BAB 3
PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Organisasi

Kedudukan selama melakukan proses kerja magang dalam perusahaan Pixel8labs Pte. Ltd. pada divisi Tinkers adalah sebagai *Fullstack Engineer* bersama dengan tim Buidl Ape Yacht Club. Buidl Ape Yacht Club Team memiliki lima anggota dengan pembagian satu orang *project manager*, satu *fullstack engineer*, satu *solidity engineer*, dan dua *backend engineer*. Tim Buidl Ape Yacht Club memiliki proyek untuk mengimplementasikan *dapps* Buidl Ape Yacht Club pada Pixel8labs Pte. Ltd. sehingga dapat memberikan kebebasan pada penggiat untuk menentukan *traits* dari NFT Bored Ape Yacht Club yang ingin mereka miliki.

Proyek Buidl Ape Yacht Club dikembangkan dengan metode *scrum* yang menerapkan prinsip *agile* pada manajemen pengembangan proyek peranti lunak. Metode ini dilakukan dengan *sprint*. *Sprint* adalah aktivitas yang harus diselesaikan oleh tim dalam periode waktu yang telah ditentukan. Setiap anggota tim mengerjakan tugas sesuai dengan *backlog* yang diberikan saat *sprint planning* dan dilakukan setiap satu minggu sekali oleh *project manager* tim. *Sprint planning* dilakukan untuk merencanakan apa yang akan dikerjakan oleh setiap anggota tim selama satu minggu kedepan dalam proyek Buidl Ape Yacht Club.[6]

3.2 Tugas yang Dilakukan

Dalam pelaksanaan kerja magang pada Pixel8labs Pte. Ltd., tugas diberikan secara daring melalui aplikasi Jira Software. Tugas yang diberikan berupa implementasi frontend dan smart contract pada *dapps* Buidl Ape Yacht Club. Selama pelaksanaan kerja magang, tanggung jawab yang diberikan adalah sebagai berikut.

1. Memahami proses bisnis proyek Buidl Ape Yacht Club
2. Mempelajari dan membangun portal dengan *framework* Next.js
3. Melakukan pengembangan terhadap *frontend* web secara *responsive* dan terintegrasi dengan *smart contract*
4. Menerapkan UI/UX dari *frontend* web yang dibangun dan menggunakan aset yang diterima dari UI/UX *designer*

Proses pembuatan *dapps* menggunakan komputer dengan spesifikasi sebagai berikut.

1. Prosesor: AMD Ryzen 5 3500 6-Core Processor 3,6GHz
2. Memori: 16GB RAM
3. Penyimpanan: SSD 512 GB

Adapun perangkat lunak yang digunakan untuk mengembangkan portal adalah sebagai berikut.

1. Sistem Operasi: Windows 10
2. Visual Studio Code, sebagai *text editor* untuk penulisan kode program *frontend* dan *smart contract*
3. MetaMask, sebagai *wallet* melakukan *testing* proses *minting*
4. Github, sebagai tempat penyimpanan kode dan kolaborasi dengan pengembang lainnya
5. Peramban: Brave dan Google Chrome

3.3 Uraian Pelaksanaan Magang

Selama praktik kerja magang, tugas yang dilakukan adalah mengimplementasikan *frontend* dan *smart contract dapps* Buidl Ape Yacht Club. Tugas-tugas yang dikerjakan setiap minggunya selama praktik kerja magang mengikuti *backlog* yang ada dan dapat dilihat lebih rinci pada Tabel 3.3.

Tabel 3.1. Pekerjaan yang dilakukan selama pelaksanaan kerja magang

Minggu Ke -	Pekerjaan yang dilakukan
1	Pengenalan proyek Buidl Ape Yacht Club dan pembuatan *frontend* dan integrasi *web3* pada *dapps*, melakukan *debugging* pada masalah *auto refreshing* pada proyek Signatures Pub, dan melakukan *revamp* pada proyek Buidl Ape Yacht Club

Tabel 3.2. Pekerjaan yang dilakukan selama pelaksanaan kerja magang (lanjutan)

Minggu Ke -	Pekerjaan yang dilakukan
2	Memperbaiki beberapa *bug*, melakukan *preview image optimization*, dan mengimplementasikan *minting feedback* pada proyek Buidl Ape Yacht Club serta membuat *responsiveness* pada proyek Buidl Ape Yacht Club
3	Membuat *responsiveness* dan integrasi *whitelist minting* pada proyek Buidl Ape Yacht Club, melakukan eksperimen pembuatan *web3* pada *website* SquareSpace
4	Melakukan *gas fee optimization* pada *smart contract* proyek Buidl Ape Yacht Club dan memperbaiki beberapa fitur yang terdapat pada *minting widget*
5	Melakukan *E2E testing* pada proyek Buidl Ape Yacht Club pada rinkeby *testnet* dan melakukan perbaikan berdasarkan hasil *E2E testing*
6	Membuat *unit test* untuk ERC721X yang diimplementasikan pada proyek Buidl Ape Yacht Club dan membuat *responsiveness* untuk fitur *minting*
7	Melakukan *revamp* pada Pixel8labs *company website* dan perbaikan *layout*, memodifikasi *smart contract* pada proyek Creep Crew
8	Mengimplementasikan integrasi *web3*, mengimplementasikan *frontend* dan fitur Globe *dashboard*, dan membuat *minting widget* pada proyek Superordinary Friends
9	Melakukan *final testing* pada proyek Buidl Ape Yacht CLub dan mengerjakan tampilan pada proyek Creep Crew dan 9 Lives Agency
10	Memperbaiki Globe *dashboard*, membuat *responsiveness* pada proyek Pixel8labs *company website*, dan penyempurnaan *minting widget* pada proyek Superordinary Friends
11	Menerapkan *library* ConnectKit, melakukan *final testing* pada proyek Superordinary Friends dan memperbaiki fitur berdasarkan hasil *final testing*

Tabel 3.3. Pekerjaan yang dilakukan selama pelaksanaan kerja magang (lanjutan)

Minggu Ke -	Pekerjaan yang dilakukan
12	Melakukan perbaikan *bug* pada proyek Signatures Pub, Superordinary Friends dan membuat *creator* dan *team section* pada proyek Creep Crew
13	Melakukan *bug bash session*, membuat *wallet checker* pada proyek Superordinary Friends, dan melakukan perbaikan proyek Signatures Pub
14	Melakukan *final testing* pada proyek Superordinary Friends dan membuat *responsiveness* pada Pixel8labs *company website*
15	Membuat *smart contract* dan UI untuk proyek Hungry Hamster Club dan mengimplementasikan UI pada proyek Metaseed
16	Melakukan implementasi integrasi *web3*, melakukan *final testing*, dan melakukan perbaikan akhir pada proyek Hungry Hamster Club

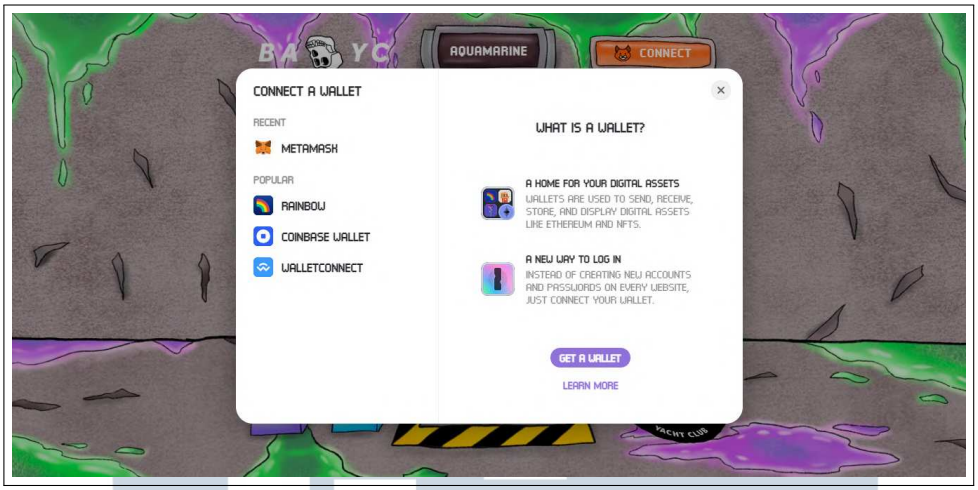
3.3.1 Implementasi

A. Frontend

1. Connect Wallet

Gambar 3.1. Halaman Awal Dapps Buidl Ape Yacht Club

Pada Gambar 3.1 ini menampilkan halaman utama Buidl Ape Yacht Club yang terdapat tombol untuk menghubungkan *wallet* pengguna dengan *dapps* Buidl Ape Yacht Club menggunakan.



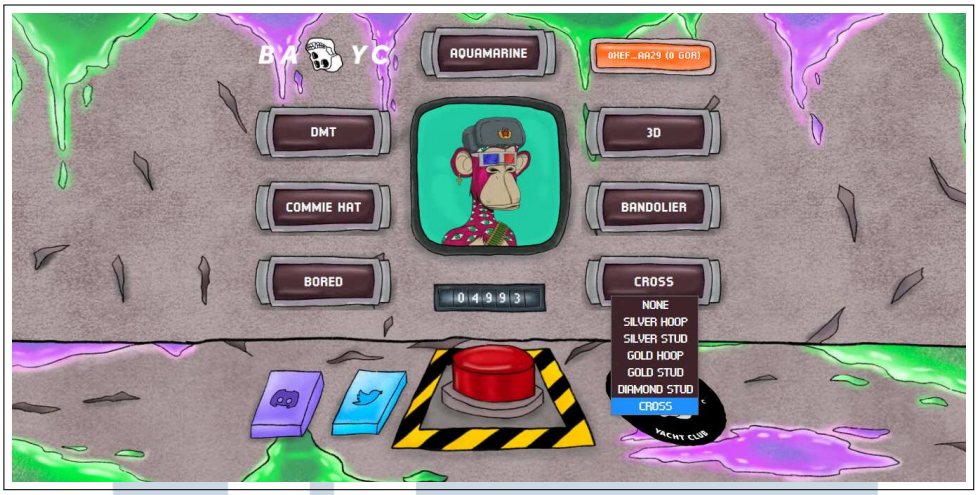
Gambar 3.2. Modal Connect Wallet menggunakan Library RainbowKit

Lalu akan ditampilkan *modal* untuk memilih metode *connect wallet* seperti pada Gambar 3.2. Implementasi fitur *connect wallet* menggunakan *library* RainbowKit.

2. Public Mint

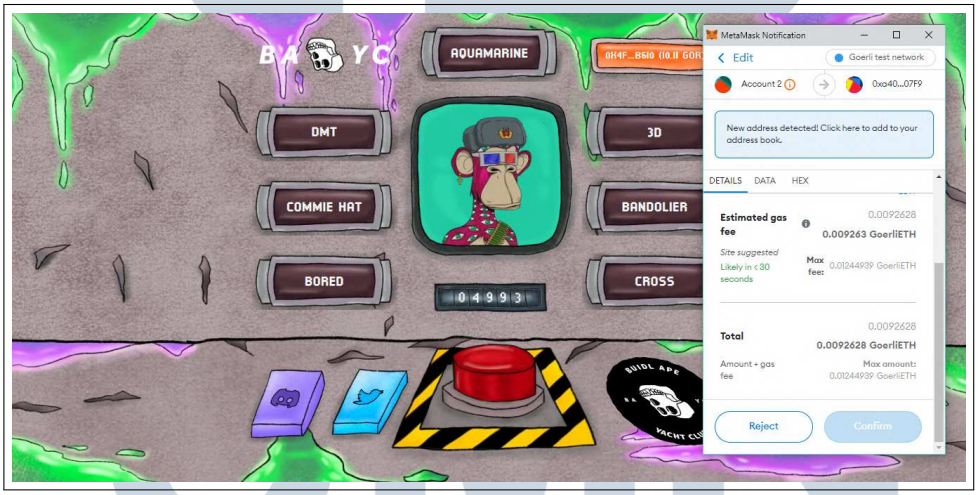
Gambar 3.3. Halaman Awal Dapps Buidl Ape Yacht Club

Pada Gambar 3.3 ini menampilkan halaman utama Buidl Ape Yacht Club yang terdapat tombol untuk melakukan *public minting* NFT Buidl Ape Yacht Club.



Gambar 3.4. Pemilihan Trait yang Diinginkan menggunakan Dropdown

Pengguna dapat memilih *trait* dari NFT yang mereka mau menggunakan *dropdown* sesuai pada Gambar 3.4.



Gambar 3.5. Halaman Konfirmasi Transaksi Public Minting

Gambar 3.6. Modal Loading Transaksi Public Minting

Lalu pengguna perlu mengonfirmasi transaksi (Gambar 3.5) dan halaman akan menampilkan *modal loading* yang dapat mengarahkan pada halaman etherscan (Gambar 3.6).

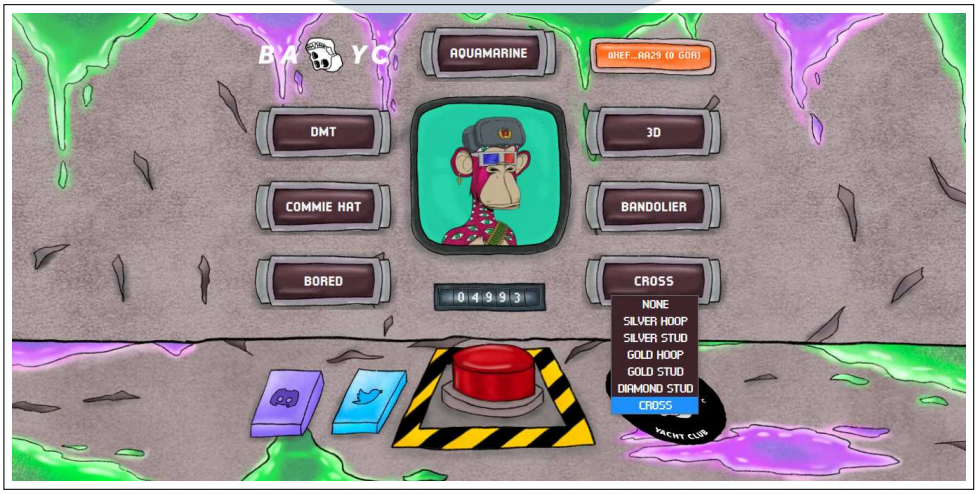
Gambar 3.7. Tampilan NFT yang sudah Berhasil Dilakukan Minting oleh Pengguna

Pada Gambar 3.7 ditampilkan NFT Buidl Ape Yacht Club yang sudah berhasil di-*minting* oleh pengguna.

3. Whitelist Mint

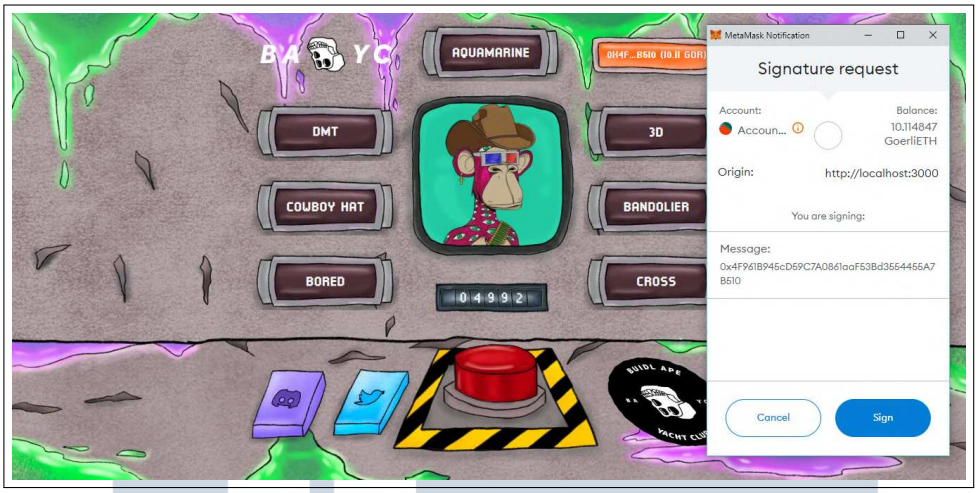
Gambar 3.8. Halaman Awal Dapps Buidl Ape Yacht Club

Pada Gambar 3.8 ini menampilkan halaman utama Buidl Ape Yacht Club yang terdapat tombol untuk melakukan *whitelist minting* NFT Buidl Ape Yacht Club.



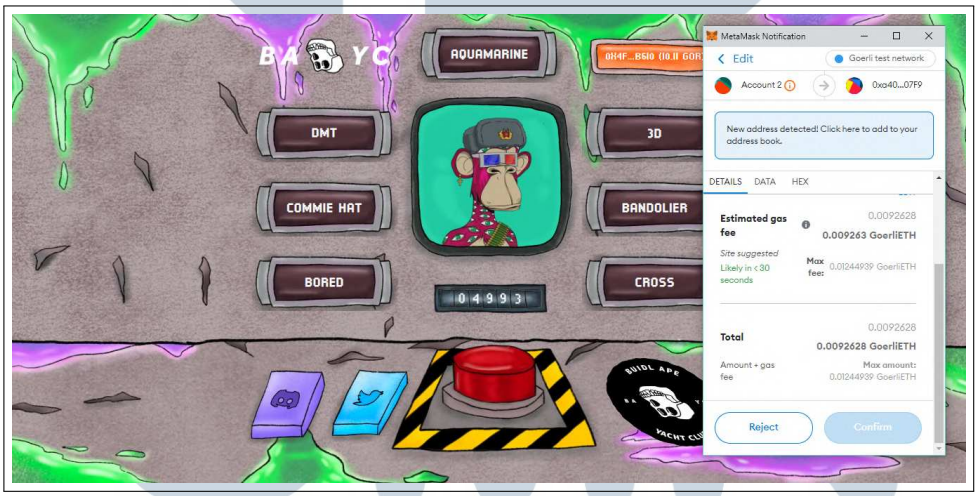
Gambar 3.9. Pemilihan Trait yang Diinginkan menggunakan Dropdown

Pengguna dapat memilih *trait* dari NFT yang mereka mau menggunakan *dropdown* sesuai pada Gambar 3.9.



Gambar 3.10. Halaman Konfirmasi Transaksi Whitelist Minting

Lalu pengguna perlu melakukan *sign message* untuk mendapatkan *signature* yang akan digunakan sebagai akses melakukan *whitelist minting* (Gambar 3.10).



Gambar 3.11. Halaman Konfirmasi Transaksi Whitelist Minting

Gambar 3.12. Modal Loading Transaksi Whitelist Minting

Lalu pengguna perlu mengonfirmasi transaksi (Gambar 3.11) dan halaman akan menampilkan *modal loading* yang dapat mengarahkan pada halaman etherscan (Gambar 3.12).

Gambar 3.13. Tampilan NFT yang sudah Berhasil Dilakukan Minting oleh Pengguna

Pada Gambar 3.13 ditampilkan NFT Buidl Ape Yacht Club yang sudah berhasil di-*minting* oleh pengguna.

B. Smart Contract

1. Public Mint

```
// @notice Mint an Ape NFT with concat uint64 trait values
function buidlApe(uint64 _traits)
    external payable
    whenNotPaused nonReentrant
{
    uint256 price = getPrice(totalSupply());

    require(Slice.isTraitsValid(Strings.toString(_traits)), "Invalid Traits!");

    require(totalSupply() <= MAX_SUPPLY, "Collection sold out!");

    require(tokenIdTraits[_traits] == 0, "The Ape already exists!");

    if (msg.value != price) revert InvalidPaymentAmount({
        expected: price,
        got: msg.value
    });

    // Mint action
    _safeMint(msg.sender, _traits);

    tokenIdTraits[_traits] = totalSupply();
}
```

Gambar 3.14. Kodingan Function Public Minting pada Smart Contract Buidl Ape Yacht Club

Pada Gambar 3.14, ditampilkan pembuatan fungsi *public minting* dengan beberapa validasi berupa pengecekan apakah angka *trait* valid, pengecekan *supply*, dan pengecekan apakah *trait* sudah pernah di-*minting* sebelumnya.

```
uint256 price = getPrice(totalSupply());
```

Gambar 3.15. Kodingan Function getPrice

Pada Gambar 3.15, ditampilkan kodingan untuk mendapatkan harga minting berdasarkan *threshold total supply* NFT yang sudah di-*minting*, semakin banyak NFT yang sudah di-*minting*, maka akan semakin tinggi harga NFT selanjutnya.

```
require(Slice.isTraitsValid(Strings.toString(_traits)), "Invalid Traits!");
```

Gambar 3.16. Kodingan Function isTraitsValid

Pada Gambar 3.16, ditampilkan kodingan untuk mengecek apakah angka *traits* yang telah dikirimkan valid dan memiliki 15 digit.

```
require(totalSupply() <= MAX_SUPPLY, "Collection sold out!");
```

Gambar 3.17. Kodingan Function Pengecekan Supply

Pada Gambar 3.17, ditampilkan kodingan untuk mengecek apakah *supply* NFT masih tersedia dan dapat di-*minting* oleh pengguna.

```
require(tokenIdTraits[_traits] == 0, "The Ape already exists!");
```

Gambar 3.18. Kodingan Function Pengecekan Traits

Pada Gambar 3.18, ditampilkan kodingan untuk mengecek apakah *traits* yang ingin di-*minting* sudah pernah di-*minting* sebelumnya atau tidak.

```
if (msg.value != price) revert InvalidPaymentAmount({
    expected: price,
    got: msg.value
});
```

Gambar 3.19. Kodingan Function Pengecekan Harga

Pada Gambar 3.19, ditampilkan kodingan untuk mengecek apakah harga yang dibayarkan pengguna sesuai dengan harga yang dibutuhkan untuk dilakukan proses *minting*.

```
// Mint action
_safeMint(msg.sender, _traits);
```

Gambar 3.20. Kodingan Function Minting

Pada Gambar 3.20, ditampilkan kodingan untuk melakukan *minting* NFT dengan angka *traits* sebagai parameternya.

```
Running 8 tests for test/BuidlApeTest/Mint.t.sol:MintTest
[PASS] testFailMintInvalidTraitsNone() (gas: 36674)
[PASS] testFailMintInvalidTraitsWithInvalidFirstDigit() (gas: 36620)
[PASS] testFailMintInvalidTraitsWithInvalidLength() (gas: 37134)
[PASS] testFailMintWrongEtherAmount(uint256) (runs: 256, μ: 39607, ~: 48217)
[PASS] testMintFailWithExistingTraits() (gas: 103028)
[PASS] testMintFailWithInvalidPrice() (gas: 77575)
[PASS] testMintSuccess() (gas: 88105)
[PASS] testMintSuccessWithPrice() (gas: 133582)
Test result: ok. 8 passed; 0 failed; finished in 25.52ms
```

Gambar 3.21. Hasil Unit Test Fungsi Public Mint

Dilakukan juga *unit test* untuk semua kasus dalam menggunakan fungsi *public minting* dengan hasil sesuai pada Gambar 3.21.

2. Whitelist Mint

```
// @notice Mint an Ape NFT with concat uint64 trait values and public signature
function buidlApeWhitelist(uint64 _traits, uint8 _v, bytes32 _r, bytes32 _s)
    external
    whenNotPaused nonReentrant
{
    require(Slice.isTraitsValid(Strings.toString(_traits)), "Invalid Traits!");

    require(totalSupply() <= MAX_SUPPLY, "Collection sold out!");

    require(tokenIdTraits[_traits] == 0, "The Ape already exists!");

    if(Signature.verify(1, msg.sender, _v, _r, _s) != signerAddress){
        revert SignatureNotValid();
    }

    // Mint action
    _safeMint(msg.sender, _traits);

    tokenIdTraits[_traits] = totalSupply();
}
```

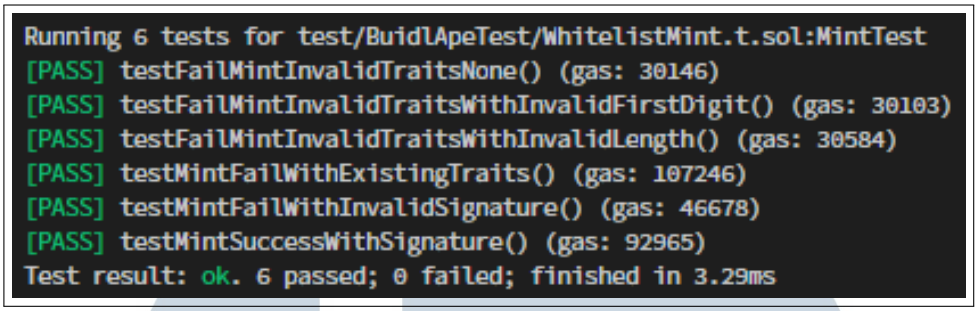
Gambar 3.22. Kodingan Function Whitelist Minting pada Smart Contract Buidl Ape Yacht Club

Pada Gambar 3.22, ditampilkan pembuatan fungsi *whitelist minting* dengan beberapa validasi berupa pengecekan apakah angka *trait* valid, pengecekan *supply*, pengecekan apakah *trait* sudah pernah di-*minting* sebelumnya, dan verifikasi *signature* untuk melakukan *whitelist minting*.

```
if(Signature.verify(1, msg.sender, _v, _r, _s) != signerAddress){
    revert SignatureNotValid();
}
```

Gambar 3.23. Kodingan Function Minting

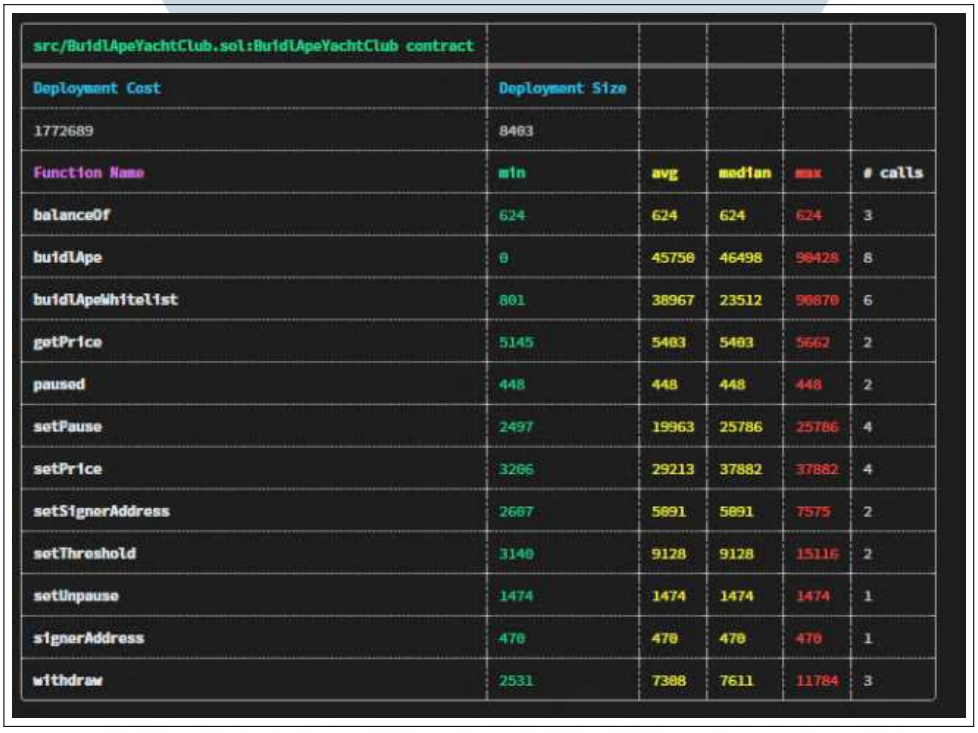
Pada Gambar 3.23, ditampilkan kodingan untuk melakukan verifikasi *signature* untuk dilakukan proses *whitelist minting* sebagai bukti bahwa pengguna terdaftar sebagai peserta *whitelist*.



Gambar 3.24. Hasil Unit Test Fungsi Whitelist Mint

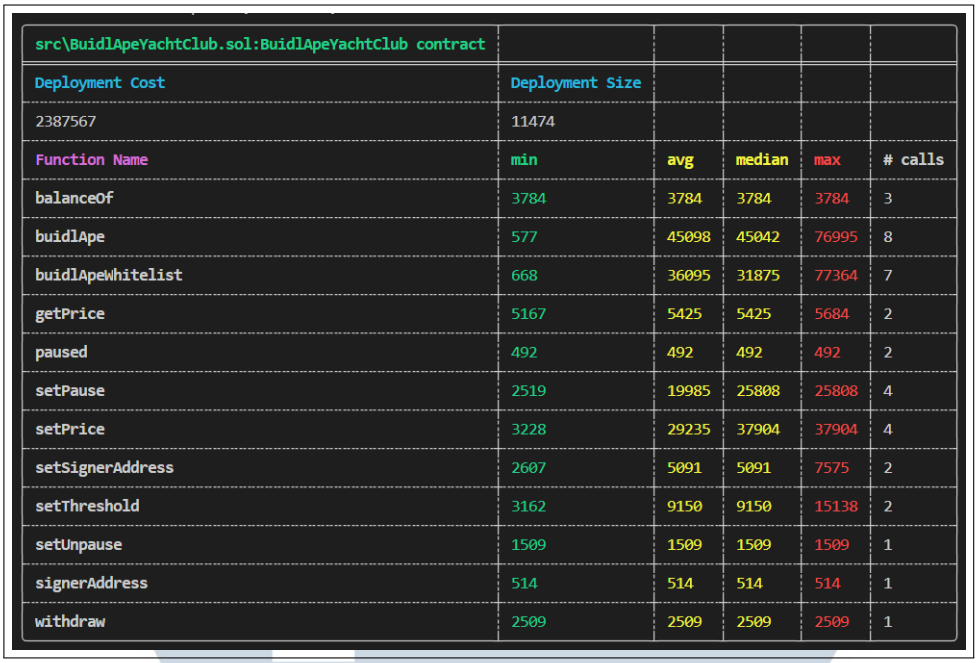
Dilakukan juga *unit test* untuk semua kasus dalam menggunakan fungsi *whitelist minting* dengan hasil sesuai pada Gambar 3.24.

C. Optimasi Gas Fee



src/BuidlApeYachtClub.sol:BuidlApeYachtClub contract					
Deployment Cost	Deployment Size				
1772689	8403				
Function Name	min	avg	median	max	# calls
balanceOf	624	624	624	624	3
buidlApe	0	45750	46498	90428	8
buidlApeWhitelist	801	38967	23512	90870	6
getPrice	5145	5403	5403	5662	2
paused	448	448	448	448	2
setPause	2497	19963	25786	25786	4
setPrice	3266	29213	37882	37882	4
setSignerAddress	2687	5891	5891	7575	2
setThreshold	3140	9128	9128	15116	2
setUnpause	1474	1474	1474	1474	1
signerAddress	470	470	470	470	1
withdraw	2531	7388	7611	11784	3

Gambar 3.25. Sebelum Optimasi Gas Fee menggunakan ERC721X



src\BuidlApeYachtClub.sol:BuidlApeYachtClub contract					
Deployment Cost	Deployment Size				
2387567	11474				
Function Name	min	avg	median	max	# calls
balanceOf	3784	3784	3784	3784	3
buidlApe	577	45098	45042	76995	8
buidlApeWhitelist	668	36095	31875	77364	7
getPrice	5167	5425	5425	5684	2
paused	492	492	492	492	2
setPause	2519	19985	25808	25808	4
setPrice	3228	29235	37904	37904	4
setSignerAddress	2607	5091	5091	7575	2
setThreshold	3162	9150	9150	15138	2
setUnpause	1509	1509	1509	1509	1
signerAddress	514	514	514	514	1
withdraw	2509	2509	2509	2509	1

Gambar 3.26. Hasil Optimasi Gas Fee menggunakan ERC721X

Optimasi *gas fee* berhasil dilakukan dengan mengubah basis *smart contract* dari ERC721A yang memiliki *gas fee* dengan kisaran 90 ribu wei (Gambar 3.25) menjadi ERC721X yang memiliki *gas fee* dengan kisaran 77 ribu wei (Gambar 3.26.

3.4 Kendala dan Solusi yang Ditemukan

3.4.1 Kendala

Dalam proses implementasi *frontend* dan *smart contract dapps* Buidl Ape Yacht Club pada Pixel8labs Pte. Ltd. ditemukan kendala sebagai berikut.

- Kurangnya bahan bacaan ataupun video pembelajaran penggunaan beberapa *library web3*
- Kesulitan dalam melakukan *gas fee optimization* saat menggunakan ERC721A pada smart contract Buidl Ape Yacht Club

3.4.2 Solusi

Solusi yang dilakukan untuk menyelesaikan kendala yang ditemukan adalah sebagai berikut.

- Membaca dokumentasi resmi dengan teliti dan bertanya pada orang yang sudah berpengalaman menggunakan *library web3* terkait
- Menggunakan ERC721X pada *smart contract* Buidl Ape Yacht Club untuk melakukan optimisasi *gas fee* sehingga lebih murah