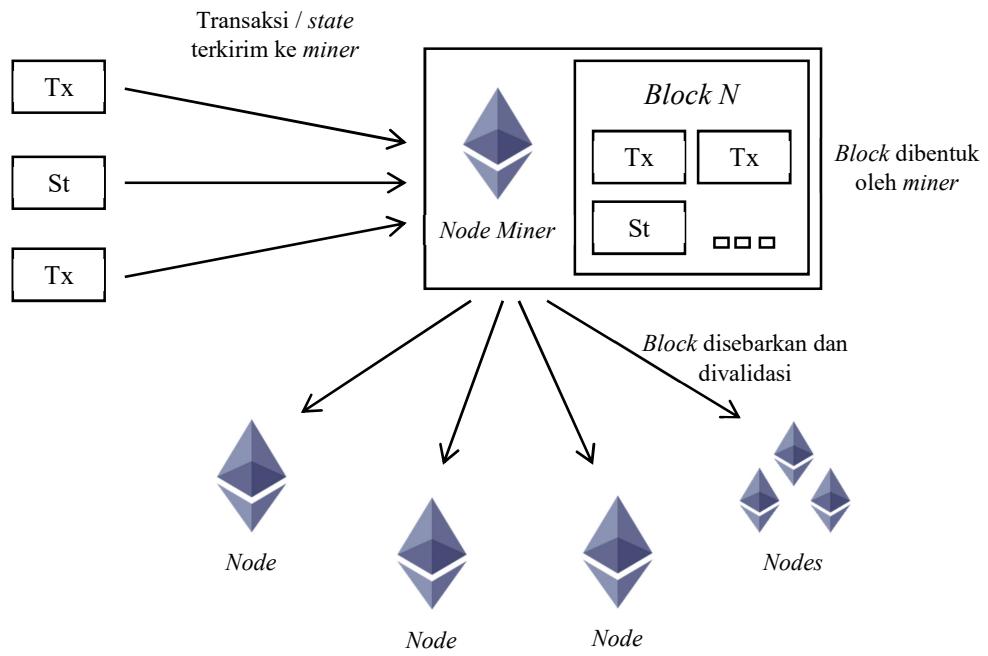
Setiap *node* yang tergabung dalam sistem *blockchain* akan berkomunikasi melalui jaringan yang bersifat *peer-to-peer*. Metode komunikasi ini digunakan atas dasar sifat terdesentralisasi dari *blockchain*. Ketika salah satu *node* melakukan pencatatan data dengan membentuk *block* baru pada rantai *block* yang ada, *block* baru ini akan diumumkan ke *node* lain dalam jaringan. Rantai *block* yang terbentuk pada dasarnya akan tersalin ke seluruh *node* dalam sistem *blockchain*. Dengan demikian, data yang tercatat dalam rantai *block* rentan untuk hilang ataupun terhapus karena tersebar di seluruh *node*. Semakin banyak *node* yang tergabung dalam jaringan *blockchain* maka semakin mustahil rantai *block* data hilang atau terhapus. Karakteristik ini mampu memberikan keandalan pada data dalam sistem *blockchain*.

c. Protokol Konsensus

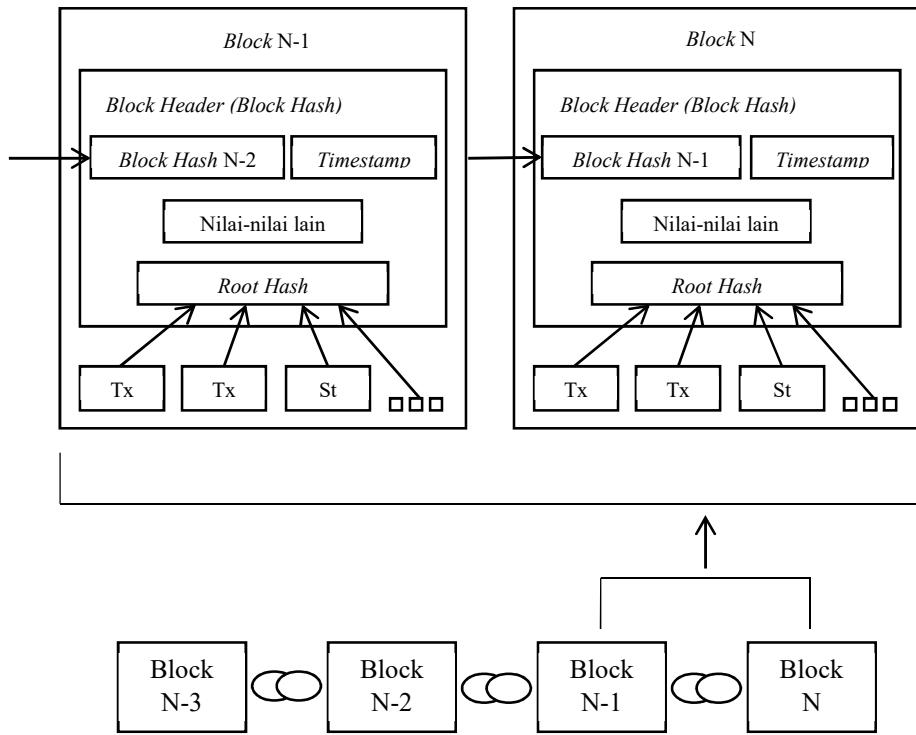
Blockchain dapat berjalan secara terdesentralisasi, dimana tidak terdapat satu pihak utama yang memiliki hak akses penuh terhadap proses yang ada, dengan adanya penerapan mekanisme menggunakan protokol konsensus seperti *Proof of Work*. Ketika salah satu *node* yang tergabung dalam jaringan *blockchain* melakukan proses pencatatan data dengan membentuk *block* baru, proses ini harus mengikuti mekanisme yang diterapkan oleh protokol konsensus sehingga *node* lain dalam jaringan dapat menyetujui dan menganggap *block* yang terbentuk sebagai *block* yang valid.

Pada protokol *Proof of Work* sendiri, suatu *block* baru akan dianggap valid ketika telah melalui proses komputasi yang besar, dimana bukti komputasi ini diberikan melalui pembentukan nilai *nonce* pada *blockchain*. *Block* yang terbentuk melalui mekanisme ini akan saling terhubung satu sama lain sehingga ketika terjadi perubahan pada suatu *block*, akan mempengaruhi keabsahan dari *block* setelahnya. Akibatnya, upaya yang perlu dilakukan oleh seorang penyerang untuk melakukan manipulasi terhadap rantai *block* yang ada membutuhkan proses yang panjang dan komputasi yang besar.

Gambaran umum proses pencatatan data dan struktur *block* data yang terbentuk dengan mekanisme protokol konsensus *Proof of Work* dapat dilihat pada Gambar 4.3 dan Gambar 4.4.



Gambar 4.3 Proses Pencatatan Data *Blockchain*



Gambar 4.4 Struktur Rantai *Block*

Protokol konsensus dapat juga dikatakan sebagai protokol yang digunakan *blockchain* untuk menghindari adanya manipulasi data oleh suatu pihak penyerang yang bergabung sebagai *node* dalam jaringan. Setiap *node* pada hakikatnya akan mempercayai dan menggunakan rantai *block* terpanjang. Rantai *block* terpanjang merupakan rantai *block* yang terbentuk melalui proses komputasi terbesar dibanding rantai *block* lain. Setiap *node* jujur akan saling bekerja sama untuk membentuk *block* yang valid, sehingga ketika seorang *node* penyerang ingin melakukan manipulasi pada rantai *block*, harus membentuk rantai *block* baru yang lebih panjang dari rantai *block* yang dibentuk oleh *node* jujur. Untuk melakukan hal itu, *node* penyerang harus memiliki kekuatan komputasi yang melebihi gabungan kekuatan komputasi *node* jujur. Semakin banyak *node* jujur yang tergabung dalam jaringan *blockchain*, maka akan semakin mustahil penyerangan dapat terjadi. Sebagai hasilnya, mekanisme ini memberikan kekekalan pada data dalam sistem *blockchain*.

d. Aksesibilitas Publik

Blockchain dikelola secara terdesentralisasi dengan masing-masing pihak *node* memiliki hak akses yang sama terhadap rantai *block* data yang ada. *Blockchain* umumnya bersifat terbuka/publik sehingga memungkinkan setiap pihak bergabung sebagai *node* dalam melakukan pengelolaan rantai *block*. Aksesibilitas yang bersifat publik ini tentu memberikan keutuhan terhadap transparansi dari setiap proses pencatatan data yang terjadi dalam sistem *blockchain*.

4.3.2. Pengujian *Blockchain* Ethereum

Pengujian kekekalan dan keandalan data pada *blockchain* dari sistem *crowdfunding* yang dibangun dilakukan dengan membentuk jaringan Ethereum yang bersifat *private*. *Smart contract* dari sistem *crowdfunding* akan dimigrasikan ke *blockchain* jaringan Ethereum yang dibentuk. Pada jaringan Ethereum yang bersifat *private* ini, terdapat tiga *node* yang saling berkomunikasi antara satu sama lain untuk melakukan pencatatan data transaksi sistem *crowdfunding*.

Pengujian *blockchain* sistem dilakukan berdasarkan tiga kondisi. Kondisi pertama merupakan kondisi dimana salah satu *node* akan dianggap sebagai *node* penyerang dan melakukan pembentukan *block (mining)* secara individu tanpa komunikasi dengan *node* lain. Pengujian dengan kondisi pertama dilakukan untuk mengetahui bagaimana keamanan dari rantai *block* data yang ada ketika terdapat *node* penyerang yang melakukan *mining* tetapi dengan kekuatan komputasi yang tidak melebihi kekuatan komputasi *node* jujur dalam jaringan. Kondisi kedua merupakan kondisi dimana terdapat dua *node* yang melakukan pencatatan data transaksi yang diindikasikan sebagai *double spending*, yaitu dua transaksi yang dilakukan oleh suatu *account* menggunakan *digital currency* yang sama. Pengujian dengan kondisi kedua dilakukan untuk mengetahui pencegahan terhadap transaksi yang diindikasikan sebagai *double spending*. Kondisi ketiga merupakan kondisi dimana beberapa data dalam rantai *block* yang ada di salah satu *node* terhapus. Pengujian dengan kondisi ketiga dilakukan untuk mengetahui pengaruh dari rantai *block* yang sudah dianggap tidak sesuai terhadap rantai *block* yang disetujui dan tersimpan di setiap *node* dalam jaringan.

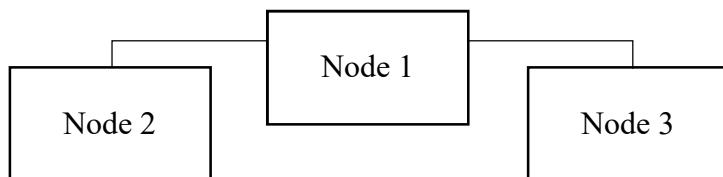
Dalam pembentukan suatu *node* Ethereum, hal utama yang perlu diperhatikan merupakan *genesis block*. *Genesis block* pada dasarnya merupakan *block* pertama dari rantai *block* yang akan tersimpan pada setiap *node*. *Genesis block* berisi konfigurasi dari jaringan Ethereum yang akan digunakan dan aturan pembentukan *account* maupun *block* data transaksi. Seluruh *node* yang tergabung pada suatu jaringan Ethereum harus memiliki *genesis block* yang sama agar dapat melakukan komunikasi antara satu sama lain dalam melakukan pembentukan rantai *block* data transaksi. Konfigurasi pada *genesis block* yang digunakan dalam pengujian dapat dilihat pada Gambar 4.5.

```
{
  "config": {
    "chainId": 5852,
    "homesteadBlock": 0,
    "eip150Block": 0,
    "eip150Hash": "0x000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000",
    "eip155Block": 0,
    "eip158Block": 0,
    "byzantiumBlock": 0,
    "constantinopleBlock": 0,
    "petersburgBlock": 0,
    "istanbulBlock": 0,
    "ethash": {}
  },
  "nonce": "0x0",
  "timestamp": "0x5f92ca5b",
  "extraData": "0x000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000",
  "gasLimit": "0x47b760",
  "difficulty": "0x8000",
  "mixHash": "0x000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000",
  "coinbase": "0x0000000000000000000000000000000000000000000000000000000000000000",
  "alloc": { ... },
  "number": "0x0",
  "gasUsed": "0x0",
  "parentHash": "0x000000000000000000000000000000000000000000000000000000000000000"
}
```

Gambar 4. 5 Konfigurasi *Genesis Block*

Node yang dibentuk pada jaringan Ethereum *private* untuk pengujian akan menggunakan *client* yang dinamakan Go-Ethereum (Geth). Ketiga *node* yang

digunakan akan dijalankan dengan *instance* geth yang berbeda dalam satu mesin yang sama. *Node 1* akan dikonfigurasikan sebagai *peer* pada *node 2* dan *node 3*. Hubungan antara ketiga *node* dalam pengujian yang dilakukan dapat dilihat pada Gambar 4.6.



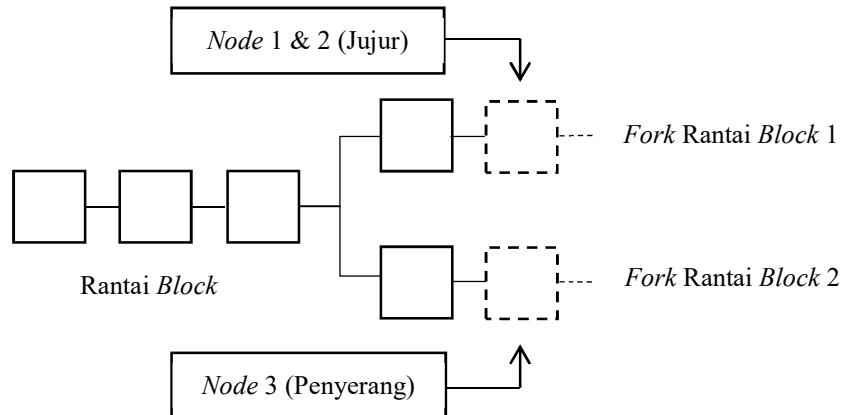
Gambar 4. 6 Konfigurasi Node Pengujian

4.3.2.1.Pengujian Mayoritas Node Jujur (Kondisi Pertama)

Pengujian pertama yang dilakukan merupakan gambaran ketika suatu *node* penyerang melakukan pencatatan *block* dengan kekuatan komputasi yang tidak melebihi dari gabungan kekuatan komputasi seluruh *node* jujur yang tergabung dalam jaringan Ethereum. Pengujian ini akan membuktikan bagaimana keamanan dari rantai *block* yang ada dapat terjaga ketika mayoritas kekuatan komputasi pada jaringan dikendalikan oleh *node* jujur.

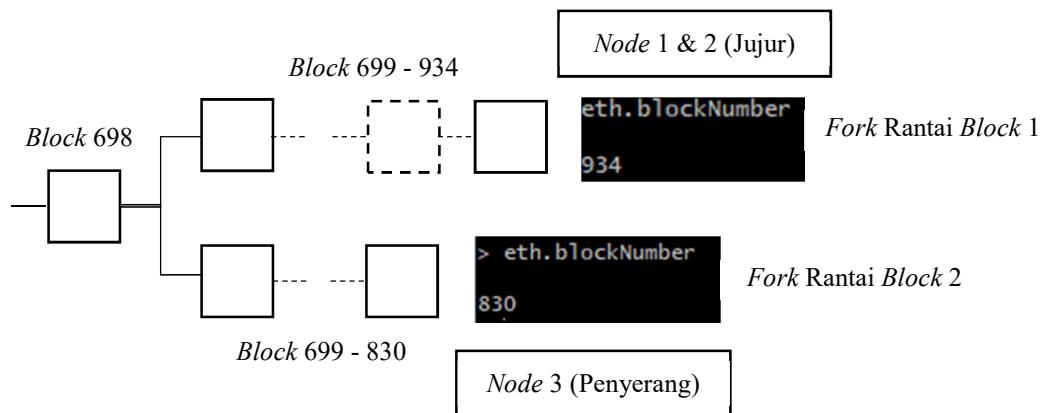
Pada pengujian ini, *node 3* akan melakukan proses mining secara individu dengan tidak melakukan *peer* pada *node 1*. Pencatatan *block* pada *node 3* akan diasumsikan sebagai pencatatan *block* yang tidak sesuai. *Node 1* dan *node 2* bekerja sama dalam melakukan proses pembentukan *block*, sementara *node 3* melakukannya secara individu. Pengujian ini akan membentuk *fork* pada rantai *block* yang ada, dimana *fork* rantai *block 1* merupakan *fork* yang dibentuk oleh mayoritas *node* (*node 1* dan *node 2*) dan *fork* rantai *block 2* merupakan *fork* yang

dibentuk oleh *node* penyerang (*node* 3). Gambaran pengujian ini dapat dilihat pada Gambar 4.7.



Gambar 4. 7 Gambaran Pengujian Mayoritas *Node* Jujur (Kondisi Pertama)

Kondisi awal rantai *block* dari ketiga *node* sebelum proses *mining* dilakukan berjumlah 698 *block*. Proses *mining* di seluruh *node* akan dilakukan dalam waktu 15 menit. Rantai *block* yang terbentuk setelah proses *mining* di setiap *node* dapat dilihat pada Gambar 4.8.



Gambar 4. 8 Rantai *Block* Setelah Proses *Mining* (Pengujian Kondisi Pertama)

Setelah proses *mining* selama 15 menit, rantai *block* yang ada pada *node* 1 dan *node* 2 (*Fork Rantai Block 1*) berjumlah 934 *block* dan rantai *block* yang ada pada *node* 3 (*Fork Rantai Block 2*) berjumlah 830 *block*. Ketika *node* 3 melakukan

peer kembali dengan *node 1*, *node 3* melakukan proses sinkronisasi *block* dikarenakan rantai *block* pada *node* lain lebih panjang daripada rantai *block* yang *node 3* bentuk secara individu. *Node 1* dan *node 2* tidak akan melakukan sinkronisasi karena rantai *block* yang dimiliki adalah rantai *block* yang terpanjang. *Fork* rantai *block* 2 yang dibentuk oleh *node 3* pada akhirnya akan diabaikan. Proses sinkronisasi pada *node 3* dapat dilihat pada Gambar 4.9.

```
INFO [10-29|18:08:55.733] Block synchronisation started
INFO [10-29|18:08:55.850] Downloader queue stats
INFO [10-29|18:08:55.870] Imported new chain segment
WARN [10-29|18:08:56.063] Large chain reorg detected
INFO [10-29|18:08:56.105] Imported new chain segment
INFO [10-29|18:08:58.904] Imported new chain segment
```

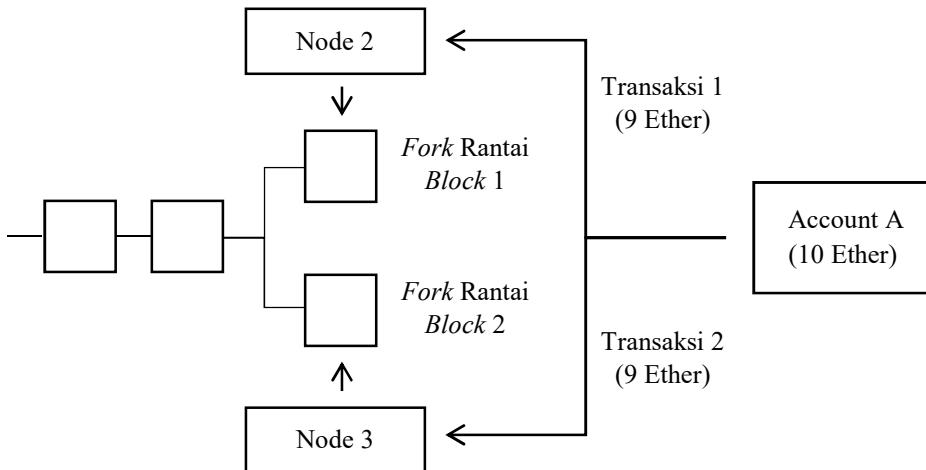
Gambar 4.9 Sinkronisasi Rantai *Block Node 3* (Pengujian Kondisi Pertama)

Dalam jaringan Ethereum, *node* pada hakikatnya hanya akan selalu mempercayai rantai *block* yang terpanjang. Hal ini merupakan bagian dari protokol *Proof of Work* yang digunakan oleh *blockchain* Ethereum, dimana *node* akan selalu mempercayai dan menggunakan rantai *block* yang terpanjang. Jika suatu *node* memiliki rantai *block* yang pendek tergabung ke dalam jaringan, maka rantai *block* tersebut akan diabaikan.

4.3.2.2.Pengujian *Double Spending* (Kondisi Kedua)

Pengujian kedua dilakukan untuk mengetahui proteksi yang dilakukan oleh sistem terhadap transaksi yang terindikasi sebagai *double spending*. Pengujian dilakukan dengan menggunakan *account A* yang memiliki jumlah saldo berjumlah 10 Ether melakukan transaksi pemberian dana bernilai 9 Ether sebanyak dua kali pada dua proyek galang dana berbeda dalam sistem *crowdfunding* yang dibangun. Untuk membentuk kedua transaksi ini, *peer* dari setiap *node* dalam jaringan akan dihilangkan sehingga masing-masing *node*

melakukan pencatatan *block* secara individu tanpa adanya proses sinkronisasi rantai *block*. Aplikasi sisi klien sistem *crowdfunding* akan dihubungkan ke *node 2* untuk melakukan proses pemberian dana yang pertama bernilai 9 Ether menggunakan *account A* ke proyek galang dana yang dinamakan Proyek 1 (Transaksi 1). Data Transaksi 1 akan tersimpan ke dalam rantai *block* yang ada pada *node 2*. Selanjutnya aplikasi sisi klien sistem *crowdfunding* dihubungkan ke *node 3* untuk melakukan proses pemberian dana yang kedua bernilai 9 Ether menggunakan *account A* ke proyek galang dana yang dinamakan Proyek 2 (Transaksi 2). Data Transaksi 2 akan tersimpan ke rantai *block* yang ada pada *node 3*. Gambaran pembentukan *double spending* dapat dilihat pada Gambar 4.10.



Gambar 4. 10 Pembentukan Double Spending (Pengujian Kondisi Kedua)

Proses ini menghasilkan adanya 2 *block* dengan nomor yang sama tetapi dengan isi yang berbeda. Dengan demikian, terbentuk *fork* pada rantai *block* yang ada dalam jaringan. *Block* data transaksi di kedua *node* dapat dilihat pada Gambar 4.11 dan Gambar 4.12.

Gambar 4. 11 *Block Data Transaksi 1 di Node 2 (Pengujian Kondisi Kedua)*

Gambar 4.12 Block Data Transaksi 2 di Node 3 (Pengujian Kondisi Kedua)

Protokol Ethereum dirancang untuk menyelesaikan permasalahan ini dengan melakukan penentuan rantai *block* yang akan digunakan berdasarkan panjang rantainya. Ketika terdapat 2 rantai *block* yang panjangnya sama dalam jaringan, setiap *node* akan menunggu salah satu rantai *block* tersebut mengalami pertambahan *block*. Pada akhirnya, *node* akan menggunakan rantai *block* yang terlebih dahulu mengalami pertambahan *block*.

Pada kondisi pengujian *double spending* yang dilakukan, rantai *block* pada *node 3* (*Fork Rantai Block 2*) kemudian mengalami pertambahan *block* lebih cepat

dibandingkan rantai *block* yang ada pada *node* 2 (*Fork Rantai Block 1*) sehingga membentuk rantai *block* yang lebih panjang. Ketika seluruh *node* pengujian melakukan *peer kembali*, *node* 1 dan *node* 2 melakukan sinkronisasi untuk menyalin rantai *block* yang ada pada *node* 3, sementara rantai *block* yang sebelumnya ada pada *node* 2 akan diabaikan. Pada akhirnya, transaksi *double spending* (Transaksi 1 dan 2) yang terjadi hanya berhasil dicatat sebanyak satu transaksi saja, yaitu Transaksi 2 yang terjadi pada rantai *block* di *node* 3. Proses dan hasil sinkronisasi ini dapat dilihat pada Gambar 4.13 dan Gambar 4.14.

```
> INFO [10-29|20:38:48.998] Block synchronisation started  
INFO [10-29|20:38:49.010] Downloader queue stats  
INFO [10-29|20:38:49.027] Chain reorg detected
```

Gambar 4. 13 Sinkronisasi Rantai *Block* (Pengujian Kondisi Kedua)

Gambar 4. 14 Block Data Transaksi Double Spending Setelah Sinkronisasi
(Pengujian Kondisi Kedua)

4.3.2.3. Pengujian Rantai *Block* Tidak Valid (Kondisi Ketiga)

Pengujian ketiga dilakukan untuk mengetahui pengaruh dari rantai *block* yang tidak valid terhadap rantai *block* utama yang digunakan dan tersimpan di setiap *node* dalam jaringan. Selain itu, pengujian juga dilakukan untuk

mengetahui kondisi pada suatu *node* ketika rantai *block* yang tersimpan di *node* tersebut sudah tidak valid.

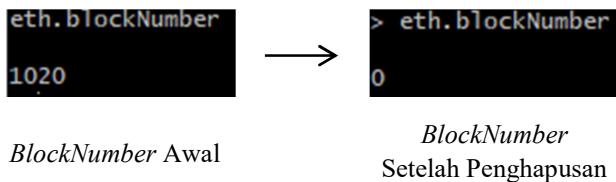
Pada pengujian ini, pembentukan rantai *block* yang tidak valid akan dilakukan pada *node* 2 dengan melakukan penghapusan salah satu *file* data pada *directory* *chaindata* dari Geth sehingga menyebabkan tidak lengkapnya data pada rantai *block* yang tersimpan. Metode penghapusan ini digunakan karena seluruh data dari rantai *block* dalam *node* Ethereum tersimpan dengan menggunakan fungsi *hash* kriptografi dan basis data yang digunakan bersifat *append-only* sehingga tidak terdapat cara lain untuk melakukan penghapusan data yang dapat menyebabkan rantai *block* menjadi tidak valid. Penghapusan *file* data pada *chaindata* Geth dapat dilihat pada Gambar 4.15.

ancient	10/23/202...	File folder	
000029	10/29/202...	Microsoft ...	423 KB
000030	10/29/202...	Microsoft ...	9 KB
000033	10/29/202...	Microsoft ...	35 KB
000034	10/30/202...	Text Docu...	156 KB
CURRENT	10/29/202...	File	1 KB
CURRENT.bak	10/29/202...	BAK File	1 KB
LOCK	10/23/202...	File	0 KB
LOG	10/30/202...	File	10 KB
MANIFEST-000035	10/29/202...	File	1 KB

Gambar 4. 15 Penghapusan Data Chaindata Geth (Pengujian Kondisi Ketiga)

Penghapusan data ini akan menyebabkan rantai *block* *node* 2 menjadi tidak valid. Hal ini dapat dibuktikan setelah melakukan penghapusan *block* data, jumlah *block* yang terbaca pada *node* 2 menjadi 0 yang sebelumnya berjumlah 1020 pada kondisi awal sebelum pengujian dilakukan. Pengecekan jumlah *block*

pada *node* 2 dilakukan dengan menjalankan *command eth.blockNumber* pada CLI Geth yang dapat dilihat pada Gambar 4.16.



Gambar 4. 16 Jumlah *Block Node* 2 Setelah Penghapusan Data (Pengujian Kondisi Ketiga)

Selain itu, penghapusan data ini juga mengakibatkan *node* 2 tidak dapat berfungsi dengan baik dalam berkomunikasi dengan *node* 1 dan 3. Hal ini terbukti saat menjalankan *command* untuk melakukan *peer* pada *node* 2, *output* dari *command* tersebut bernilai *empty*. Output *command peer* pada *node* 2 dapat dilihat pada Gambar 4.17.

```
> admin.addPeer("enode://3992c7d10fc2fe3f73c8468a31408abc5c2e...")  
true  
INFO [10-29|21:44:26.987] Looking for peers  
> admin.peers  
[]
```

Gambar 4. 17 Output *Command Peer* Pada *Node* 2 (Pengujian Kondisi Ketiga)

Tidak berfungsinya komunikasi antara *node* 2 yang berisi rantai *block* yang tidak valid dengan *node* lain pada jaringan tentu tidak akan mengganggu kondisi dari rantai *block* utama yang disetujui dan tersimpan dalam jaringan. *Blockchain* Ethereum pada hakikatnya akan berjalan secara terdesentralisasi, dimana ketika terdapat satu *node* yang mengalami gangguan, tidak akan berdampak pada proses yang dilakukan oleh *node* lain yang tergabung dalam jaringan. Dengan demikian, rantai *block* utama tetap dapat terjaga dan tersimpan

dengan utuh di *node* lain walaupun terdapat satu *node* yang berisi rantai *block* tidak valid.

4.3.3. Analisis Terhadap Serangan Digital Lainnya

Berdasarkan karakteristik dan pengujian yang dilakukan pada *platform blockchain* Ethereum yang digunakan dalam sistem *crowdfunding* yang dibangun, dapat disimpulkan bahwa sifat terdesentralisasi dan protokol-protokol khusus dari Ethereum yang menyebabkan kekekalan dan keandalan pada data memberikan manfaat pada keamanan sistem sehingga berbagai jenis serangan yang umumnya terjadi pada sistem berbasis aplikasi *web* yang bersifat tersentralisasi dapat teratasi. Contoh jenis serangan ini dapat dilihat pada Tabel 4.5.

Tabel 4.5 Jenis Serangan yang Teratas

No.	Jenis Serangan	Penjelasan
1.	<i>SQL Injection</i>	Merupakan serangan yang digunakan untuk mendapatkan akses ke <i>database</i> yang tersimpan di <i>server</i> . Serangan ini umumnya dilakukan dengan menginjeksi <i>query SQL</i> pada <i>form</i> yang ada dalam <i>web</i> seperti <i>login form</i> . Dengan penerapan <i>blockchain</i> pada sistem <i>crowdfunding</i> yang dibangun, jenis penyerangan ini tidak dapat dilakukan karena sistem dibangun secara terdesentralisasi. Seluruh <i>node</i> pada dasarnya memiliki akses data yang ada dalam jaringan <i>blockchain</i> , tetapi untuk melakukan penulisan data

		ataupun transaksi baru, harus melalui protokol konsensus <i>Proof of Work</i> yang diimplementasikan sehingga hanya data valid saja yang dapat tercatat.
2.	<i>Cross Site Scripting (XSS)</i>	Merupakan serangan yang terjadi ketika suatu <i>web</i> memiliki kelemahan yang memungkinkan adanya injeksi <i>script</i> . Pada sistem <i>crowdfunding</i> yang dibangun, jenis serangan ini tidak dapat dilakukan. Pencatatan <i>smart contract</i> yang dilakukan dalam sistem hanya dapat dilakukan 1 kali saja dan bersifat kekal. Logika bisnis sistem <i>crowdfunding</i> yang dikembangkan melalui <i>smart contract</i> akan tersimpan ke dalam <i>blockchain</i> sehingga tidak dapat berubah.
3.	<i>Denial of Service (DoS) / Distributed Denial of Service (DDoS)</i>	Merupakan serangan yang dilakukan oleh penyerang untuk melumpuhkan <i>service</i> yang diberikan oleh suatu <i>server</i> dengan cara mengirim <i>traffic</i> yang jumlahnya sangat besar dalam satu waktu. Serangan ini tidak dapat dilakukan pada sistem <i>crowdfunding</i> yang dibangun karena sistem bersifat terdesentralisasi, dengan kata lain tidak terpusat pada satu <i>node</i> saja. Ketika serangan ini dilakukan pada suatu <i>node</i> dalam jaringan sistem, maka <i>node</i> lain dapat menggantikan fungsi sebagai

		penyedia <i>service</i> sehingga sistem tetap dapat berfungsi dengan baik.
4.	<i>OS Command Injection</i>	<p>Merupakan serangan yang dilakukan oleh penyerang untuk mengontrol <i>web</i> dalam hal modifikasi data yang ada dengan cara menginjeksi perintah sistem operasi pada server aplikasi <i>web</i>.</p> <p>Pada sistem <i>crowdfunding</i> yang dibangun, serangan ini tidak dapat dilakukan karena sistem dibangun secara terdesentralisasi. Modifikasi data yang ada dalam sistem tidak dapat dilakukan, dan untuk melakukan pencatatan data yang baru harus melalui protokol <i>Proof of Work</i>, dimana hanya data valid saja yang dapat tercatat.</p>
5.	<i>Brute Force Attack</i>	<p>Merupakan serangan kata sandi (<i>password</i>) suatu akun yang dilakukan oleh penyerang dengan cara mencoba satu persatu kombinasi <i>username</i> dan <i>password</i> yang berbeda sampai akun berhasil dibuka.</p> <p>Dalam sistem <i>crowdfunding</i> yang dibangun, manajemen akun secara langsung dikelola melalui <i>platform Ethereum</i>. Untuk mengakses akun yang ada dalam Ethereum, dibutuhkan <i>private key</i> untuk masing-masing akunnya yang dibentuk secara acak (256 bits).</p> <p>Apabila serangan ini dilakukan untuk</p>

		mendapatkan <i>private key</i> suatu akun Ethereum, maka untuk mendapatkan suatu <i>private key</i> , kesempatan yang dimiliki oleh penyerang adalah 1 dibanding 2^{256} , dengan kata lain mustahil dilakukan secara praktik.
6.	Jenis serangan lainnya	Jenis-jenis serangan lain yang umumnya dilakukan pada sistem tersentralisasi dengan tujuan mendapatkan akses penuh ataupun modifikasi data pada sistem dapat diatasi oleh sistem <i>crowdfunding</i> yang dibangun secara terdesentralisasi dimana tidak terdapat otoritas utama yang memiliki akses penuh dalam melakukan modifikasi pada sistem.

4.4. Pengujian Besar Biaya Transaksi

4.4.1. Pemrosesan Pembentukan Proyek

Transaksi untuk eksekusi pemrosesan pembentukan proyek galang dana dalam sistem *crowdfunding* yang dibangun akan dilakukan menggunakan *function createProject* pada *smart contract* melalui *request* seorang *fundraiser* (pengirim transaksi). Pada transaksi tersebut, *fundraiser* akan membayar biaya transaksi yang besarnya ditentukan berdasarkan jumlah *gas* yang terpakai (*gasUsed*) dalam eksekusi proses komputasinya.

Untuk mengetahui nilai biaya transaksi (*transactionFee*) yang dibutuhkan pada pemrosesan ini, terlebih dahulu akan dilakukan pengujian untuk mengetahui nilai *gasUsed*.

Pengujian untuk mengetahui nilai *gasUsed* yang dibutuhkan pada transaksi eksekusi pemrosesan ini dilakukan dengan menjalankan proses pembentukan proyek menggunakan lima kondisi berbeda berdasarkan *input* data proyek yang diberikan dalam pembentukannya. Hasil pengujian ini dapat dilihat pada Tabel 4.6.

Tabel 4.6 Hasil Pengujian Nilai *gasUsed* untuk Lima Kondisi Pemrosesan

Pembentukan Proyek

No.	Kondisi Pemrosesan (<i>Input</i> data proyek)	<i>gasUsed</i>
1.	Nama proyek : 1 <i>character</i> (<i>string</i>) Target dana : 1 Ether (<i>integer</i>) Durasi : 1 hari (<i>integer</i>) Deskripsi : 1 <i>character</i> (<i>string</i>)	925408
2.	Nama proyek : 44 <i>character</i> (<i>string</i>) Target dana : 1 Ether (<i>integer</i>) Durasi : 1 hari (<i>integer</i>) Deskripsi : 414 <i>character</i> (<i>string</i>)	1253143
3.	Nama proyek : 50 <i>character</i> (<i>string</i>) Target dana : 1 Ether (<i>integer</i>) Durasi : 1 hari (<i>integer</i>) Deskripsi : 447 <i>character</i> (<i>string</i>)	1259153
4.	Nama proyek : 46 <i>character</i> (<i>string</i>) Target dana : 1 Ether (<i>integer</i>)	1363417

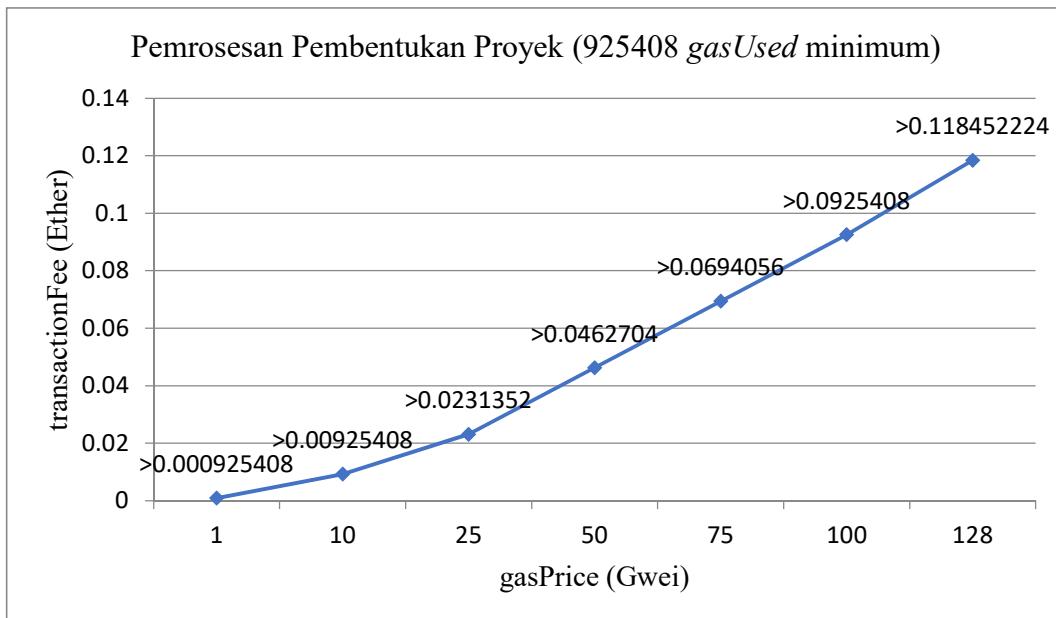
	Durasi : 1 hari (<i>integer</i>) Deskripsi : 581 <i>character</i> (<i>string</i>)	
5.	Nama proyek : 42 <i>character</i> (<i>string</i>) Target dana : 1 Ether (<i>integer</i>) Durasi : 1 hari (<i>integer</i>) Deskripsi : 872 <i>character</i> (<i>string</i>)	1551740

Kondisi pemrosesan pada Tabel 4.6 memiliki nilai *gasUsed* yang berbeda berdasarkan *input* data proyek. Semakin besar ukuran *input* data, maka akan semakin besar pula alokasi penyimpanan yang dibutuhkan pada saat eksekusi pemrosesan. Pemakaian *gas* akan semakin besar seiring dengan besarnya kebutuhan penyimpanan pada eksekusi pemrosesan.

Pada kondisi pemrosesan yang pertama dalam Tabel 4.6, terlihat bahwa *input* yang diberikan merupakan ukuran standar dari data yang diperlukan untuk pembentukan proyek. Dapat disimpulkan bahwa diluar dari besar ukuran *input* yang diberikan sebagai data dari proyek pada sistem, nilai *gasUsed* minimum yang dibutuhkan dalam menjalankan *function createProject* untuk proses pembentukan proyek pada sistem adalah 925408.

Pengujian besar biaya transaksi (*transactionFee*) untuk eksekusi pemrosesan ini dilakukan dengan menggunakan kondisi pertama Tabel 4.6 dengan nilai *gasUsed* sebesar 925408, sebagai standar minimum untuk penentuan nilai *transactionFee*. Selain itu, parameter *gasPrice* yang digunakan untuk pengujinya bernilai 1, 10, 25, 33 (Rata-rata nilai *gasPrice* Ethereum pada waktu

studi dilakukan), 50, 75, 100 dan 128. Hasil pengujian ini dapat dilihat pada Gambar 4.18 dan tabel data Lampiran 3.



Gambar 4. 18 Besar Biaya Transaksi (*transactionFee*) Eksekusi Pemrosesan Pembentukan Proyek dalam *Ether*

Besar biaya transaksi minimum untuk permrosesan pembentukan proyek berkisar antara $0.000925408 - 0.118452224$ Ether. Jika dikonversikan ke dalam Rupiah (konversi 1:3451991.21), besar biaya transaksi minimum tersebut beriksar antara Rp3.194,00 – Rp408.896,00.

4.4.2. Pemrosesan Pemberian Dana pada Proyek

Transaksi untuk eksekusi pemrosesan pemberian dana dalam sistem *crowdfunding* yang dibangun akan dilakukan menggunakan *function funding* pada *smart contract* melalui *request* seorang *funder* (pengirim transaksi). Pada transaksi tersebut, *funder* akan membayar biaya transaksi yang besarnya

ditentukan berdasarkan jumlah *gas* yang terpakai (*gasUsed*) dalam eksekusi proses komputasinya.

Untuk mengetahui nilai biaya transaksi (*transactionFee*) yang dibutuhkan, terlebih dahulu akan dilakukan pengujian untuk mengetahui nilai *gasUsed* dari proses eksekusinya.

Pengujian nilai *gasUsed* yang dibutuhkan dalam melakukan transaksi eksekusi pemrosesan ini dilakukan dengan menjalankan proses pemberian *dana* pada suatu proyek menggunakan tiga kondisi berbeda berdasarkan urutan penerimaan dana yang diterima suatu proyek. Hasil pengujian ini dapat dilihat pada Tabel 4.7.

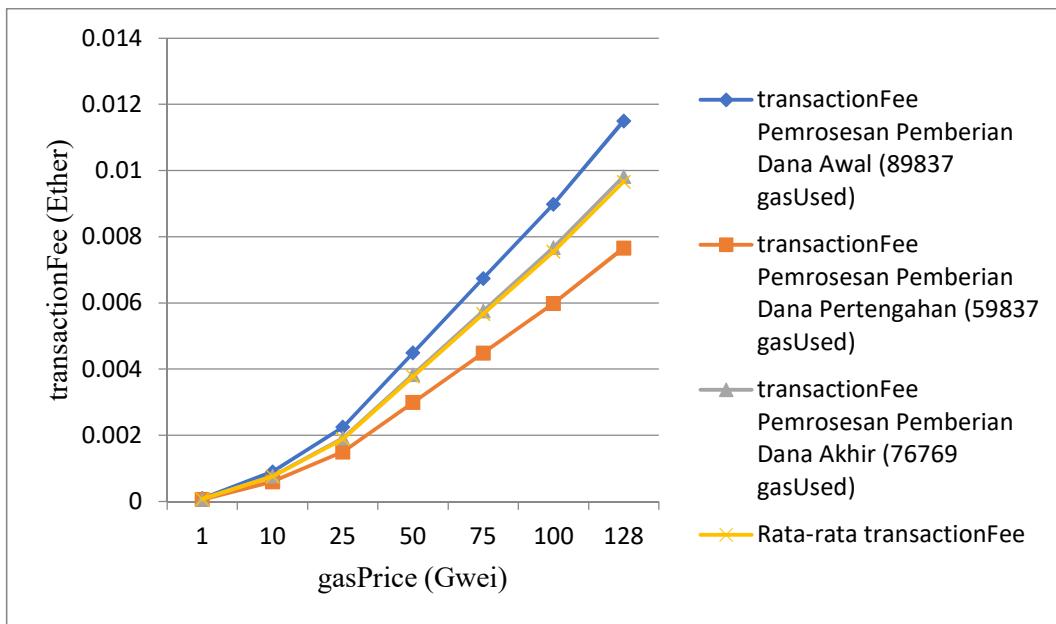
Tabel 4. 7 Hasil Pengujian Nilai *gasUsed* untuk Tiga Kondisi Pemrosesan Pemberian Dana pada Proyek

No.	Kondisi Pemrosesan (Urutan pemberian dana pada suatu proyek)	<i>gasUsed</i>	Penjelasan
1.	Pemrosesan pemberian dana awal	89837	Merupakan pemrosesan pemberian dana pertama oleh seorang <i>funder</i> pada suatu proyek galang dana.
2.	Pemrosesan pemberian dana pertengahan	59837	Merupakan pemrosesan pemberian dana oleh seorang <i>funder</i> pada suatu proyek galang dana, yang sebelumnya telah menerima

			pemberian dana, namun belum menyebabkan pencapaian target dana yang ditetapkan pada proyek.
3.	Pemrosesan pemberian dana akhir	76769	Merupakan pemrosesan pemberian dana oleh seorang <i>funder</i> pada suatu proyek galang dana, yang sebelumnya telah menerima pemberian dana, sehingga menyebabkan pencapaian target dana yang ditetapkan pada proyek.

Ketiga kondisi pemrosesan pada Tabel 4.7 memiliki nilai *gasUsed* yang berbeda. Hal tersebut terjadi karena adanya perbedaan proses komputasi pada *function funding* (pemberian dana) dalam *smart contract* sistem yang dieksekusi oleh *miner*. Ketiga kondisi pemrosesan ini pada dasarnya memiliki *function* yang sama pada *smart contract* yang dibangun. Namun pada pemrosesan pemberian dana awal dan akhir terdapat komputasi perubahan nilai pada beberapa *variable* dan *function* lain yang juga terkomputasi sehingga menyebabkan adanya pemakaian *gas* lebih dalam eksekusinya.

Pengujian besar biaya transaksi (*transactionFee*) untuk eksekusi pemrosesan ini dilakukan dengan menggunakan nilai *gasUsed* dari ketiga kondisi pemrosesan Tabel 4.7, dan parameter *gasPrice* yang bernilai 1, 10, 25, 33 (Rata-rata nilai *gasPrice* Ethereum pada waktu studi dilakukan), 50, 75, 100 dan 128. Hasil pengujian ini dapat dilihat pada Gambar 4.19 dan tabel data Lampiran 4.



Gambar 4. 19 Besar Biaya Transaksi (*transactionFee*) Eksekusi Pemrosesan Pemberian Dana pada Proyek dalam *Ether*

Pada kondisi *gasPrice* yang bernilai 1 – 128 Gwei, besar biaya transaksi untuk permrosesan pemberian dana pada proyek berkisar antara 0.000089837 – 0.011499136 Ether (pemberian dana awal), 0.000059837 – 0.007659136 Ether (pemberian dana pertengahan), dan 0.000076769 – 0.009826432 Ether (pemberian dana akhir). Jika dikonversikan ke dalam Rupiah (konversi 1:3451991.21), besar biaya transaksi tersebut beriksar antara Rp310,00 – Rp39.694,00 (pemberian dana awal), Rp206,00 – Rp26.439,00 (pemberian dana pertengahan), dan Rp265,00 – Rp33.920,00 (pemberian dana akhir).

Rata-rata biaya transaksi untuk eksekusi pemrosesan pemberian dana pada proyek berkisar antara 0.000075481 – 0.009661568 Ether atau Rp260,00 – Rp33.351,00.

4.4.3. Evaluasi Hasil Pengujian Besar Biaya Pemrosesan

Biaya transaksi yang ada dalam sistem *crowdfunding* yang dibangun menggunakan *platform* Ethereum untuk implementasi *blockchain* dengan integrasi *smart contract* pada dasarnya merupakan biaya operasional yang dibutuhkan untuk mengeksekusi permrosesan yang ada, yaitu pemrosesan pembentukan proyek galang dana dan pemrosesan pemberian dana pada proyek galang dana. Seluruh pemrosesan ini dieksekusi oleh *node miner* yang tergabung dalam jaringan Ethereum. Biaya transaksi ini akan dibayar oleh pengguna sistem sebagai pengirim *request* pemrosesan dan diberikan kepada *miner* sebagai hadiah karena telah mengeksekusi pemrosesan yang diminta. Dengan demikian, kebutuhan pembangunan dan pengoperasian infrastuktur dalam menjalankan pemrosesan yang ada tidak diperlukan.

Jika dibandingkan dengan sistem *crowdfunding* yang ada saat ini, seperti Kickstarter, GoFundMe, dan Kitabisa, ketiga sistem tersebut memiliki biaya operasional yang dibebankan kepada penggunanya untuk memenuhi kebutuhan penyediaan layanannya. Setiap pemrosesan yang ada dalam sistem tersebut dieksekusi dengan menggunakan infrastruktur yang dioperasikan oleh masing-masing pengelola sistem, sehingga terdapat beban biaya yang diperlukan dalam proses pengoperasiannya.

Adapun jenis dan besar pembebanan biaya pada ketiga sistem *crowdfunding* tersebut dapat dilihat pada Tabel 4.8.

Tabel 4. 8 Pembebanan Biaya pada Beberapa Sistem *Crowdfunding*

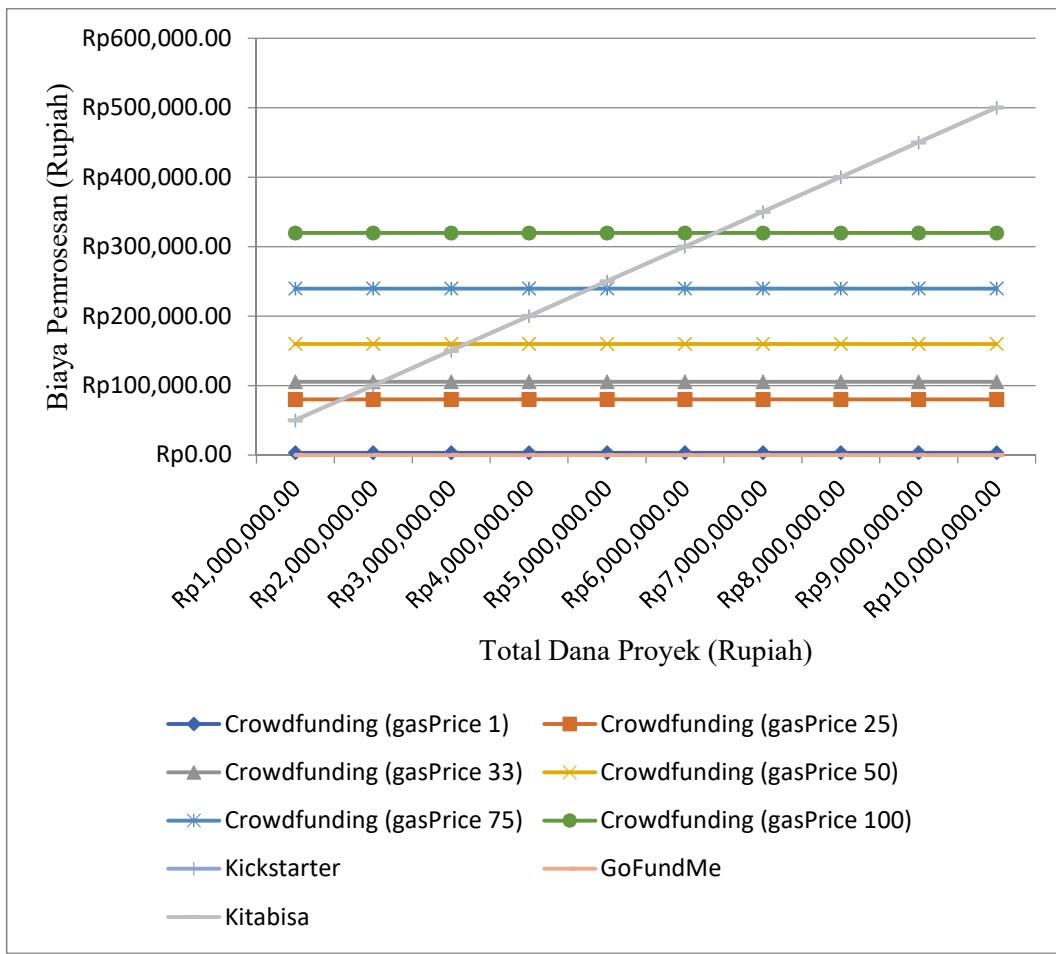
No.	Sistem	Jenis Biaya

1.	Kickstarter	<i>Platform fee</i> (Per proyek) : 5% dari total dana yang diterima. <i>Payment processing fee</i> (Per pemberian dana) : 3% dari dana yang diberikan + \$0.20 / Rp3.000,00
2.	GoFundMe	<i>Platform fee</i> (Per proyek) : 0% total dana yang diterima. <i>Payment processing fee</i> (Per pemberian dana) : 2.9% dari dana yang diberikan + \$0.30 / Rp4.500,00
3.	Kitabisa	<i>Platform fee</i> (Per proyek) : 5% total dana yang diterima. <i>Payment processing fee</i> (Per pemberian dana) : 1,5% dari dana yang diberikan + Rp1.650,00

Pada Tabel 4.8, jenis biaya *platform fee* pada dasarnya merupakan biaya untuk membentuk suatu proyek galang dana pada sistem *crowdfunding* Kickstarter, GoFundMe dan Kitabisa. Sementara untuk *payment processing fee* merupakan biaya untuk pemrosesan pemberian dana pada suatu proyek galang dana dalam ketiga sistem tersebut. Besar biaya *platform fee* ini akan bergantung pada jumlah total dana yang dikumpulkan proyek dan besar biaya *payment processing fee* ini akan bergantung pada jumlah dana yang diberikan pada suatu proyek. Dengan kata lain, biaya pemrosesan yang ditetapkan pada ketiga sistem *crowdfunding* ini memiliki tarif berbasis persentase berdasarkan jumlah dana yang diterima oleh proyek galang dana.

Pada sistem *crowdfunding* yang dibangun, besar biaya yang dibutuhkan dalam pemrosesan pembentukan proyek galang dana dan pemrosesan pemberian dana memiliki nilai yang tidak bergantung pada jumlah dana yang diterima oleh proyek galang dana, melainkan hanya akan bergantung pada kondisi *gasPrice* yang dapat diterima pada jaringan Ethereum. Perhitungan kedua biaya pemrosesan ini dilakukan menggunakan tarif *gasPrice* Ethereum yang bersifat sama rata (*flat rate*) terlepas dari jumlah dana yang diterima oleh proyek galang dana. Berdasarkan hasil pengujian yang dilakukan dengan menggunakan kondisi *gasPrice* bernilai 1 – 128 Gwei, besar biaya pemrosesan minimum untuk pembentukan proyek galang dana berkisar antara Rp3.194,00 – Rp408.896,00 dan rata-rata besar biaya untuk pemrosesan pemberian dana pada proyek galang dana berkisar antara Rp260,00 – Rp33.351,00.

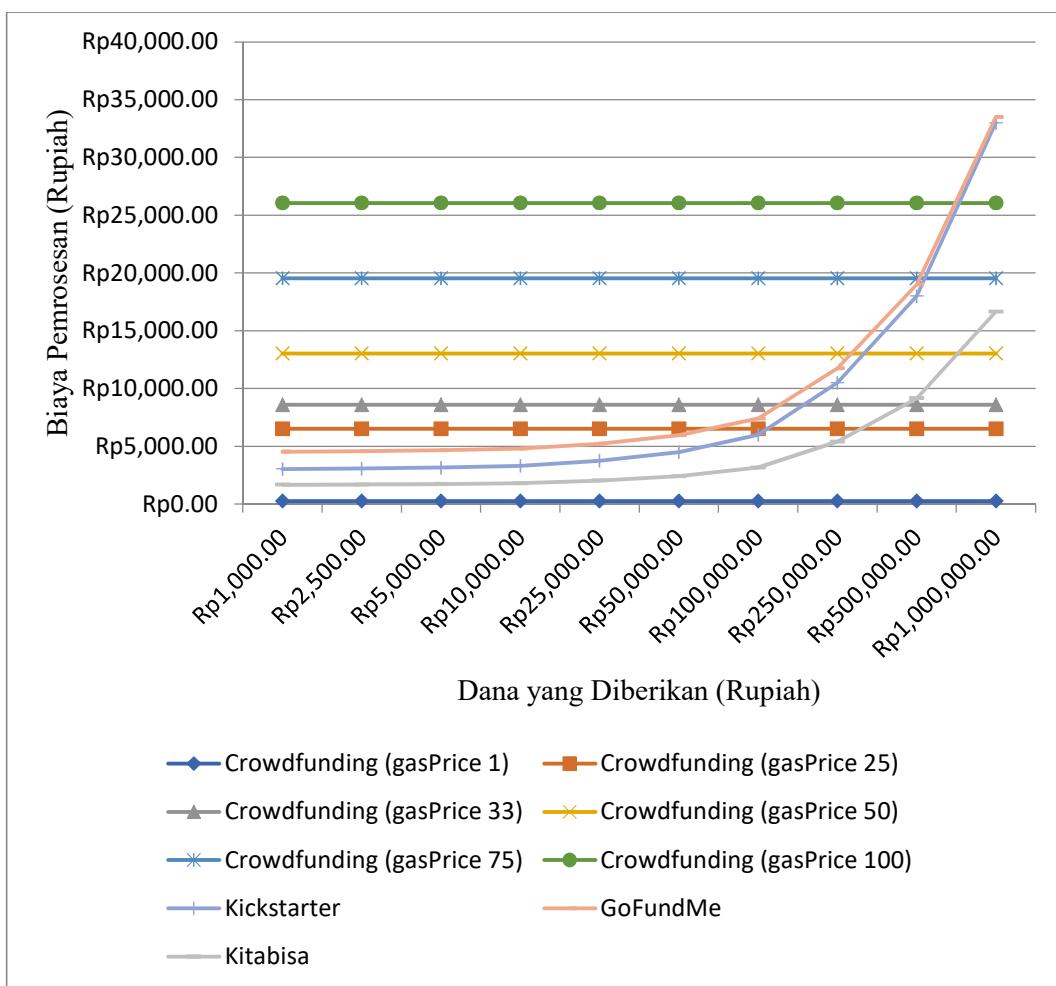
Perbandingan biaya pemrosesan pembentukan proyek galang dana antara sistem *crowdfunding* yang dibangun (*transactionFee*) dengan beberapa kondisi nilai *gasPrice*, termasuk nilai rata-rata *gasPrice* Ethereum saat studi dilakukan (33 Gwei), dan sistem *crowdfunding* yang ada saat ini seperti Kickstarter, GoFundMe ataupun Kitabisa (*platform fee*) dapat dilihat pada Gambar 4.20 dan tabel data Lampiran 5.



Gambar 4. 20 Perbandingan Biaya Pemrosesan Pembentukan Proyek Galang Dana

Berdasarkan Gambar 4.20 dan tabel data Lampiran 5, dapat diketahui bahwa pada kondisi nilai *gasPrice* sebesar 33 Gwei (nilai rata-rata *gasPrice* Ethereum), sistem *crowdfunding* yang dibangun memiliki biaya pemrosesan pembentukan proyek galang dana yang lebih optimal untuk total dana proyek yang besarnya sekitar Rp2.000.000,00 atau lebih jika dibandingkan dengan sistem *crowdfunding* Kickstarter dan Kitabisa. Sementara itu, pada sistem *crowdfunding* GoFundMe, tidak terdapat biaya untuk pemrosesan pembentukan proyek galang dana.

Adapun perbandingan biaya pemrosesan pemberian dana pada proyek galang dana antara sistem *crowdfunding* yang dibangun (*transactionFee*) dengan beberapa kondisi nilai *gasPrice*, termasuk nilai rata-rata *gasPrice* Ethereum saat studi dilakukan (33 Gwei), dan sistem *crowdfunding* yang ada saat ini seperti Kickstarter, GoFundMe ataupun Kitabisa (*payment processing fee*) dapat dilihat pada Gambar 4.21 dan tabel data Lampiran 5.



Gambar 4. 21 Perbandingan Biaya Pemrosesan Pemberian Dana pada Proyek Galang Dana

Berdasarkan Gambar 4.21 dan tabel data Lampiran 5, dapat diketahui bahwa pada kondisi nilai *gasPrice* sebesar 33 Gwei (nilai rata-rata *gasPrice* Ethereum), sistem *crowdfunding* yang dibangun memiliki biaya pemrosesan pemberian dana pada proyek galang dana yang lebih optimal untuk jumlah dana yang besarnya sekitar Rp500.000,00 atau lebih jika dibandingkan dengan sistem *crowdfunding* Kickstarter, GoFundMe dan Kitabisa.

Biaya pemrosesan pembentukan proyek galang dana ataupun pemberian dana yang ada pada sistem *crowdfunding* yang dibangun pada dasarnya memiliki nilai yang lebih optimal untuk kondisi jumlah dana proyek yang bernilai besar jika dibandingkan dengan sistem *crowdfunding* yang ada saat ini. Hal ini dikarenakan penetapan biaya pemrosesan pada sistem *crowdfunding* yang dibangun dilakukan berdasarkan tarif *gasPrice* Ethereum yang bersifat sama rata (*flat rate*) terlepas dari jumlah dana yang diterima oleh proyek galang dana. Pada kondisi dana yang bernilai kecil, pengoptimalan biaya pemrosesan masih dapat tercapai walaupun akan sangat bergantung pada penggunaan *gas* dan kondisi dari *gasPrice* yang dapat diterima oleh jaringan Ethereum.

BAB V

PENUTUP

5.1. Kesimpulan

Dalam tugas akhir ini, telah berhasil dibangun sebuah sistem *crowdfunding* berbasis *blockchain* dengan integrasi *smart contract*. Implementasi teknologi *blockchain* dengan integrasi *smart contract* dilakukan dengan menggunakan *platform blockchain* Ethereum.

Implementasi *blockchain* mampu meningkatkan properti keamanan dan transparansi kegiatan pemberian dana yang terjadi pada setiap proyek galang dana dalam sistem *crowdfunding* yang dibangun jika dibandingkan dengan berbagai sistem *crowdfunding* yang ada saat ini. Arsitektur terdesentralisasi, aksesibilitas data yang bersifat publik dan protokol-protokol khusus pada proses pencatatan data transaksi yang didapatkan dari implementasi *blockchain* merupakan penyebab terjadinya peningkatan pada properti tersebut.

Selain analisis keamanan dan transparansi, berhasil ditunjukkan bahwa integrasi *smart contract* dapat mengoptimalkan biaya pemrosesan pembentukan proyek galang dana (*platform fee*) dan pemberian dana (*payment processing fee*) yang umumnya diterapkan dengan tarif berbasis persentase berdasarkan jumlah dana yang diterima oleh proyek galang dana pada berbagai sistem *crowdfunding* konvensional yang ada saat ini. Setiap pemrosesan pembentukan proyek galang dana dan pemberian dana dalam sistem *crowdfunding* yang dibangun dilakukan secara otomatis menggunakan *smart contract* Ethereum dengan biaya pemrosesan yang ditetapkan berdasarkan tarif sama rata (*flat rate*) terlepas dari jumlah dana

yang diterima oleh proyek galang dana. Hasil perbandingan yang dilakukan menunjukkan bahwa pada sistem *crowdfunding* yang dibangun, dengan kondisi nilai *gasPrice* sebesar 33 Gwei (nilai rata-rata *gasPrice* Ethereum saat studi dilakukan), pengoptimalan biaya pemrosesan pembentukan proyek galang dana tercapai untuk total dana yang besarnya sekitar Rp2.000.000,00 atau lebih dan pengoptimalan biaya pemrosesan pemberian dana tercapai untuk jumlah pemberian dana yang besarnya sekitar Rp500.000,00 atau lebih.

5.2. Saran

Saran untuk penelitian selanjutnya :

- *Smart contract* sistem dikembangkan dengan menggunakan bahasa pemrograman Solidity, sehingga kedepannya dapat dilakukan perancangan sistem menggunakan bahasa pemrograman yang lain seperti Viper.
- Berbagai macam *service* dapat diintegrasikan dengan Ethereum untuk mengembangkan sistem dengan skala yang lebih besar.
- *Platform blockchain* lain dapat digunakan sebagai alternatif dari Ethereum.

DAFTAR PUSTAKA

- Abbas, Qalab E, dan Jang Sung-Bong. "A Survey of Blockchain and Its Applications." *International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, 2019.
- Belleflamme, Paul, Thomas Lambert, dan Armin Schwienbacher. "Crowdfunding: Tapping The Right Crowd." *Journal of Business Venturing*, 2013: 25.
- Ethereum Community. "Ethereum Whitepaper." *Ethereum Web site*. 5 Juni 2020. <https://ethereum.org/whitepaper> (diakses Juni 25, 2020).
- Fakhri, Dinan, dan Kusprasapta Mutijarsa. "Secure IoT Communication using Blockchain Technology." *International Symposium on Electronics and Smart Devices (ISESD)*. 2018.
- Gao, Weichao, William G. Hatcher, dan Wei Yu. "A Survey of Blockchain: Techniques, Applications, and Challenges." *2018 27th International Conference on Computer Communication and Networks (ICCCN)*. Hangzhou, China: IEEE, 2018.
- Gardiner, Ross. "Free Trade: Composable Smart Contracts." Dissertation, Bristol, 2018.
- Hileman, Garrick, dan Michel Rauchs. "Global Blockchain Benchmarking Study." 2017.
- Islam, Mohammad Rabiul, Imad Fakhri Al-Shaikhli, Rizal Mohd Nor, dan Kabir Sardar Mohammad. "Cryptocurrency vs Fiat Currency: Architecture, Algorithm, Cashflow & Ledger Technology on Emerging Economy." *2018 International Conference on Information and Communication Technology for the Muslim World (ICT4M)*. 2018.
- Jani, Shailak. "Smart Contracts: Building Blocks for Digital Transformation." Thesis, New-Delhi, India, 2020.
- Kasireddy, Preethi. "How does Ethereum work, anyway?" *Medium Corporation Web site*. 27 September 2017. <https://medium.com/@preethikasireddy/how-does-ethereum-work-anyway-22d1df506369> (diakses Juni 26, 2020).
- Kaushik, Akanksha, Archana Choudhary, Chinmay Ektare, dan Deepti Thomas. "Blockchain – Literature Survey." *2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*. India, 2017.
- Kim, Kiyun. "Modified Merkle Patricia Trie — How Ethereum saves a state." *Medium Corporation Web site*. 26 Juni 2018.

- <https://medium.com/codechain/modified-merkle-patricia-trie-how-ethereum-saves-a-state-e6d7555078dd> (diakses Juni 2020).
- Lin, Iuon-Chang, dan Tzu-Chun Liao. “A Survey of Blockchain Security Issues and Challenges.” *International Journal of Network Security*, Vol.19, 2017.
- Nakamoto, Satoshi. “Bitcoin: A Peer-to-peer Electronic Cash System.” *Bitcoin Organization Web site*. 2008. <https://nakamotoinstitute.org/bitcoin/>.
- Rennock, Michael J.W., Alan Cohn, dan Jared R. Butcher. “BlockChain Technology.” 2018.
<https://www.steptoe.com/images/content/1/7/v3/171269/LIT-FebMar18-Feature-Blockchain.pdf> (diakses Maret 16, 2020).
- Sarmah, Simanta Shekhar. “Understanding Blockchain Technology.” 2018.
- Sayeed, Sarwar, Hector Marco-Gisbert, dan Tom Caira. “Smart Contract: Attacks and Protections.” *IEEE Access*, 2020.
- Shrivastava, Mahendra Kumar, dan Thomas Yeboah. “The Disruptive Blockchain: Types, Platforms and Applications.” *5th Texila World Conference for Scholars (TWCS)*. 2018.
- Singla, Vinayak, Indra Kumar Malav, Jaspreet Kaur, dan Sumit Kalra. “Develop Leave Application using Blockchain Smart Contract.” *11th International Conference on Communication Systems & Networks (COMSNETS)*. 2019.
- Szabo, Nick. “Smart Contracts: Building Blocks for Digital Markets.” 1996.
https://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smarty_contracts_2.html (diakses Juni 13, 2020).
- Taş, Ruhi, dan Ömer Özgür Tanrıöver. “Building A Decentralized Application on the Ethereum Blockchain.” *2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*. Ankara, Turkey: IEEE, 2019.
- Vujičić, Dejan, Dijana Jagodić, dan Siniša Randić. “Blockchain Technology, Bitcoin, and Ethereum: A Brief Overview.” *17th International Symposium INFOTEH-JAHORINA (INFOTEH)*. Čačak, Serbia, 2018.
- Wood, Gavin. *Ethereum: A Secure Decentralised Generalised Transaction Ledger*. 20 Oktober 2019.

LAMPIRAN

LAMPIRAN 1

- Tampilan halaman *New Project*

The screenshot shows the 'New Project' form. At the top, there is a header bar with the title 'Eth-Crowdfunding' and navigation links for 'New Project', 'Project List', and a project ID '0xb4831F5F23604a3C2066c777Af5ae0CfEeb1c752'. The main form area has fields for 'Title', 'Fund Goal' (with a note 'In Ether Currency. Example : 2 Eth'), 'Funding Duration' (with a note 'Days'), and 'Description'. A 'SUBMIT ➤' button is located at the bottom right of the form.

- Tampilan halaman *Project List*

The screenshot shows the 'Project List' page. At the top, there is a header bar with the title 'Eth-Crowdfunding' and navigation links for 'New Project', 'Project List', and a project ID '0xb4831F5F23604a3C2066c777Af5ae0CfEeb1c752'. Below the header, three projects are listed in cards:

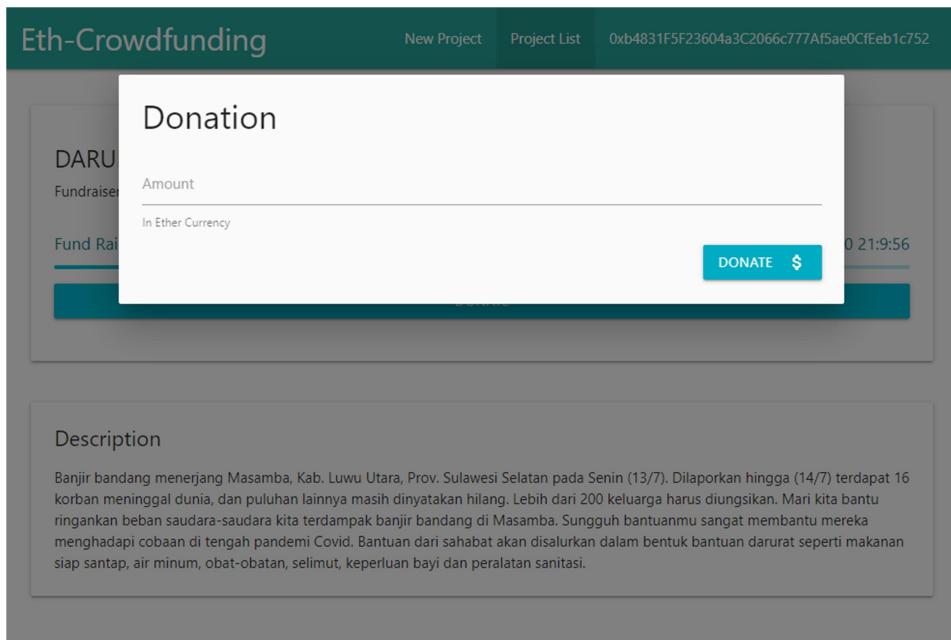
- Selamatkan Nyawa Sesama!...**
Penyebaran virus corona di Indonesia terus meluas. Dampak virus ini sangat mengkhawatirkan:...
FAILED [DETAIL](#)
- Lawan Corona: Sembako Untuk Peker...**
Seorang pengemudi ojek online di daerah Sunter Jakarta Utara bernama Pak Aji harus berjuang di...
SUCCESSFUL [DETAIL](#)
- Sedekah Santri Penghafal Al-Quran...**
Yuk Ayahanda Bunda sedekahkan sebagian hartanya untuk kebutuhan pendidikan dan pangan santri...
SUCCESSFUL [DETAIL](#)

- Tampilan halaman *Project Detail* (Status *Ongoing*)



Description

Banjir bandang menerjang Masamba, Kab. Luwu Utara, Prov. Sulawesi Selatan pada Senin (13/7). Dilaporkan hingga (14/7) terdapat 16 korban meninggal dunia, dan puluhan lainnya masih dinyatakan hilang. Lebih dari 200 keluarga harus diungsikan. Mari kita bantu ringankan beban saudara-saudara kita terdampak banjir bandang di Masamba. Sungguh bantuanmu sangat membantu mereka menghadapi cobaan di tengah pandemi Covid. Bantuan dari sahabat akan disalurkan dalam bentuk bantuan darurat seperti makanan siap santap, air minum, obat-obatan, selimut, keperluan bayi dan peralatan sanitasi.



- Tampilan halaman *Project Detail* (Status *Successful*)

The screenshot shows a crowdfunding project titled "Sedekah Santri Penghafal Al-Quran dan Kitab Kuning". The project has raised 0.15 Ether out of a goal of 0.1 Ether. It is marked as "SUCCESSFUL". The funding period ended on July 16, 2020, at 23:25:48. The project description discusses helping underprivileged students.

- Tampilan halaman *Project Detail* (Status *Failed*)

The screenshot shows a crowdfunding project titled "Selamatkan Nyawa Sesama! #BersamaLawanCorona". The project has raised 0.11 Ether out of a goal of 0.5 Ether. It is marked as "FAILED". The funding period ended on July 15, 2020, at 22:53:52. The project description discusses the impact of the COVID-19 pandemic.

LAMPIRAN 2

- Kode *smart contract*

```
1 pragma solidity ^0.5.0;
2
3 import "./SafeMath.sol";
4
5 contract Crowdfunding {
6     using SafeMath for uint256;
7
8     Project[] private projects;
9
10    event ProjectCreated(
11        address projectAddress,
12        address fundraiser,
13        string title,
14        string description,
15        uint256 deadline,
16        uint256 fundGoal
17    );
18
19    function createProject(
20        string calldata title,
21        string calldata description,
22        uint deadline,
23        uint fundGoal
24    ) external {
25        uint deadlineDate = now.add(deadline.mul(1 days));
26        Project _project = new Project(msg.sender, title, description, deadlineDate, fundGoal);
27        projects.push(_project);
28        emit ProjectCreated(address(_project), msg.sender, title, description, deadlineDate, fundGoal);
29    }
30
31    function getProjects() external view returns(Project[] memory) {
32        return projects;
33    }
34}
35
36 contract Project {
37     using SafeMath for uint256;
38
39     // State data
40     enum State {
41         Ongoing,
42         Failed,
43         Successful
44     }
45
46     // Variables
47     address payable public fundraiser;
48     uint public fundGoal;
49     uint public completeAt;
50     uint256 public currentFund;
51     uint256 public fundraised;
52     uint public deadline;
53     string public title;
54     string public description;
55     mapping (address => uint) public funders;
56
57     // Event project received the funds
58     event ProjectFunded(
59         address funder,
60         uint fundAmount,
61         uint currentFund
62     );
63
64     // Event fundraiser received the funds
65     event FundraiserPaid(
66         address fundraiser
67     );
68
69     // Modifier to check if the function caller is the project creator
70     modifier isFundraiser() {
71         require(msg.sender == fundraiser);
72         -
73     }
74
75     constructor(
76         address payable projectFundraiser,
77         string memory projectTitle,
78         string memory projectDescription,
79         uint projectDeadline,
80         uint projectFundGoal
81     ) public {
82         fundraiser = projectFundraiser;
83         title = projectTitle;
84         description = projectDescription;
85         deadline = projectDeadline;
86         fundGoal = projectFundGoal;
```

```

87 |     currentFund = 0;
88 |     fundRaised = 0;
89 |
90 |
91 | // Funding process function
92 | function funding() external payable {
93 |     require(now < deadline);
94 |     require(msg.sender != fundraiser);
95 |     funders[msg.sender] = funders[msg.sender].add(msg.value);
96 |     currentFund = currentFund.add(msg.value);
97 |     fundRaised = currentFund;
98 |     emit ProjectFunded(msg.sender, msg.value, currentFund);
99 |     if (currentFund >= fundGoal) {
100 |         giveFund();
101 |         completeAt = now;
102 |     }
103 |
104 |
105 | // Give funds to fundraiser
106 | function giveFund() internal returns (bool) {
107 |     uint256 totalFund = currentFund;
108 |     currentFund = 0;
109 |
110 |     if (fundraiser.send(totalFund)) {
111 |         emit FundraiserPaid(fundraiser);
112 |         return true;
113 |     } else {
114 |         currentFund = totalFund;
115 |     }
116 |
117 |     return false;
118 | }
119 |
120 | // Retrieve funds to funders
121 | function getRefund() public returns (bool) {
122 |     require(funders[msg.sender] > 0);
123 |
124 |     uint amountToRefund = funders[msg.sender];
125 |     funders[msg.sender] = 0;
126 |
127 |     if (!msg.sender.send(amountToRefund)) {
128 |         funders[msg.sender] = amountToRefund;
129 |         return false;
130 |     } else {
131 |         currentFund = currentFund.sub(amountToRefund);
132 |     }
133 |
134 |     return true;
135 | }
136 |
137 | // Get project details
138 | function projectDetails() public view returns (
139 |     address payable projectFundraiser,
140 |     string memory projectTitle,
141 |     string memory projectDesc,
142 |     uint256 projectDeadline,
143 |     uint256 projectCurrentFund,
144 |     uint256 projectFundRaised,
145 |     uint256 projectFundGoal,
146 |     State currentState
147 | ) {
148 |     projectFundraiser = fundraiser;
149 |     projectTitle = title;
150 |     projectDesc = description;
151 |     projectDeadline = deadline;
152 |     projectCurrentFund = currentFund;
153 |     projectFundRaised = fundRaised;
154 |     projectFundGoal = fundGoal;
155 |     if (fundRaised >= fundGoal) {
156 |         currentState = State.Successful;
157 |     } else {
158 |         if (now < deadline) {
159 |             currentState = State.Ongoing;
160 |         } else {
161 |             currentState = State.Failed;
162 |         }
163 |     }
164 | }
165 |

```

LAMPIRAN 3

- Tabel hasil pengujian perhitungan biaya transaksi minimum (*transactionFee*) eksekusi pemrosesan pembentukan proyek

No.	<i>gasPrice</i> (Gwei)	<i>gasUsed</i>	<i>transactionFee</i> (Ether)	<i>transactionFee</i> (Rupiah)
1.	1	925408	0.000925408	3194.500282
2.	10	925408	0.00925408	31945.00282
3.	25	925408	0.0231352	79862.50704
4.	50	925408	0.0462704	159725.0141
5.	75	925408	0.0694056	239587.5211
6.	100	925408	0.0925408	319450.0282
7.	128	925408	0.118452224	408896.0361

LAMPIRAN 4

- Tabel hasil pengujian perhitungan biaya transaksi (*transactionFee*) eksekusi pemrosesan pemberian dana pada proyek untuk kondisi pemberian dana awal

No.	<i>gasPrice</i> (Gwei)	<i>gasUsed</i>	<i>transactionFee</i> (Ether)	<i>transactionFee</i> (Rupiah)
1.	1	89837	0.000089837	310.1165343
2.	10	89837	0.00089837	3101.165343
3.	25	89837	0.002245925	7752.913358
4.	50	89837	0.00449185	15505.82672
5.	75	89837	0.006737775	23258.74007
6.	100	89837	0.0089837	31011.65343
7.	128	89837	0.011499136	39694.91639

- Tabel hasil pengujian perhitungan biaya transaksi (*transactionFee*) eksekusi pemrosesan pemberian dana pada proyek untuk kondisi pemberian dana pertengahan

No.	<i>gasPrice</i> (Gwei)	<i>gasUsed</i>	<i>transactionFee</i> (Ether)	<i>transactionFee</i> (Rupiah)
1.	1	59837	0.000059837	206.556798
2.	10	59837	0.00059837	2065.56798
3.	25	59837	0.001495925	5163.919951
4.	50	59837	0.00299185	10327.8399
5.	75	59837	0.004487775	15491.75985
6.	100	59837	0.0059837	20655.6798

7.	128	59837	0.007659136	26439.27015
----	-----	-------	-------------	-------------

- Tabel hasil pengujian perhitungan biaya transaksi (*transactionFee*) eksekusi pemrosesan pemberian dana pada proyek untuk kondisi pemberian dana akhir

No.	<i>gasPrice</i> (Gwei)	<i>gasUsed</i>	<i>transactionFee</i> (Ether)	<i>transactionFee</i> (Rupiah)
1.	1	76769	0.000076769	265.0059132
2.	10	76769	0.00076769	2650.059132
3.	25	76769	0.001919225	6625.14783
4.	50	76769	0.00383845	13250.29566
5.	75	76769	0.005757675	19875.44349
6.	100	76769	0.0076769	26500.59132
7.	128	76769	0.009826432	33920.75689

- Tabel perhitungan rata-rata biaya transaksi (*transactionFee*) eksekusi pemrosesan pemberian dana pada proyek

No.	<i>gasPrice</i> (Gwei)	<i>gasUsed</i>	<i>transactionFee</i> (Ether)	<i>transactionFee</i> (Rupiah)
1.	1	75481	0.000075481	260.5597485
2.	10	75481	0.00075481	2605.597485
3.	25	75481	0.001887025	6513.993713
4.	50	75481	0.00377405	13027.98743
5.	75	75481	0.005661075	19541.98114
6.	100	75481	0.0075481	26055.97485

7.	128	75481	0.009661568	33351.64781
----	-----	-------	-------------	-------------

LAMPIRAN 5

- Tabel perbandingan biaya pemrosesan pembentukan proyek galang dana untuk beberapa kondisi total dana proyek antara sistem *crowdfunding* yang dibangun (kondisi *gasPrice* 10, 25, 33, 50, 75 dan 100 Gwei) dan sistem *crowdfunding* yang ada saat ini (Kickstarter, GoFundMe dan Kitabisa)

No .	Total Dana Proyek (Rupiah)	Biaya Pemrosesan Pembentukan Proyek Galang Dana (Rupiah)								
		Sistem Crowdfunding yang Dibangun						Kickstart er (5% Total Dana)	GoFund Me (0% Total Dana)	Kitabisa (5% Total Dana)
		<i>gasPrice</i> 1 Gwei	<i>gasPrice</i> 25 Gwei	<i>gasPrice</i> 33 Gwei	<i>gasPrice</i> 50 Gwei	<i>gasPrice</i> 75 Gwei	<i>gasPrice</i> 100 Gwei			
1.	1000000	3194.5002 82	79862.507 04	105418.50 93	159725.01 41	239587.52 11	319450.028 2	50000	0	50000
2.	2000000	3194.5002 82	79862.507 04	105418.50 93	159725.01 41	239587.52 11	319450.028 2	100000	0	100000
3.	3000000	3194.5002 82	79862.507 04	105418.50 93	159725.01 41	239587.52 11	319450.028 2	150000	0	150000

4.	4000000	3194.5002	79862.507	105418.50	159725.01	239587.52	319450.028	200000	0	200000
5.	5000000	3194.5002	79862.507	105418.50	159725.01	239587.52	319450.028	250000	0	250000
6.	6000000	3194.5002	79862.507	105418.50	159725.01	239587.52	319450.028	300000	0	300000
7.	7000000	3194.5002	79862.507	105418.50	159725.01	239587.52	319450.028	350000	0	350000
8.	8000000	3194.5002	79862.507	105418.50	159725.01	239587.52	319450.028	400000	0	400000
9.	9000000	3194.5002	79862.507	105418.50	159725.01	239587.52	319450.028	450000	0	450000
10.	10000000	3194.5002	79862.507	105418.50	159725.01	239587.52	319450.028	500000	0	500000

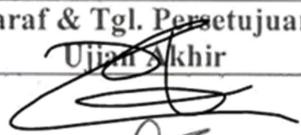
- Tabel perbandingan biaya pemrosesan pemberian dana pada proyek galang dana untuk beberapa kondisi dana yang diberikan antara sistem *crowdfunding* yang dibangun (kondisi *gasPrice* 10, 25, 33, 50, 75 dan 100 Gwei) dan sistem *crowdfunding* yang ada saat ini (Kickstarter, GoFundMe dan Kitabisa)

No .	Dana yang Diberikan (Rupiah)	Biaya Pemrosesan Pemberian Dana pada Proyek Galang Dana (Rupiah)								
		Sistem Crowdfunding yang Dibangun						Kickstart er (3% Dana + 3000)	GoFund Me (2.9% Dana + 4500)	Kitabisa (1.5% Dana + 1650)
		<i>gasPrice</i> 1 Gwei	<i>gasPrice</i> 25 Gwei	<i>gasPrice</i> 33 Gwei	<i>gasPrice</i> 50 Gwei	<i>gasPrice</i> 75 Gwei	<i>gasPrice</i> 100 Gwei			
1.	1000	260.55974	6513.9937	8598.4717	13027.987	19541.981	26055.9748	3030	4529	1665
	85	13	01	43	14	5				
2.	2500	260.55974	6513.9937	8598.4717	13027.987	19541.981	26055.9748	3075	4572.5	1687.5
	85	13	01	43	14	5				
3.	5000	260.55974	6513.9937	8598.4717	13027.987	19541.981	26055.9748	3150	4645	1725
	85	13	01	43	14	5				
4.	10000	260.55974	6513.9937	8598.4717	13027.987	19541.981	26055.9748	3300	4790	1800

		85	13	01	43	14	5			
5.	25000	260.55974	6513.9937	8598.4717	13027.987	19541.981	26055.9748	3750	5225	2025
		85	13	01	43	14	5			
6.	50000	260.55974	6513.9937	8598.4717	13027.987	19541.981	26055.9748	4500	5950	2400
		85	13	01	43	14	5			
7.	100000	260.55974	6513.9937	8598.4717	13027.987	19541.981	26055.9748	6000	7400	3150
		85	13	01	43	14	5			
8.	250000	260.55974	6513.9937	8598.4717	13027.987	19541.981	26055.9748	10500	11750	5400
		85	13	01	43	14	5			
9.	500000	260.55974	6513.9937	8598.4717	13027.987	19541.981	26055.9748	18000	19000	9150
		85	13	01	43	14	5			
10.	1000000	260.55974	6513.9937	8598.4717	13027.987	19541.981	26055.9748	33000	33500	16650
		85	13	01	43	14	5			

KARTU BIMBINGAN SKRIPSI
 Departemen S1 Teknik Informatika Universitas Hasanuddin

Stambuk	Nama Mahasiswa
D42115304	Fiqar Aprialim

Pembimbing	Nama Pembimbing	Paraf & Tgl. Persetujuan Ujian Akhir
I	Adnan, S.T., M.T., Ph.D.	
II	Dr. Eng. Ady Wahyudi Paundu, S.T., M.T.	

No. SK Pembimbing :

Judul Skripsi :	Penerapan <i>Blockchain</i> dengan Integrasi <i>Smart Contract</i> pada Sistem <i>Crowdfunding</i>
-----------------	----------------------------------------------------------------------------------------------------

No.	Tanggal Bimbingan	Uraian Kegiatan Bimbingan	Paraf
1	4 Maret 2020	Konsultasi dengan Pembimbing II terkait arsitektur sistem <i>crowdfunding</i> yang dibangun	
2	19 Juni 2020	Konsultasi dengan Pembimbing II terkait skema pengujian dan <i>progress</i> pengembangan sistem	
3	30 Juni 2020	Menunjukkan sistem <i>crowdfunding</i> yang dibangun kepada Pembimbing II	
4	28 Juli 2020	Konsultasi dengan Pembimbing II mengenai penulisan dan pengujian sistem	
5	3 Agustus 2020	Konsultasi dengan Pembimbing I mengenai perbaikan literatur dan pengujian sistem	
6	12 Agustus 2020	Konsultasi dengan Pembimbing II mengenai perbaikan penulisan dan pengujian sistem	
7	29 Agustus 2020	Konsultasi dengan Pembimbing II mengenai perbaikan penulisan	
8	8 September 2020	Konsultasi dengan Pembimbing I mengenai penulisan dan presentasi seminar hasil	

9	28 September 2020	Konsultasi dengan Pembimbing I terkait perbaikan skema pengujian sistem	
10	28 September 2020	Konsultasi dengan Pembimbing II terkait perbaikan skema pengujian sistem	
11	2 November 2020	Konsultasi dengan Pembimbing II mengenai perbaikan hasil pengujian sistem	
12	3 November 2020	Konsultasi dengan Pembimbing I mengenai <i>paper</i> (jurnal)	
13	16 November 2020	Konsultasi dengan Pembimbing I mengenai presentasi ujian skripsi	

LEMBAR PERBAIKAN SKRIPSI

"PENERAPAN BLOCKCHAIN DENGAN INTEGRASI SMART CONTRACT PADA SISTEM CROWDFUNDING"

OLEH:

FIQAR APRIALIM
D421 15 304

Skripsi ini telah dipertahankan pada Ujian Akhir Sarjana tanggal 2 Desember 2020. Telah dilakukan perbaikan penulisan dan isi skripsi berdasarkan usulan dari penguji dan pembimbing skripsi.

Persetujuan perbaikan oleh tim penguji:

	Nama	Tanda Tangan
Ketua	Adnan, S.T., M.T., Ph.D.	
Sekretaris	Dr.Eng. Ady Wahyudi Paundu, S.T., M.T.	
Anggota	Dr. Amil Ahmad Ilham, S.T., M.IT.	
	Iqra Aswad, S.T., M.T.	

Persetujuan perbaikan oleh pembimbing:

Pembimbing	Nama	Tanda Tangan
I	Adnan, S.T., M.T., Ph.D.	
II	Dr.Eng. Ady Wahyudi Paundu, S.T., M.T.	