BAB 3
METODOLOGI PENELITIAN

3.1 Metodologi Pengerjaan

Penelitian ini dilakukan dengan menggunakan metodologi *Software Development Life Cycle* (SDLC). Metodologi ini terdiri dari beberapa bagian yaitu:

1. Perencanaan

   Pada bagian ini dilakukan perancangan terhadap sistem yang ingin dibuat.

2. Analisa

   Bagian ini melakukan analisa terhadap kebutuhan sistem yang telah dirancang.

3. Desain

   Bagian ini melakukan desain bagaimana sistem akan ditampilkan.

4. Implementasi

   Bagian ini melakukan implementasi terhadap sistem yang telah dirancang dan didesain.

5. Integrasi dan Pengujian

   Bagian ini melakukan peluncuran serta pengujian terhadap sistem yang telah di implementasi.

6. Perawatan

   Bagian ini merawat sistem serta menuliskan dokumentasi terhadap sistem yang telah dibuat.

Metodologi ini akan di iterasi jika ada kebutuhan penambahan fungsi atau perubahan *requirement* yang terjadi pada sistem.

3.2 Penetapan Fitur

Untuk menyelesaikan rumusan masalah yang telah ditentukan maka media sosial pada penelitian ini memiliki beberapa fitur utama yang digunakan untuk menjawab rumusan tersebut. Fitur-fitur yang ada pada *decentralized application* media sosial ini adalah:

- Registrasi

  Bagian ini mengharuskan para pengguna media sosial untuk mendaftarkan dirinya terlebih dahulu. Pada bagian ini para pengguna harus memasukkan alias yang digunakan dalam media sosial ini. Alias yang digunakan para pengguna tidaklah unik. perbedaan antara pengguna alias yang sama adalah *wallet address* yang juga ditampilkan pada platform media sosial.

- Tokenomics

  Tokenomics pada media sosial ini bertujuan untuk memberikan hukuman dan insentif bagi para pengguna yang membantu untuk menyaring pos yang muncul ke publik. Adanya *tokenomics* memberikan hadiah bagi para *voters* untuk menjadi dorongan atau motivasi untuk melakukan pekerjaan[20], dalam hal ini melakukan penyaringan. Adanya *tokenomics* juga dapat mendukung sistem *voting* dengan memberikan hukuman yang dapat membuat para *voters* melakukan kinerja dengan baik [21] dalam hal ini tidak melakukan *vote* secara asal. Selain itu *tokenomics* juga membantu untuk pembuat pos dapat mendapatkan insentif agar mendorong para pembuat pos untuk membuat pos yang bermutu. *Tokenomics* dijalankan dengan menggunakan token yang dibuat yang diberi nama *Credibility Token*.

- Pos

  Pengguna yang telah melakukan registrasi dapat melakukan pos yang diseleksi terlebih dahulu. Pos yang dapat dilakukan oleh para pengguna dibatasi pada 160 karakter, hal ini dikarenakan sebuah pesan yang memiliki jumlah karakter lebih dari 160 karakter biasanya tidak ditangkap sepenuhnya oleh para pembaca [22]. Batasan ini juga bertujuan untuk memudahkan para *stakeholder* untuk menangkap pesan dengan cepat dan tanpa adanya bias.

- Voting

  Voting sistem merupakan sebuah cara yang bagus untuk mendapatkan jawaban dari pihak *majority* dari sebuah sistem demokrasi [23]. Fitur *voting* bertujuan untuk para *stakeholder* untuk melakukan *censorship* terhadap konten sebelum konten tersebut dapat dilihat oleh publik. Fitur ini digunakan untuk menyaring konten dikarenakan pos dengan konten negatif memiliki jumlah *dislike* yang lebih banyak dan sebaliknya untuk konten positif [24]. Hasil pos ditentukan dari jumlah setuju atau tidak setuju yang terlebih dahulu

menyentuh angka yang sudah ditentukan sebelumnya (*threshold* dalam kasus ini angka tersebut adalah tiga). Apabila hasil setuju maka pos dapat muncul ke *public page*. Jika hasil *vote* tidak setuju maka content diubah menjadi *empty string* dan pos status dimatikan. Para pengguna dapat melakukan *like* (*vote* setuju) dan *dislike* (*vote* tidak setuju) terhadap pos tersebut.

- Like

  Like dapat dilakukan pada pos yang sudah muncul pada *public page* pengguna dapat melakukan *like* dengan membayarkan satu token. Token tersebut diberikan ke pemilik pos sebanyak 0.9 token. *Burn* dilakukan terhadap 0.1 token. Fungsi ini juga menampilkan pos menjadi urutan pertama. Hal ini membuat para pengguna dapat mendukung pembuat konten secara langsung.

- Hide

  Fungsi *hide* disediakan agar para pengguna yang merasa terganggu atau tidak setuju terhadap konten yang ada pada *public page* dapat menghilangkan pos tersebut dari tampilan. Fitur ini disediakan untuk meningkatkan kenyamanan para pengguna yang memiliki latar belakang adat dan budaya yang berbeda.

- Public Page

  Halaman ini menampilkan pos yang telah disaring oleh para *stakeholder*. Konten yang sudah ada pada halaman ini merupakan konten yang telah dianggap layak oleh komunitas untuk dilihat atau di konsumsi oleh publik.

- Private Page

  Halaman ini menunjukkan pos yang disaring oleh para *stakeholder*. Halaman ini tidak dapat di akses oleh para pengguna biasa. Untuk mendapatkan akses ke halaman ini pengguna harus memiliki 100 *ICredibility Token*. Pada halaman ini pengguna dapat melakukan *voting* untuk melakukan penyaringan konten.

3.3 Perancangan Sistem

Perancangan sistem terdiri dari dua bagian utama yaitu perancangan *smart contract* yang berfungsi sebagai *back-end* dan perancangan *front-end* yang

menghubungkan antara *smart-contract* dengan pengguna. Namun untuk merancang fungsionalitas pada *smart contract* dibutuhkannya skema basis data untuk menggambarkan bagaimana data disimpan pada *blockchain*.

3.3.1 Skema Struktur Data

Penyimpanan data pada *blockchain* melalui *smart contract* dilakukan dengan cara menyimpan data tersebut pada sebuah variabel. Sehingga untuk menyimpan sebuah objek serta propertinya maka basis data yang diperlukan haruslah menggunakan *struct*. Sehingga sebuah tabel pada *database* konvensional digantikan *struct* pada *smart contract*. Berikut *struct* yang diperlukan dalam penelitian ini.

Tabel 3.1. *Struct profile*

Nama	Tipe Data	Deskripsi
uid	uint256	menyimpan userid (uid)
username	string	menyimpan *username*
active	bool	menyimpan status *wallet*
postInteraction	mapping(uint256 $\rightarrow$ bool)	menyimpan status interaksi pos

Struct profile yang digambarkan pada Tabel 3.1 bertujuan untuk menyimpan segala informasi yang berhubungan dengan pengguna itu sendiri.

Tabel 3.2. *Struct post*

Nama	Tipe Data	Deskripsi
postId	uint256	menyimpan id pos
content	string	menyimpan konten pos
date	uint256	menyimpan tanggal pos
owner	address	menyimpan *address* pemilik pos
like	uint256	menyimpan jumlah *like*
dislike	uint256	menyimpan jumlah *dislike*
isActive	bool	menyimpan status pos
Lanjut pada halaman berikutnya		

Tabel 3.2 *Struct Post* (lanjutan)

Nama	Tipe Data	Deskripsi
likers	array of address	menyimpan *address* yang melakukan *like*
dislikers	array of address	menyimpan *address* yang melakukan *dislike*

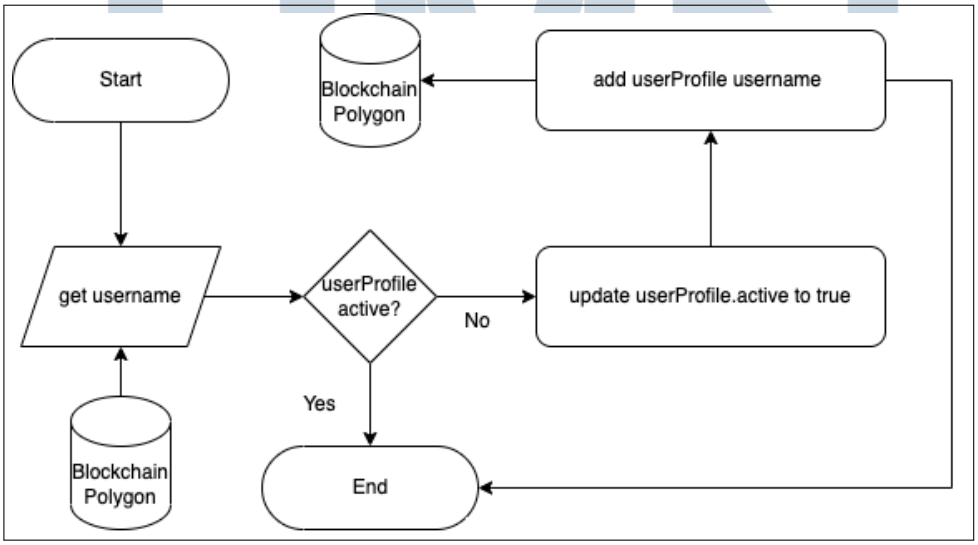
Struct Post yang digambarkan pada Tabel 3.2 menyimpan seluruh informasi pos yang diperlukan untuk berjalannya media sosial.

3.3.2 Smart Contract

Smart contract memiliki beberapa fungsi yang bertujuan untuk menyelesaikan rumusan masalah yang ada. Bagian ini menjelaskan kegunaan dari setiap fungsi yang diperlukan pada *smart contract* berserta dengan diagram alir dari fungsi tersebut.

A Register

Fungsi ini bertujuan untuk mendaftarkan *wallet address* yang belum terdaftar. Pengguna dapat memasukkan *username* yang diinginkan, *username* ini sendiri tidak bersifat unik sehingga dapat lebih dari satu orang.

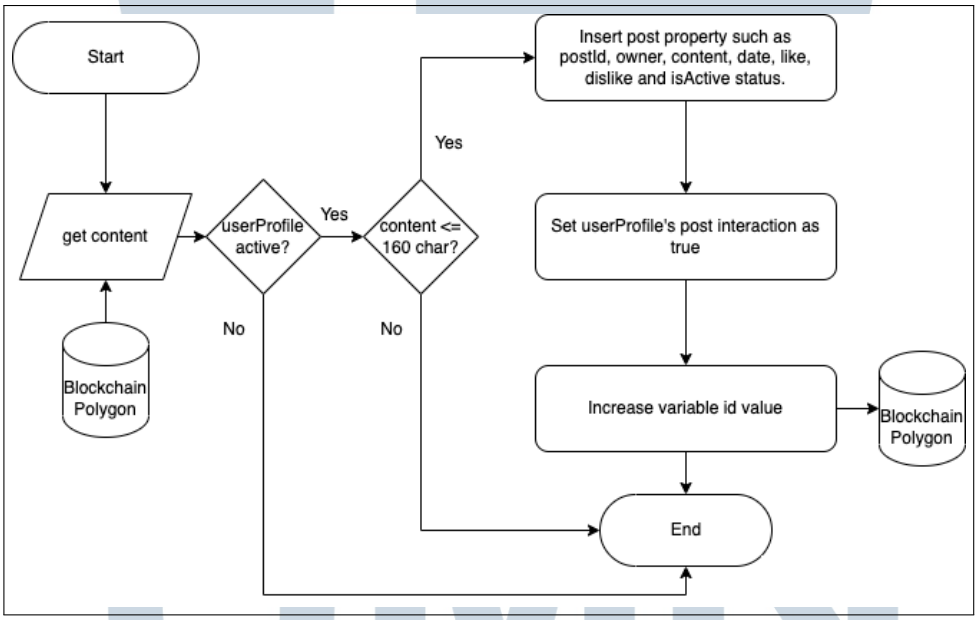


Gambar 3.1. Diagram alir *register*

Pada diagram alir pada Gambar 3.1 dapat terlihat *smart contract* menerima *input username* dari pengguna lalu mengecek apakah pengguna tersebut sudah terdaftar. Apabila pengguna sudah terdaftar maka fungsi tidak dilanjutkan. Apabila pengguna belum mendaftar maka fungsi dijalankan. Fungsi melakukan *update* terhadap profil pengguna serta menetapkan *username* dari pengguna atau *wallet address* tersebut.

B Pos

Fungsi ini bertujuan untuk menunjukkan proses ketika pengguna mencoba untuk melakukan pos. Alur dari proses ini digambarkan pada Gambar 3.2.

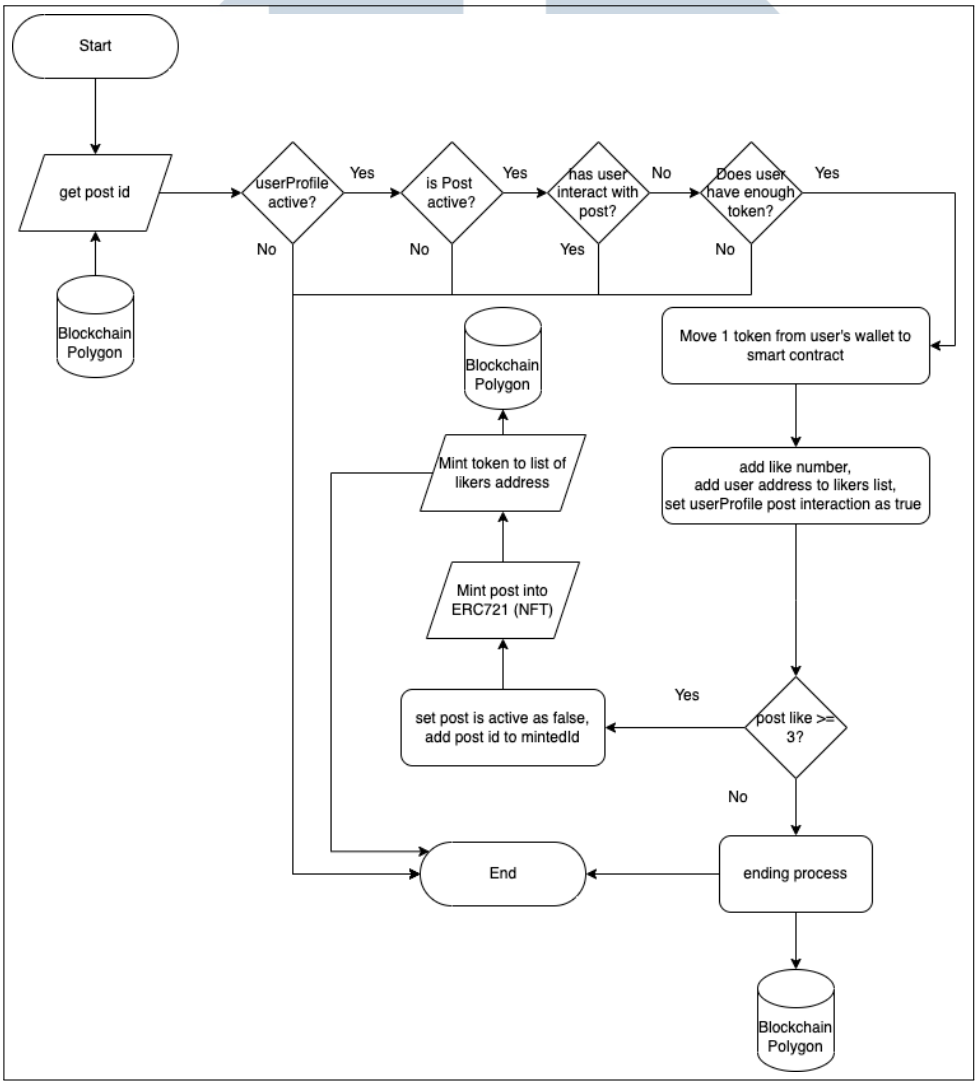


Gambar 3.2. Diagram alir pos

Pertama-tama fungsi menerima *string content* yang menjadi parameter pada fungsi. Setalh itu *smart contract* melakukan pengecekan terhadap status *wallet address* apakah sudah terdaftar atau belum. Apabila sudah maka konten yang telah dibuat pengguna dimasukkan ke dalam *blockchain* beserta dengan properti yang diperlukan. Pos yang telah dibuat pengguna di seleksi pada *private page*. Pada *private page* ditentukan apakah pos tersebut dapat di muncul ke *public page* atau tidak.

C Like (Vote Setuju)

Fungsi ini bertujuan untuk memberikan *vote* setuju terhadap konten yang telah di pos. Alur dari fungsi ini dapat terlihat pada Gambar 3.3.



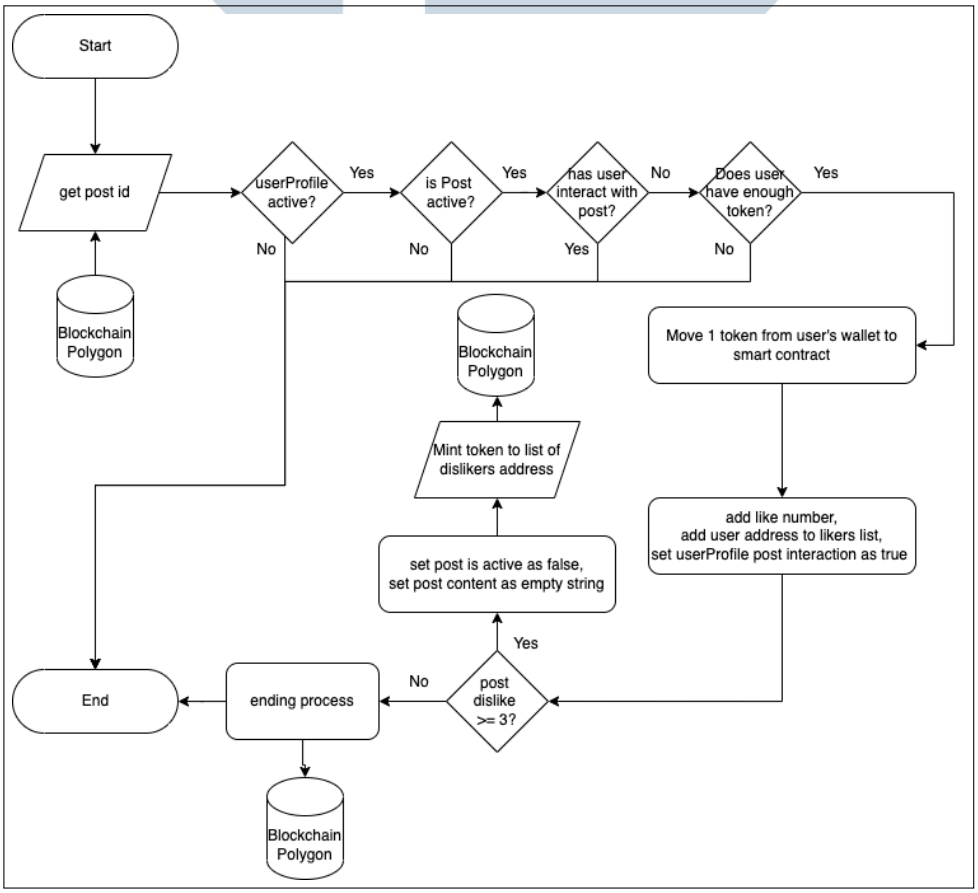
Gambar 3.3. Diagram alir *like* (*vote* setuju)

Para pengguna yang memiliki 100 token atau lebih dapat melakukan menggunakan fungsi ini. Pertama-tama fungsi ini menerima *input* berupa pos id yang di *like*. Setelah itu fungsi melakukan pengecekan terhadap status *wallet address* apakah sudah terdaftar. Jika sudah maka dicek kembali apakah pos masih aktif (belum di ditolak maupun diterima). Setelah itu pengguna sudah pernah melakukan interaksi (*like/dislike/post*) terhadap pos ini. Saldo pengguna juga di cek apakah mencukupi untuk menggunakan fungsi ini. Apabila seluruh persyaratan sudah ter-

penuhi maka fungsi memindahkan satu token dari *wallet* pengguna ke *smart contract*. Properti dari pos diperbaharui dan fungsi melakukan pengecekan terhadap jumlah *vote* setuju yang telah diterima. Jika jumlah *vote* setuju sudah mencapai angka *threshold* dalam hal ini tiga maka pos diterima dan melakukan *update* kembali terhadap properti pos. Setelah properti pos diperbaharui pos dapat muncul ke *public page*. Selain itu fungsi juga mengembalikan seluruh token yang di bayarkan oleh para pengguna yang melakukan *vote* setuju beserta dengan insentif. Fungsi ini insentif yang diberikan adalah sebesar 1%. Apabila seluruh proses telah selesai dilakukan maka data ditambahkan dan diperbaharui pada jaringan *blockchain Polygon*.

D Dislike (Vote Tidak Setuju)

Fungsi ini bertujuan untuk memberikan *vote* tidak setuju terhadap kontenyang telah di pos. Alur dari fungsi ini dapat terlihat pada Gambar 3.4.

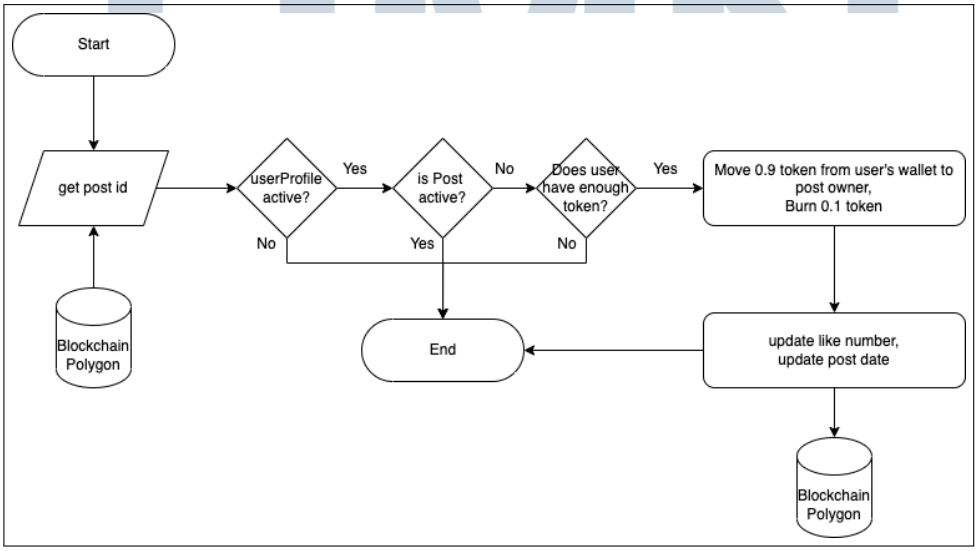


Gambar 3.4. Diagram alir *dislike* (*vote* tidak setuju)

Para pengguna yang memiliki 100 token atau lebih dapat melakukan menggunakan fungsi ini. Pertama-tama fungsi ini menerima *input* berupa pos id yang di *dislike*. Setelah itu fungsi melakukan pengecekan terhadap status *wallet address* apakah sudah terdaftar. Jika sudah maka dicek kembali apakah pos masih aktif (belum di ditolak maupun diterima). Setelah itu pengguna sudah pernah melakukan interaksi (*like/dislike/post*) terhadap pos ini. Saldo pengguna juga di cek apakah mencukupi untuk menggunakan fungsi ini. Apabila seluruh persyaratan sudah terpenuhi maka fungsi memindahkan satu token dari *wallet* pengguna ke *smart contract*. Properti dari pos diperbaharui dan fungsi melakukan pengecekan terhadap jumlah *vote* setuju yang telah diterima. Jika jumlah *vote* setuju sudah mencapai angka *threshold* dalam hal ini tiga maka pos diterima dan melakukan *update* kembali terhadap properti pos. Setelah properti pos diperbaharui pos dapat muncul ke *public page*. Selain itu fungsi juga mengembalikan seluruh token yang di bayarkan oleh para pengguna yang melakukan *vote* setuju beserta dengan insentif. Fungsi ini insentif yang diberikan adalah sebesar 1%. Apabila seluruh proses telah selesai dilakukan maka data ditambahkan dan diperbaharui pada jaringan *blockchain Polygon*.

E Like (Public Page)

Fungsi ini dapat digunakan untuk melakukan *like* pada *public page*. Alur dari fungsi ini dapat terlihat pada Gambar 3.5.

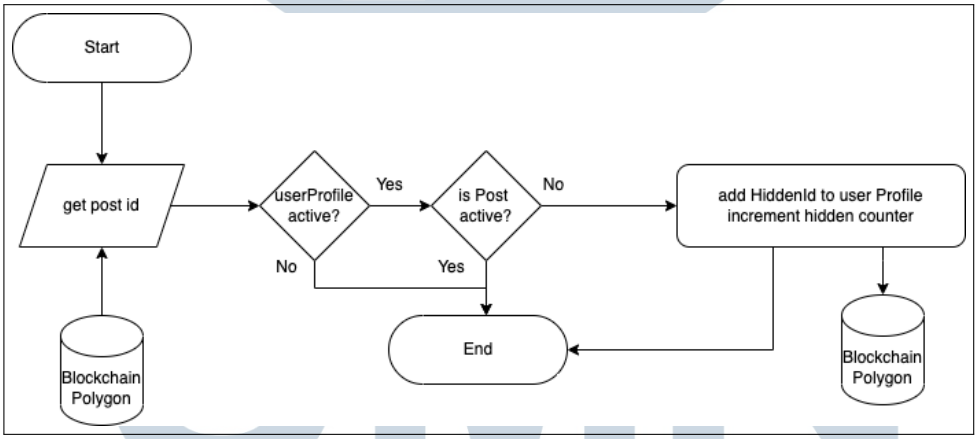


Gambar 3.5. Diagram alir *like* (*public page*)

Pertama-tama fungsi melakukan pengecekan terhadap *wallet address* apakah sudah terdaftar pada media sosial atau belum. Jika sudah maka dilakukan pengecekan terhadap status pos apakah sudah selesai di *vote* atau belum. Apabila pos sudah menyelesaikan *vote* dan diterima maka fungsi melakukan pengecekan terhadap jumlah token yang dimiliki oleh pengguna cukup atau tidak untuk melakukan transaksi. Apabila cukup maka fungsi ini mengambil satu buah token dari *wallet* pengguna yang melakukan *like* dan lalu token tersebut dibagikan ke pemilik pos sebanyak 0.9 token dan dilakukan *burn* terhadap token sisanya yaitu sebesar 0.1 token. Setelah itu fungsi melakukan pembaharuan terhadap jumlah *like* pos. Pembaharuan juga dilakukan terhadap properti *date* sebagai tanda *last interaction* yang dilakukan terhadap pos. Setelah itu seluruh proses yang telah dijalankan oleh fungsi dimasukkan ke dalam *blockchain*.

F hideMinted

Fungsi *hideMinted* berjalan seperti pada Gambar 3.11.

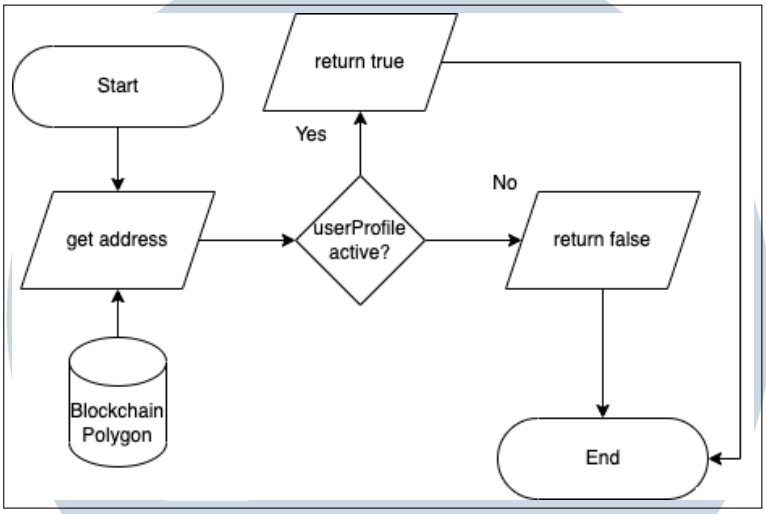


Gambar 3.6. Diagram alir *hideMinted*

Fungsi ini melakukan pengecekan terhadap status *wallet* dan lalu melakukan pengecekan terhadap status pos. Apabila *address wallet* belum terdaftar atau pos masih dalam keadaan aktif (proses *voting* sedang berlangsung) makan fungsi langsung berakhir. Jika proses pengecekan berjalan lancar maka fungsi menambahkan id pos ke *userProfile* dan melakukan penambahan terhadap *hidden counter* dari profil tersebut. Setelah proses berhasil dilakukan maka fungsi berkhir dan data diperbaharui pada *blockchain Polygon*.

G *isRegistered*

Fungsi *isRegistered* berjalan seperti pada Gambar 3.7.

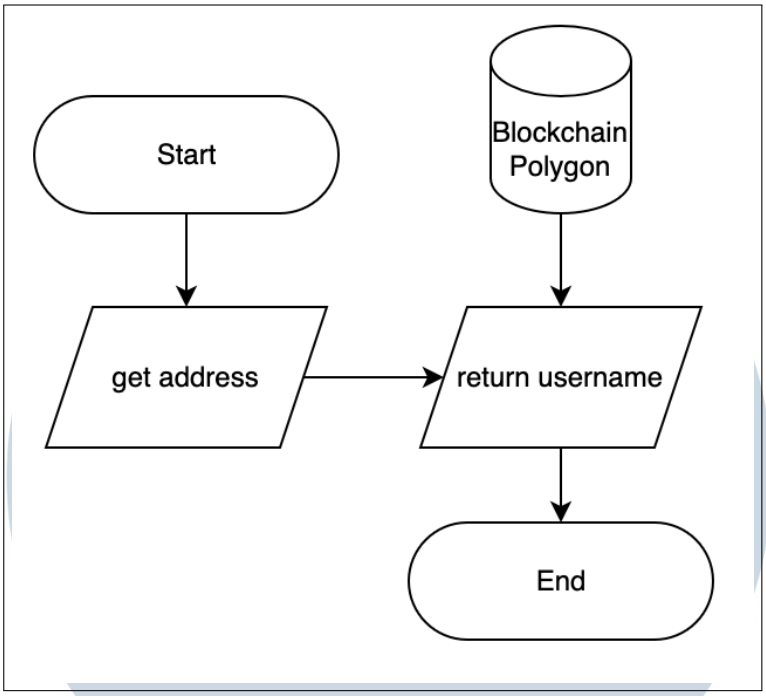


Gambar 3.7. Diagram alir *isRegistered*

Fungsi ini bertujuan untuk melakukan pengecekan terhadap status *wallet address* pada media sosial. Fungsi menerima *address* dari *wallet* yang ingin dilakukan pengecekan ke jaringan *blockchain*. Setelah mendapatkan jawaban maka fungsi mengembalikan hasil dari fungsi dalam bentuk *boolean* yaitu *true* apabila sudah terdaftar dan *false* apabila belum terdaftar.

H *getUsername*

Fungsi *getUsername* berjalan seperti pada Gambar 3.8.

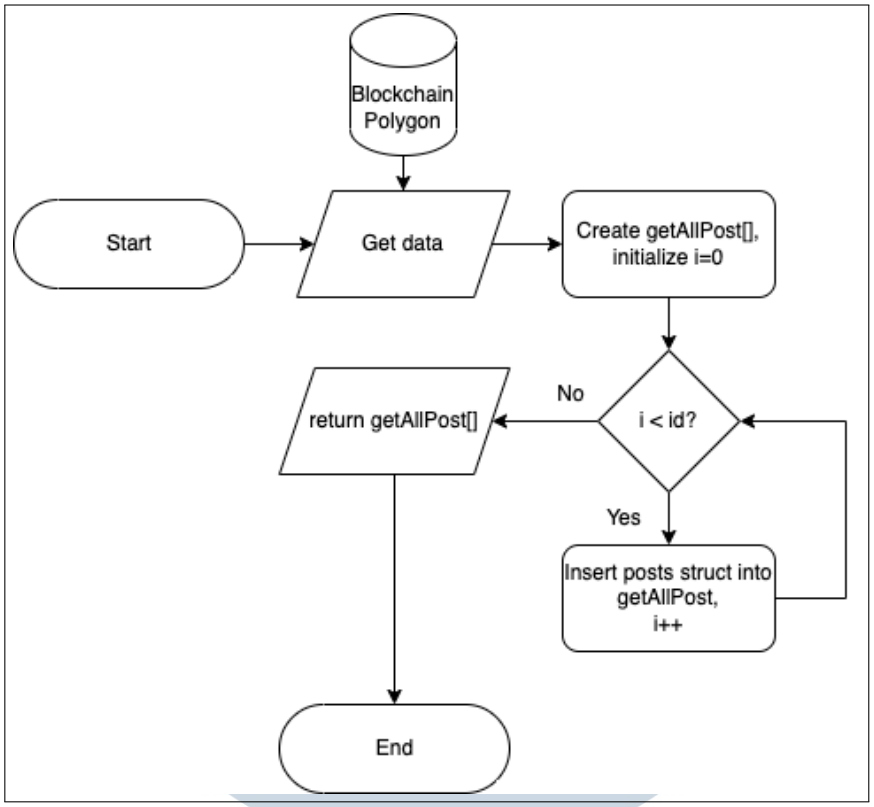


Gambar 3.8. Diagram alir *getUsername*

Fungsi ini bertujuan untuk mendapatkan *username* dari *blockchain*. Pertama-tama fungsi menerima *address* dari *wallet* yang ingin dilakukan pengecekan lalu fungsi langsung mengembembalikan *username* yang telah didapatkan dari blockchain.

I *getAllPost*

Fungsi *getAllPost* berjalan seperti pada Gambar 3.9.

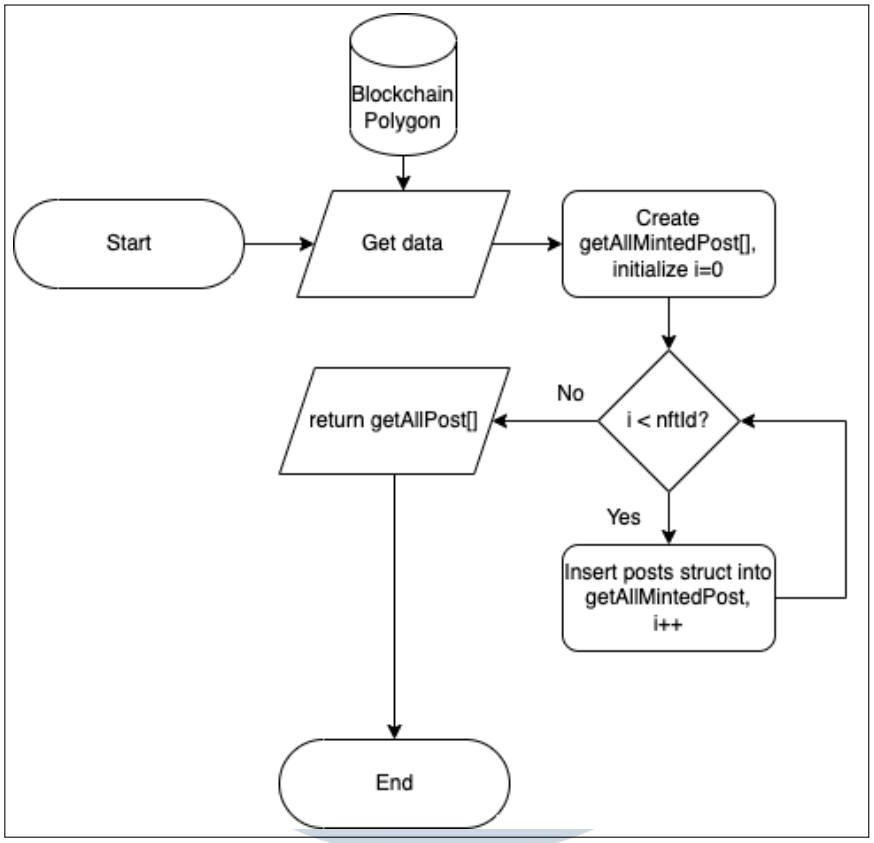


Gambar 3.9. Diagram alir *getAllPost*

Fungsi ini bertujuan untuk mengambil seluruh pos yang telah dibuat pada *smart contract*. Pertama-tama fungsi mengambil data yang dibutuhkan dari blockchain, lalu fungsi membuat *array of struct* untuk menampung pos, setelah itu dilakukan *loop* sejumlah variable id yang ada pada *smart contract*. Setiap *loop* memasukan seluruh pos yang ada pada *smart contract*. Setelah *loop* selesai dijalankan fungsi mengembalikan *array of struct* yang sudah diisi dengan seluruh pos yang ada pada media sosial.

J *getAllMintedPost*

Fungsi *getAllMintedPost* berjalan seperti pada Gambar 3.10.

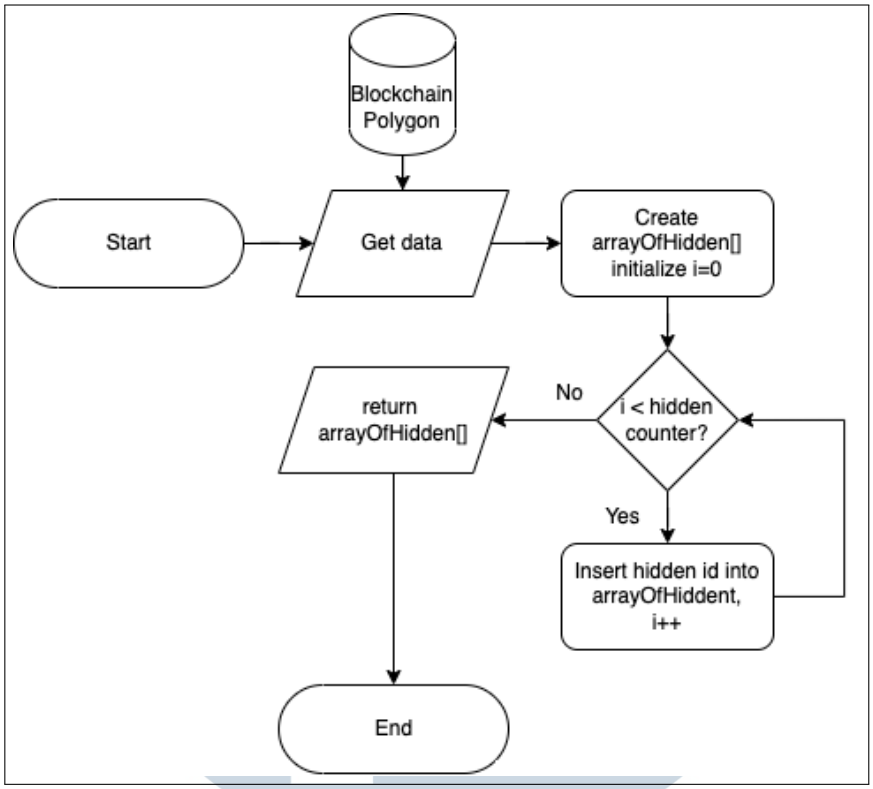


Gambar 3.10. Diagram alir *getAllMintedPost*

Fungsi ini bertujuan untuk mengambil seluruh pos yang telah di*mint* oleh *smart contract*. Pertama-tama fungsi mengambil data yang dibutuhkan dari blockchain, lalu fungsi membuat *array of struct* untuk menampung informasi pos yang telah di*mint*, setelah itu dilakukan *loop* sejumlah variable nftId yang ada pada *smart contract*. Setiap *loop* memasukan seluruh pos yang telah di*mint* oleh *smart contract*. Setelah *loop* selesai dijalankan fungsi mengembalikan *array of struct* yang sudah diisi dengan seluruh pos yang telah di*mint* oleh media sosial.

K getHiddenId

Fungsi *getHiddenId* bekerja seperti digambarkan pada Gambar 3.11.



Gambar 3.11. Diagram alir *getHiddenId*

Fungsi mengambil data dari *blockchain Polygon* lalu membuat *array of unsigned integer* untuk menampung *hidden id*. Fungsi melakukan *looping* selama nilai dari i lebih kecil dari variable *hidden counter*. *Looping* terus menambahkan *hidden id* yang didapat ke dalam *arrayyOfHidden*. Setelah looping selesai dilakukan maka fungsi mengembalikan *arrayOfHidden*.

3.3.3 Front-end

Front-end menghubungkan seluruh fungsi yang ada pada *smart contract* selain itu *front-end* juga melakukan pengecekan akses terhadap penampilan halaman *private page*, pengecekan akses tersebut dilakukan berdasarkan jumlah token yang dimiliki oleh pengguna dan jumlah token tersebut didapatkan dari jaringan blockchain secara langsung. Berikut kegunaan yang ada pada halaman masing-masing:

- *Public Page*

  Public page memiliki fungsi utama yaitu menampilkan pos yang telah diseleksi pada *private page*, selain itu pada *public page* para pengguna juga da-

pat melakukan *like* terhadap pos. Selain itu pada bagian atas halaman pengguna dapat melakukan pos terhadap konten yang ingin dibagikan pada media sosial.

- *Private Page*

  Private page menampilkan pos yang perlu diseleksi sebelum memasuki *public page*, sehingga fungsi utama dari halaman ini adalah agar para pengguna dapat melakukan *voting* dengan cara melakukan *like* maupun *dislike*. Untuk memasuki halaman ini para pengguna wajib memiliki minimal 100 *ICredibility Token*.

3.4 Pengujian dan Evaluasi

Pengujian akan dilakukan dengan melakukan pengetesan setiap fungsi untuk mengukur *gas used* yang diperlukan. Pengujian juga akan melakukan transaksi sebanyak 20 kali untuk melihat rata-rata dari *gas fee* serta *transaction time* yang diperlukan oleh jaringan *blockchain Polygon* untuk menjalankan fungsi yang ada pada *smart contract*. Pengujian pada saat yang bersamaan akan membuktikan bahwa *smart contract* dapat berjalan dengan baik.

Evaluasi lalu akan dilakukan menggunakan struktur pertanyaan EUCS untuk mendapatkan kepuasan dari para responden terhadap sistem yang telah dibuat. EUCS merupakan sebuah metodologi yang digunakan untuk mengukur tingkat kepuasan pengguna terhadap sistem [25]. Pertanyaan yang diberikan kepada para pengguna berbentuk satu pertanyaan terhadap kepuasan sistem *voting*, satu pertanyaan terhadap konten yang ditampilkan. Satu pertanyaan mengenai waktu yang dibutuhkan untuk melakukan sebuah transaksi, waktu tersebut akan didapat dari rata-rata *transaction time* waktu yang dilakukan pada bagian pengujian. Satu pertanyaan terhadap *transaction fee* yang harus dibayarkan untuk melakukan interaksi, *transaction fee* didapatkan dari melakukan perkalian antara rata-rata *gas used* serta *gas fee* dari pengujian sebelumnya. Target dari pengujian serta metode dapat dilihat pada Tabel 3.3.

Tabel 3.3. Target pengujian dan evaluasi

Variabel	Metode	Hasil
Gas used	menjalankan seluruh fungsi	rata-rata *gas used*
Gas fee	melakukan transaksi 20 kali	rata-rata *gas fee*
Transaction time	melakukan transaksi 20 kali	rata-rata waktu yang diperlukan
Tingkat kepuasan	menyebarkan kuesioner	nilai tingkat kepuasan