BAB III
METODOLOGI PENELITIAN

3.1. Subyek dan Obyek Penelitian

Subyek penelitian ini merupakan sistem *e-voting*. Objek penelitian ini adalah pembuatan *website dApps e-voting* dengan *blockchain Polygon*.

3.2. Alat dan Bahan Penelitian

3.2.1 Perangkat Keras

Perangkat keras yang dibutuhkan dalam proses penelitian ini adalah Laptop HP Pavillion Gaming Laptop 15-ec1xxx dengan spesifikasi sebagai berikut:

1. *Processor* AMD Ryzen 5 4600H
2. *Memory* 16GB DDR4
3. *Storage* 500GB SATA SSD
4. *Display* 15” IPS

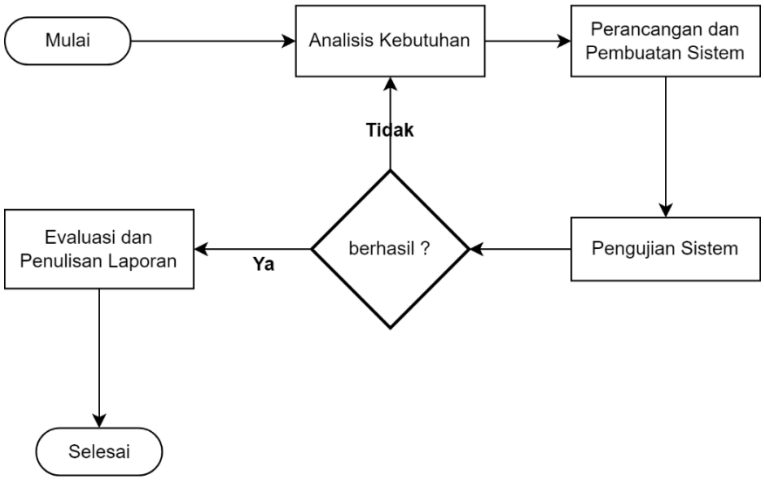
3.2.2 Perangkat Lunak

Perangkat lunak yang dibutuhkan dalam penelitian ini adalah sebagai berikut:

1. Windows 11 *Home Single Language*
2. Visual studio code
3. Node
4. Brave sebagai web browser

3.3. Diagram Alir Penelitian

Penelitian dimulai dengan perumusan masalah, tujuan dan manfaat terkait dengan teknologi *blockchain*. Tahapan selanjutnya analisis kebutuhan dengan cara studi literatur mengumpulkan dan memahami informasi terkait pengimplementasian *blockchain* untuk sistem *e-voting*. Tahapan ketiga perancangan aplikasi dan pembuatan sistem *e-voting*. Tahapan selanjutnya pengujian sistem jika sistem berjalan dengan baik maka lanjut ke tahap evaluasi dan penulisan laporan.



Gambar 3.1 Diagram Alir Penelitian

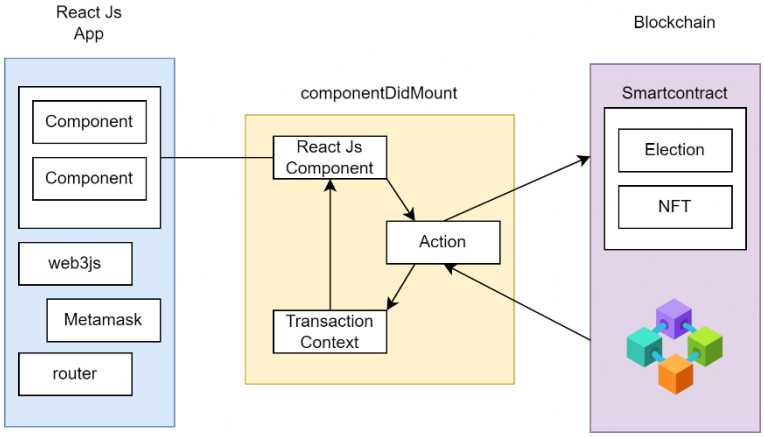
3.3.1 Analisis Kebutuhan

Penelitian ini diawali dengan studi literatur, peneliti mengumpulkan informasi-informasi terkait tentang *e-voting* yang akan diteliti. Informasi diperoleh dari jurnal, buku, paper, artikel dan skripsi terdahulu. Permasalahan diperkuat dengan cara studi literatur sehingga dapat digunakan dalam penelitian ini dan penelitian berikutnya. Peneliti menganalisis kebutuhan dalam merancang *website dApp e-voting* untuk mengetahui apa saja yang diperlukan dalam pembangunan *website*.

3.3.2 Perancangan dan Pembuatan Sistem

3.2.2.1 Perancangan Sistem

a. Diagram Arsitektur Sistem

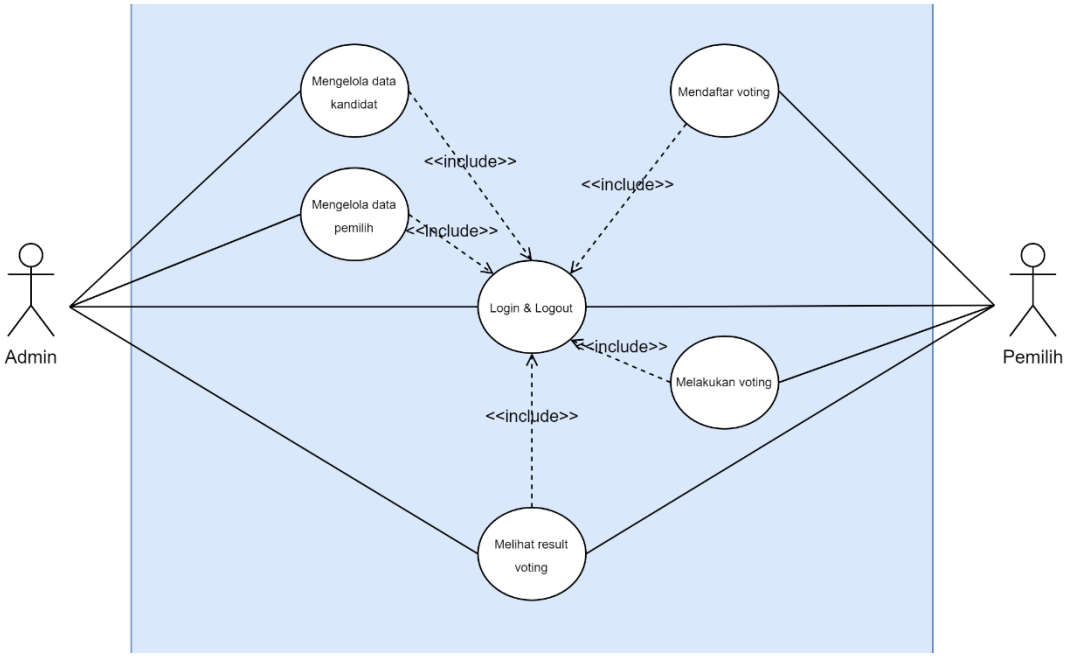


Gambar 3.2 Diagram Arsitektur Sistem

Pada Gambar 3.2 diagram arsitektur alurnya adalah *react component* dan *dependency* di-render menggunakan *componentDidMount* secara asinkron. Setelah ada action maka akan memanggil data yang ada pada *blockchain Polygon*.

b. Use Case Diagram

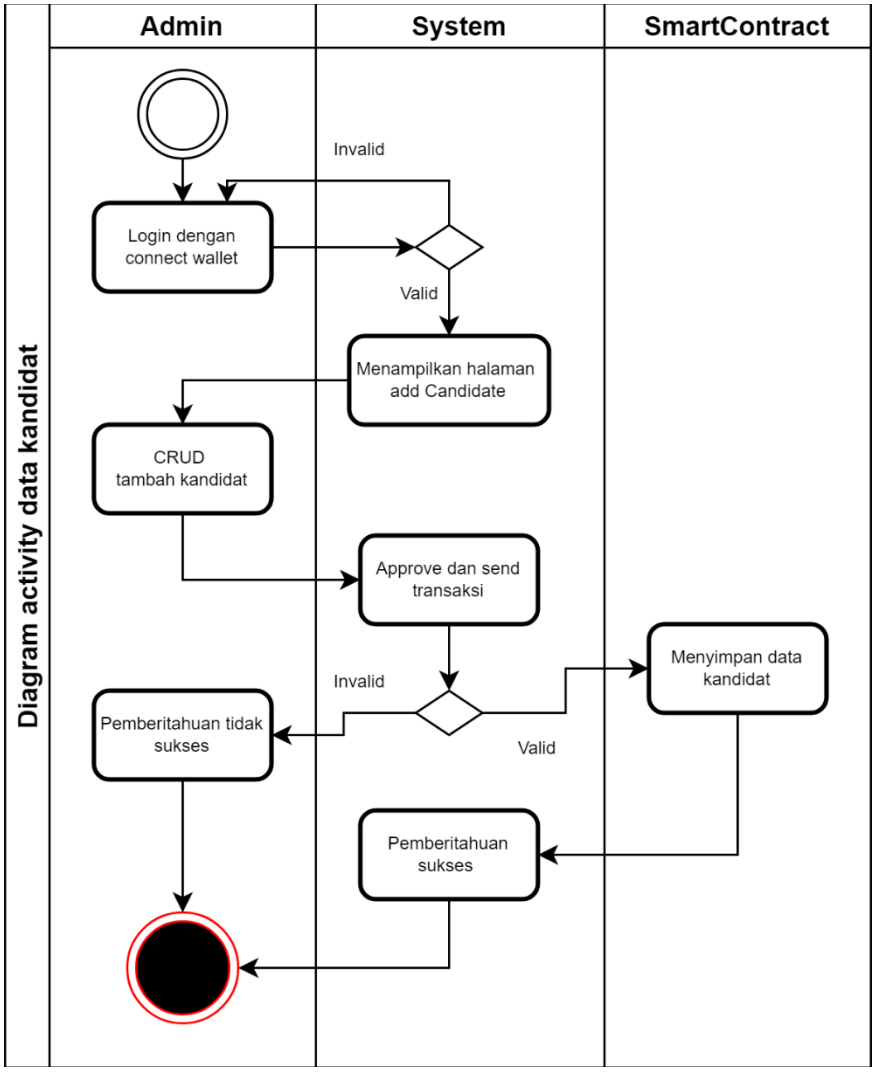
Sistem *e-voting* ini memiliki rangkaian interaksi antara aktor dan sistem terlihat pada Gambar 3.3. Pada sistem terdapat dua aktor yaitu admin dan pemilih. Aktor admin dapat mengelola data kandidat, mengelola data pemilih, melihat result voting, *login* dan *logout*. Sedangkan aktor pemilih bisa melihat result voting, mendaftar voting, melakukan voting, *login* dan *logout*.



Gambar 3.3 *Use Case* Diagram

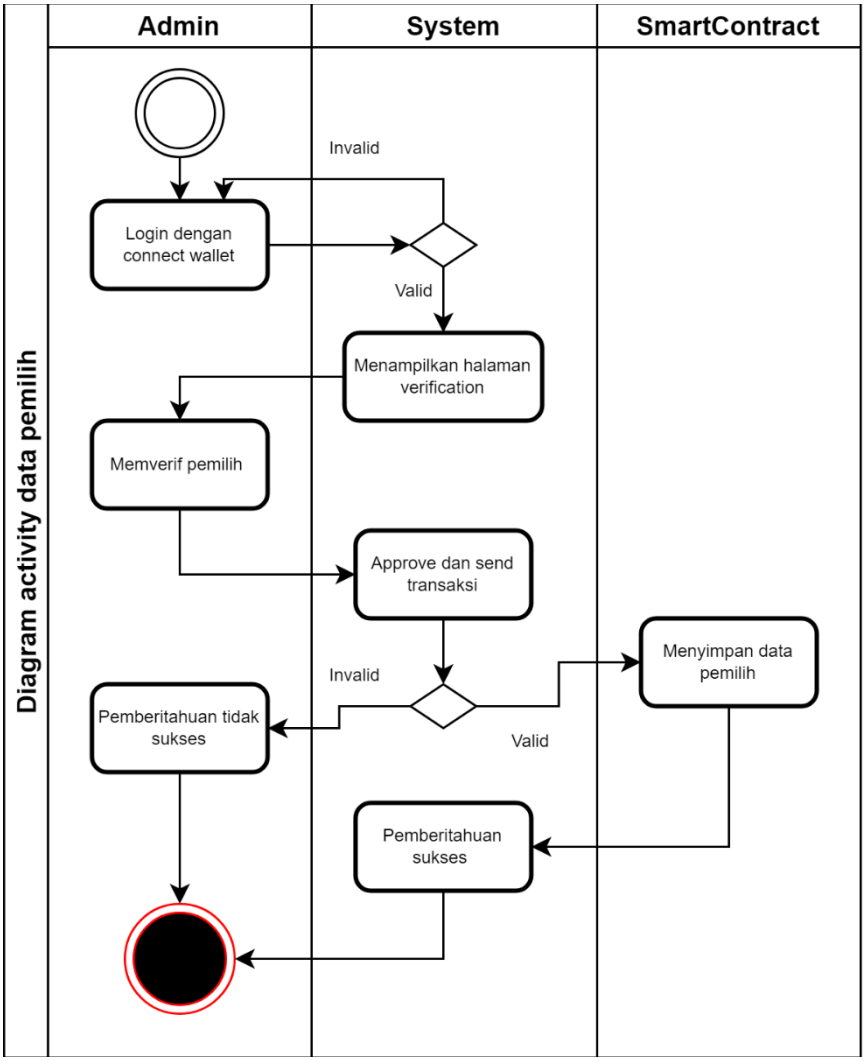
c. *Activity* Diagram

Activity diagram mengelola data kandidat merupakan salah satu *activity use case* mengelola data kandidat. Pada Gambar 3.4 admin memiliki wewenang untuk menambahkan data kandidat. Data kandidat yang sudah ditambahkan dan terdeploy ke *smart contract* tidak dapat dihapus karena sudah tercatat pada *blockchain Polygon*.



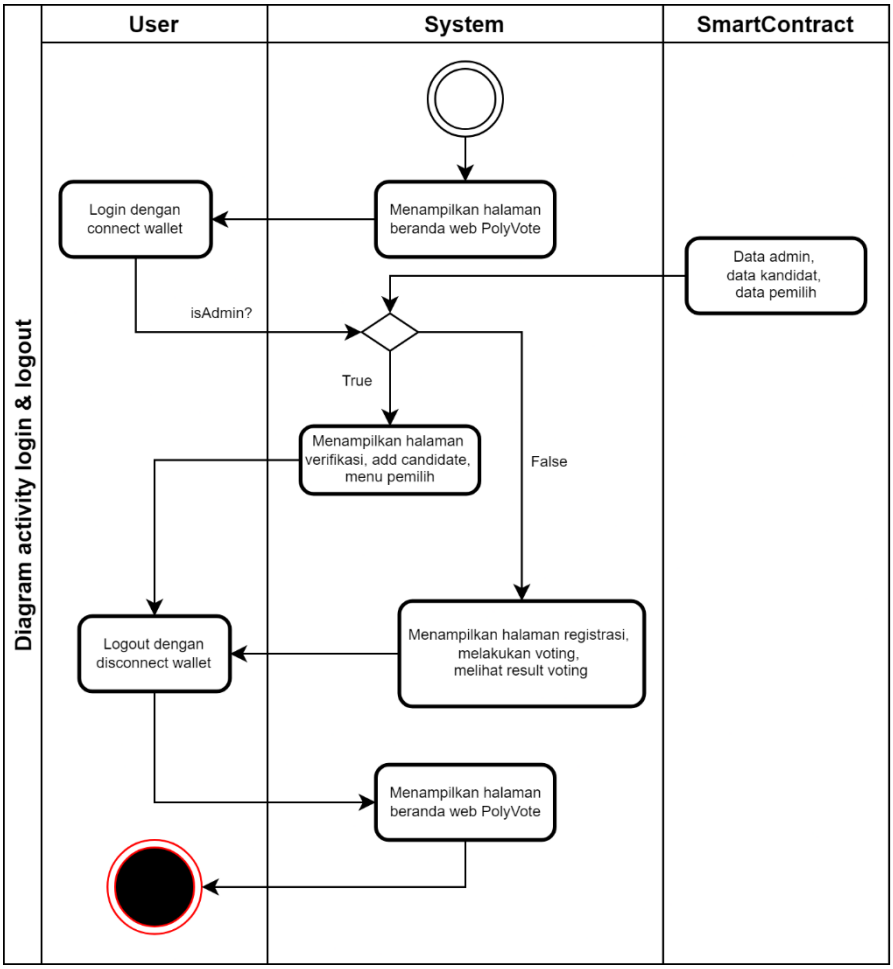
Gambar 3.4 Diagram *Activity* mengelola Data Kandidat

Activity diagram mengelola data pemilih merupakan salah satu *activity use case* mengelola data pemilih. Pada Gambar 3.5 admin memiliki wewenang untuk memverifikasi data pemilih yang telah mendaftar. Admin mengecek data pemilih untuk memastikan bahwa data yang dikirim oleh user valid. Setelah dicek dan admin mengkonfirmasi data pemilih maka data akan disimpan pada *smart contract*.



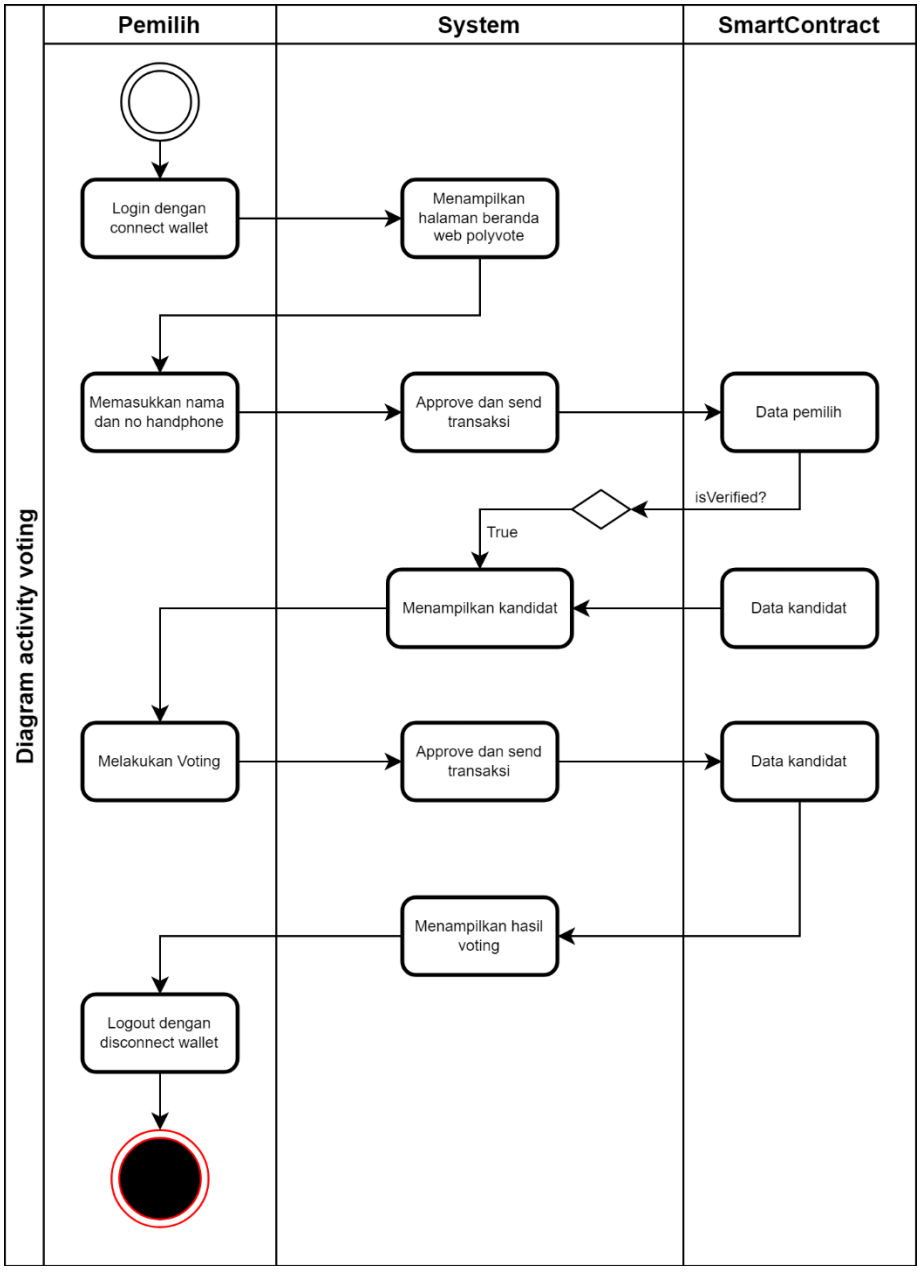
Gambar 3.5 Diagram *Activity* mengelola Data Pemilih

Activity diagram *login* dan *logout* merupakan salah satu *activity use case login* dan *logout*. Pada Gambar 3.6 user dapat *login* dengan *connect wallet* metamask. User menghubungkan *wallet* metamask dengan *website* dan kemudian sistem memverifikasi alamat address yang terkait dengan akun tersebut merupakan admin dari *smart contract* atau bukan. User logout dengan cara *disconnect wallet*. Hal ini mengakhiri sesi akses pengguna ke dalam sistem dan hanya bisa mengakses halaman beranda sistem.



Gambar 3.6 Diagram *Activity Login* & *Logout*

Activity diagram melakukan *voting* merupakan salah satu *activity use case* melakukan *voting*. Pada Gambar 3.7 pemilih melakukan registrasi, melihat kandidat, melakukan *voting*, *mint* NFT dan melihat result sementara. Ketika user melakukan registrasi dan melakukan *voting* memerlukan biaya transaksi berupa *coin* matic. Langkah awal adalah user registrasi, pemilih dapat melihat daftar kandidat sebelum melakukan *voting.* Setelah berhasil melakukan *voting* data kandidat akan terupdate secara langsung. Sistem menampilkan hasil sementara *voting* sebelum sesi *voting* berakhir.

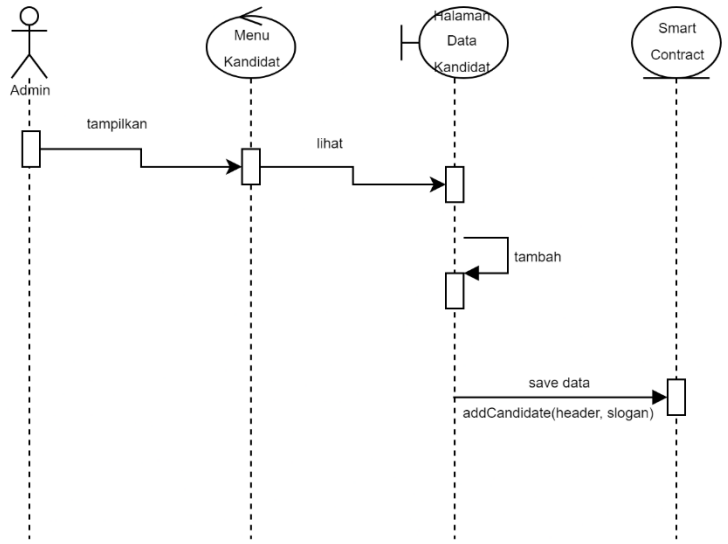


Gambar 3.7 Diagram *Activity* melakukan *Voting*

d. *Sequence* Diagram

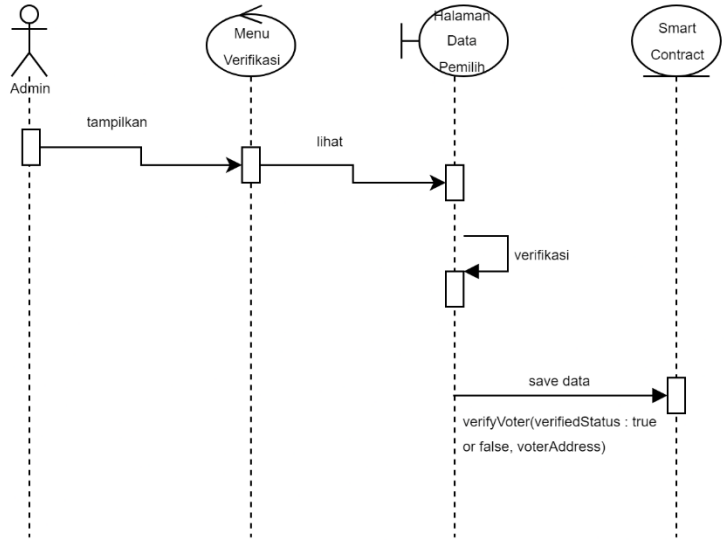
Gambar 3. 8 adalah saat admin menambahkan data kandidat ke *blockchain*. Data yang disimpan berupa header dan slogan masing-masing kandidat memiliki id. Input data ke *blockchain*

berupa method *Add Candidate* dengan *field* sesuai dari *smart contract* yang telah dibuat.

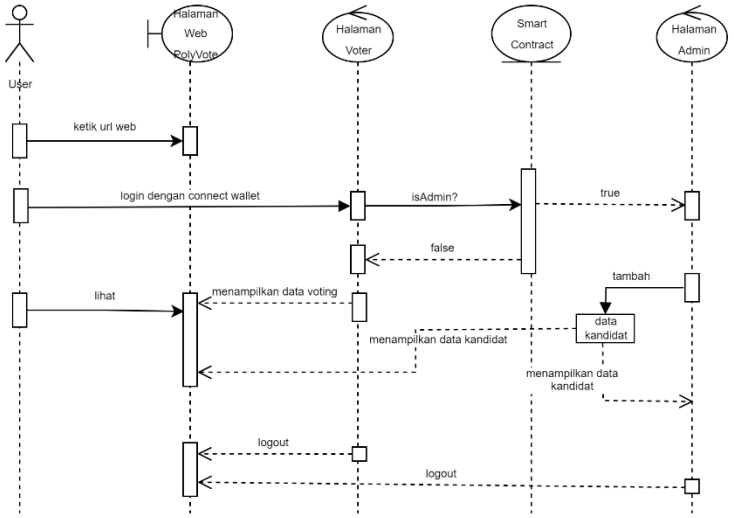


Gambar 3.8 *Sequence* Diagram mengelola Data Kandidat

Gambar 3.9 adalah saat admin mengelola data pemilih dengan memverifikasi tiap user. *verifyVoter* merupakan method *smart contract* untuk memberikan *true* atau *false* pada setiap address yang telah mendafatar hanya bisa dilakukan oleh admin.



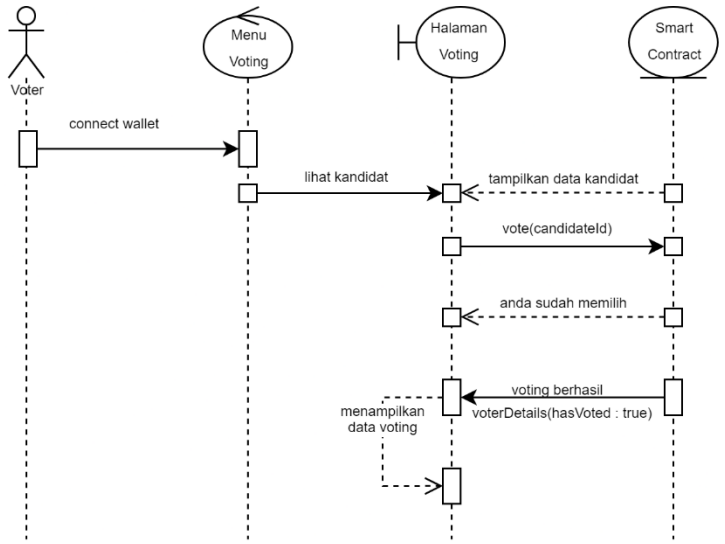
Gambar 3.9 *Sequence* Diagram mengelola Data Pemilih



Gambar 3.10 *Sequence* Diagram *Login* & *Logout*

Gambar 3.10 merupakan diagram sequence saat user login dan *logout* pada *website*. User *login* dengan *connect wallet* dan *logout* dengan *disconnect wallet* menggunakan *Remote Procedure Call* (RPC) jaringan *polygon* mumbai. Ketika user *login* akan dicek apakah admin atau bukan data diambil dari *blockchain*. Admin adalah orang yang mendeploy *smart contract*, jika user admin makan akan menampilkan seluruh halaman admin. Jika user bukan admin maka halaman admin akan disembunyikan.

Gambar 3.11 merupakan diagram *sequence* yang menggambarkan langkah-langkah saat *voter* melakukan *voting* melalui *website*. Hanya user yang telah mendaftar menggunakan method *RegisterAsVoter* dan diverifikasi admin yang bisa melakukan *vote* kandidat. Setelah *voter* memilih kandidat, suara *voter* dicatat dan disimpan ke *blockchain.* Status *voterDetail voter* akan berubah ketika berhasil melakukan *vote* dengan method *vote* pada *smart contract* melalui *website* sistem.



Gambar 3.11 *Sequence* Diagram melakukan *Voting*

3.2.2.2 Pembuatan Sistem

Tahap perancangan telah selesai selanjutnya dilakukan pengkodean sistem dengan bahasa pemrograman. *Smart Contract* dibuat dengan bahasa solidity dengan framework truffle untuk developmenya menggunakan jaringan *polygon* mumbai. Pembuatan *website dApps* menggunakan react js dengan beberapa dependencies yang di install untuk berinteraksi dengan *smart contract* dan user *dApps*.

3.3.3 Pengujian Sistem

Tahapan pengujian sistem akan dilakukan setelah tahap pembuatan sistem telah selesai. Pengujian dilakukan dengan *blackbox testing* sistem polyvote.

Pengujian *website dApp* menggunakan metode pengujian *blackbox testing.* Metode *blackbox testing* digunakan dengan harapan dapat mengetahui fungsionalitas dari *e-voting* ini. Langkah awal untuk melakukan pengujian sistem adalah dengan mendefinisikan fitur yang akan diuji pada setiap page oleh admin dan user. Daftar fitur pengujian dapat dilihat pada Tabel 3.1 dan Tabel 3.2.

Tabel 3.1 Fitur Pengujian oleh User

No	Item yang Diuji	Cara Pengujian	Hasil yang diharapkan
1	*Connect wallet*	Memasukkan *password* dan mengganti Metamask ke jaringan *Polygon Mumbai*	User dapat melakukan *connect wallet*
2	Melakukan registrasi	Mengisi formulir registrasi	Registrasi terkirim ke jaringan *blockchain*
3	Melihat kandidat	Klik *Navbar Voting*	User dapat melihat kandidat yang ada
4	Melakukan *vote*	Klik *vote* pada salah satu kandidat	*Voting* berupa id terkirim
5	*Minting NFT*	Klik *button Mint NFT*	User mendapatkan NFT setelah sukses voting

Tabel 3.2 Fitur Pengujian oleh Admin

No	Item yang Diuji	Cara Pengujian	Hasil yang diharapkan
1	*Connect wallet*	Memasukkan password dan mengganti Metamask ke jaringan *Polygon Mumbai*	Admin dapat melakukan *connect wallet*
2	Memulai *voting*	Klik *button start*	Memulai sesi *voting*
3	Mengakhiri *voting*	Klik *button end*	Mengakhiri sesi *voting* menampilkan kandidat pemenang
4	Menambah kandidat	Klik *button add* setelah isi formulir kandidat	Menambah kandidat
5	Memverifikasi registrasi	Klik *button verif* pada setiap user yang *register*	Verifikasi user agar dapat memilih