

## BAB 4

## PEMBAHASAN

## 4.1 Implementasi

Dalam menjalankan penelitian ini peneliti merangkum beberapa spesifikasi, alat, dan versi yang dibutuhkan dalam penelitian ini yaitu

- Python versi 3.8 untuk menjalankan CMS
- Framework Django menggunakan Library Wagtail
- GCP Instance zona Asia-southeast1-a
- GCP Instance tipe mesin e2-medium
- Pragma solidity versi 0.4.24
- Token ERC-20

## 4.2 Membuat *Smart Contract*

#### 4.2.1 Modifikasi *Smart Contract*

```

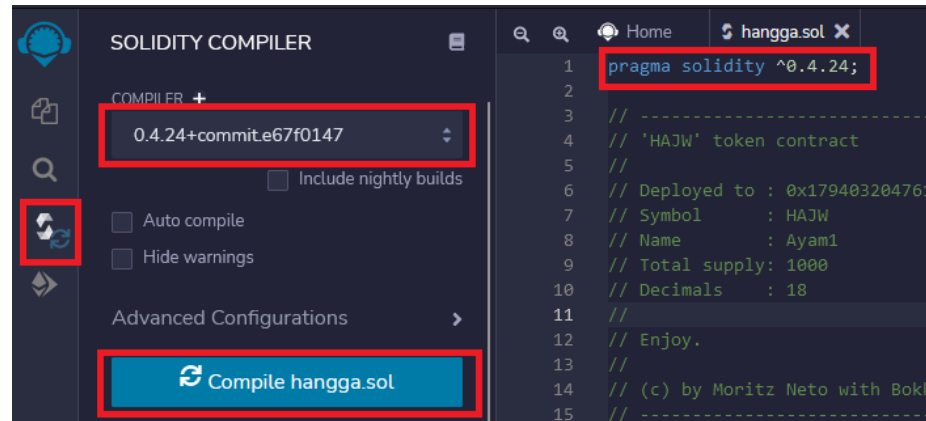
102 contract HAJW is ERC20Interface, Owned, SafeMath {
103     string public symbol;
104     string public name;
105     uint8 public decimals;
106     uint public _totalSupply;
107
108     mapping(address => uint) balances;
109     mapping(address => mapping(address => uint)) allowed;
110
111
112     // -----
113     // Constructor
114     // -----
115     constructor() public {
116         symbol = "HAJW";
117         name = "Ayami";
118         decimals = 18;
119         _totalSupply = 1000000000000000000;
120         balances[0x4c103be00E08978cB3e8A77498080F7622CE3075] = _totalSupply;
121         emit Transfer(0, 0x4c103be00E08978cB3e8A77498080F7622CE3075, _totalSupply);
122     }
123 }

```

### Gambar 4.1 Modifikasi Smart Contract

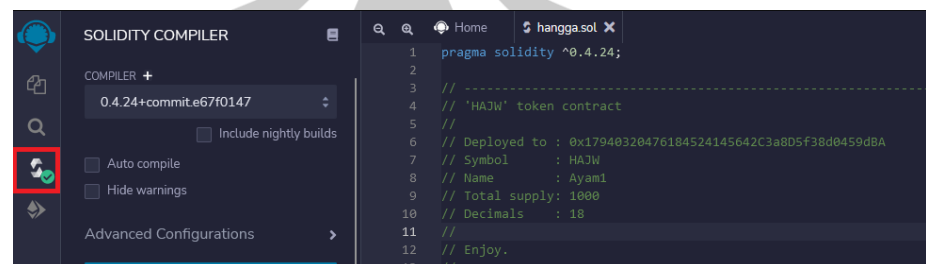
Pada **Gambar 4.1** peneliti membuat token HAJW dengan *smart contract* yang tertera pada poin **3.3** menggunakan bahasa solidity versi 0.4.24. Contoh *smart contract* bisa diakses melalui tautan berikut (<https://github.com/hangga/Prototype-Thesis/blob/main/Token/Skripsi/.workspaces/Skripsi2/hangga.sol>)

#### 4.2.2 Meng-compile Smart Contract



**Gambar 4.2** *Compile Smart Contract*

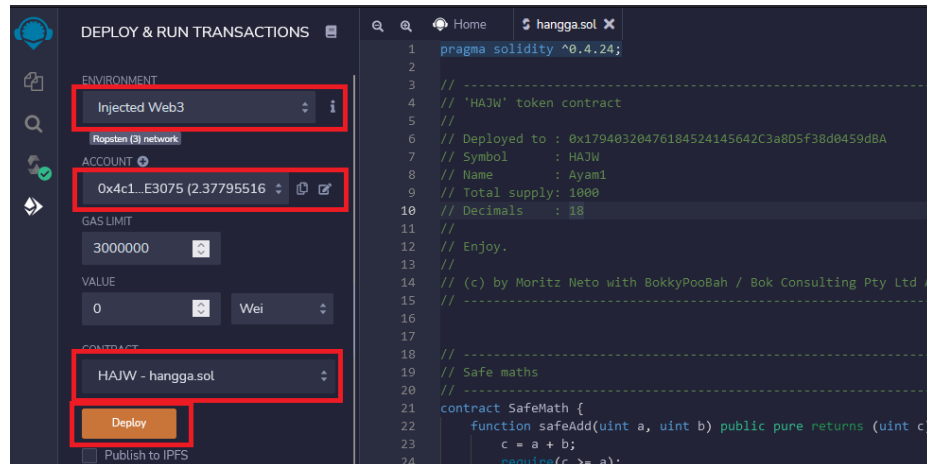
Pada **Gambar 4.2** sebelum peneliti meng-compile smart contract langkah yang harus dilakukan adalah mencocokkan versi compiler dengan versi solidity yang digunakan dalam smart contract.



**Gambar 4.3** *Sukses Compile Smart Contract*

Pada **Gambar 4.3** terlihat bahwa peneliti sukses meng-compile smart contract, jika ada kesalahan maka harus modifikasi ulang smart contract yang telah dibuat.

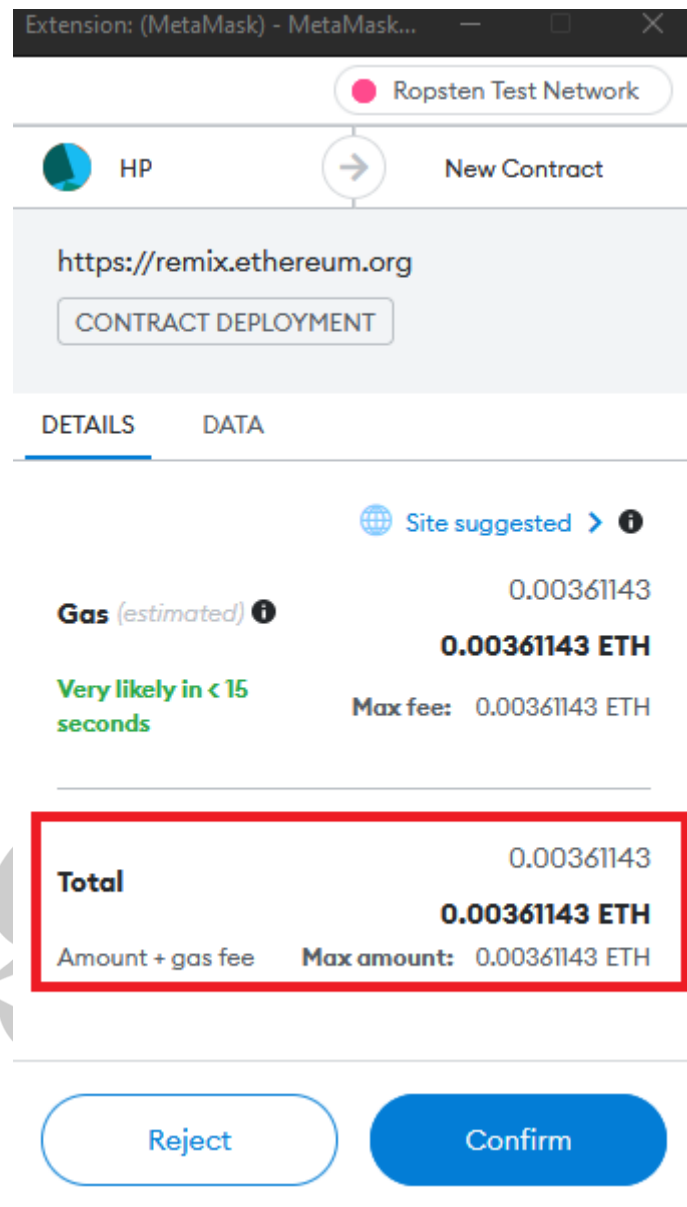
### 4.2.3 Deploy Smart Contract



**Gambar 4.4** Deploy Smart Contract Dengan Injected Web 3

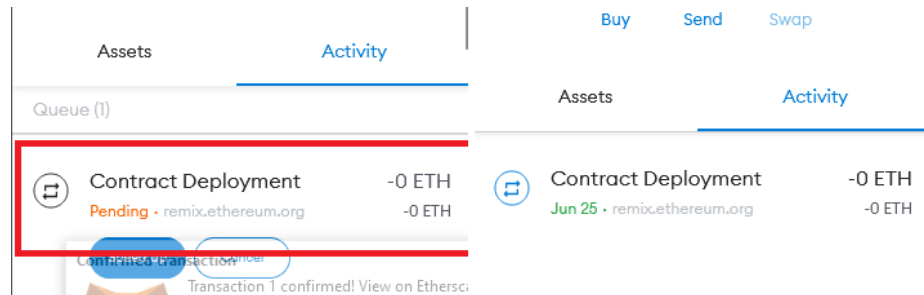
Pada **Gambar 4.4** peneliti memilih environment Injected Web3 yang terhubung dengan akun Ethereum yang berada di MetaMask untuk membayar biaya *deploy* dan berhubungan dengan blockchain. Lalu pastikan apakah akun yang terhubung sudah sesuai dan memiliki beberapa Ether untuk biaya pembuatan kontrak. Kemudian pilih kontrak yang telah dimodifikasi untuk di-*deploy*.

#### 4.2.4 Konfirmasi *Deploy Smart Contract*



**Gambar 4.5** Konfirmasi *Deploy Smart Contract*

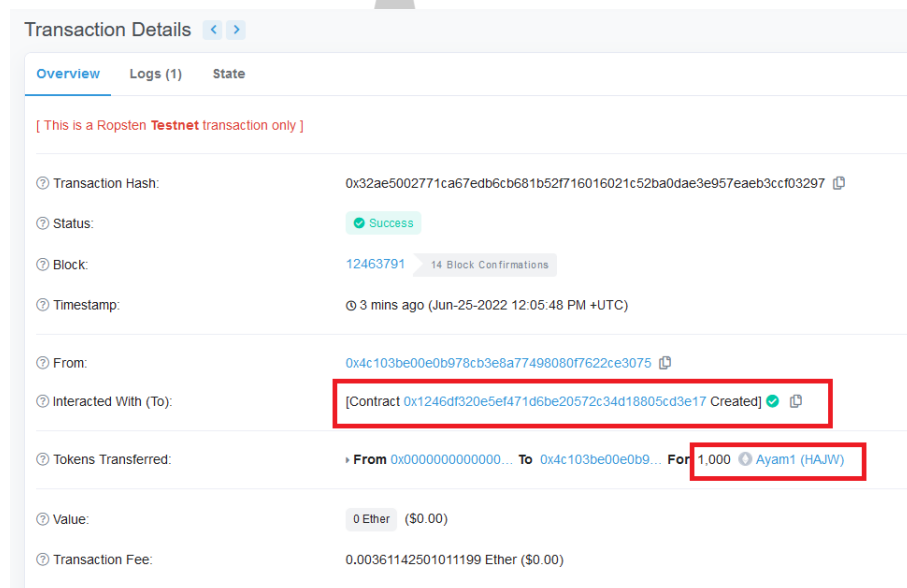
Pada **Gambar 4.5** terlihat bahwa proses *deploy smart contract* dalam pembuatan token membutuhkan biaya 0.00361143 ETH (Ether).



**Gambar 4.6** Aktivitas *Deploy Smart Contract*

Pada **Gambar 4.6** terlihat aktivitas setelah pembayaran biaya deploy yang selanjutnya bisa melanjutkan proses *deploy smart contract* ke dalam blockchain. Jika *deploy* telah selesai maka akan muncul *pop-up* MetaMask.

#### 4.2.5 Melihat Token



**Gambar 4.7** Token Berhasil Dibuat

**Token Ayam1**

Overview [ERC-20]		Profile Summary	
Max Total Supply:	1,000 HAJW	Contract:	0x1246df320e5e471d5be20572c34d18805cd3e17
Holders:	1	Decimals:	18
Transfers:	1		

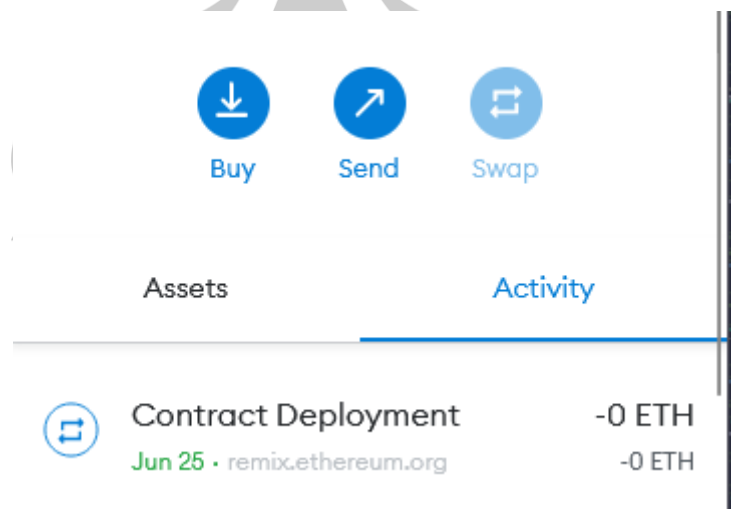
Transfers					
Txn Hash	Method	Age	From	To	Quantity
0x32ae5002771ca67ed0...	0x00000000	5 mins ago	0x000000000000000000000000...	0x4c103be00e0b978cb3...	1,000

**Gambar 4.8** HAJW Token

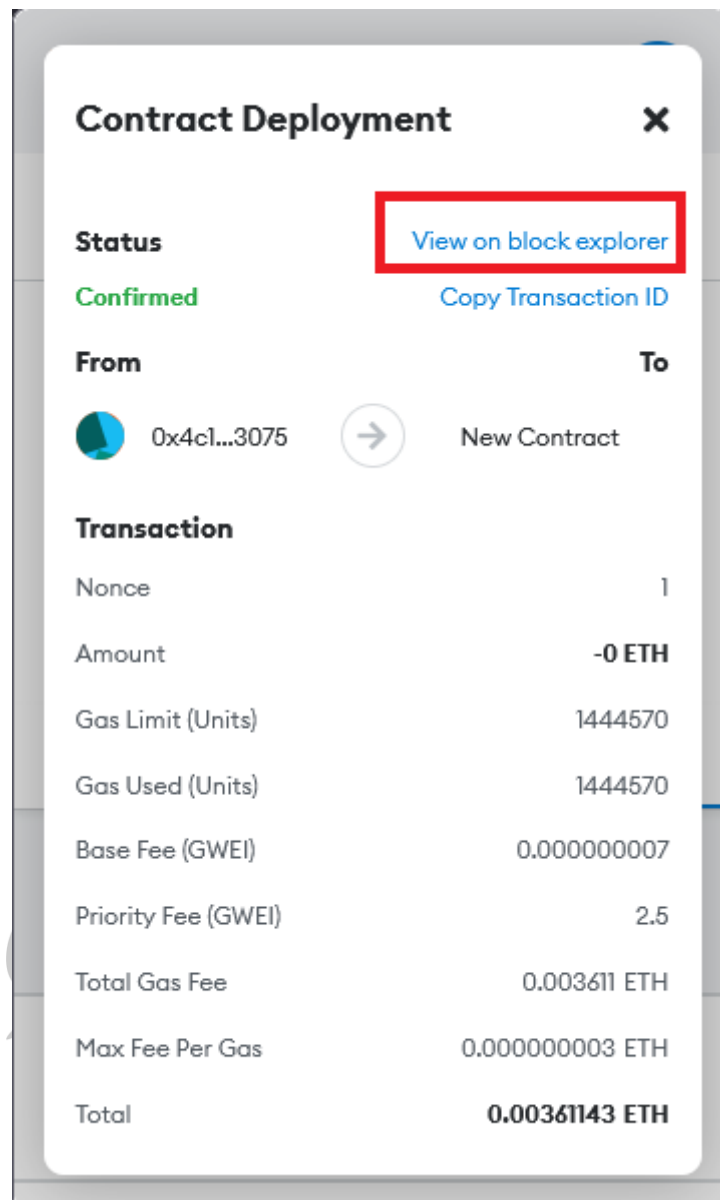
Pada **Gambar 4.7** dan **Gambar 4.8** terlihat bahwa peneliti berhasil membuat token Ayam1 dengan simbol HAJW sebanyak 1000 supply dengan alamat contract tertera pada gambar.

### 4.3 Pasang Token di MetaMask

#### 4.3.1 Detail Contract Deployment



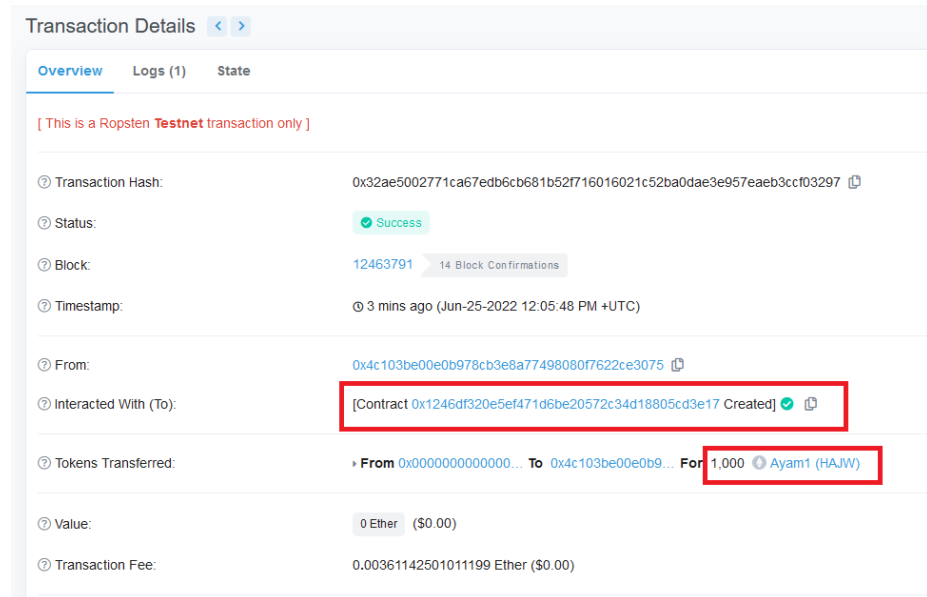
**Gambar 4.9** Aktivitas Contract Deployment



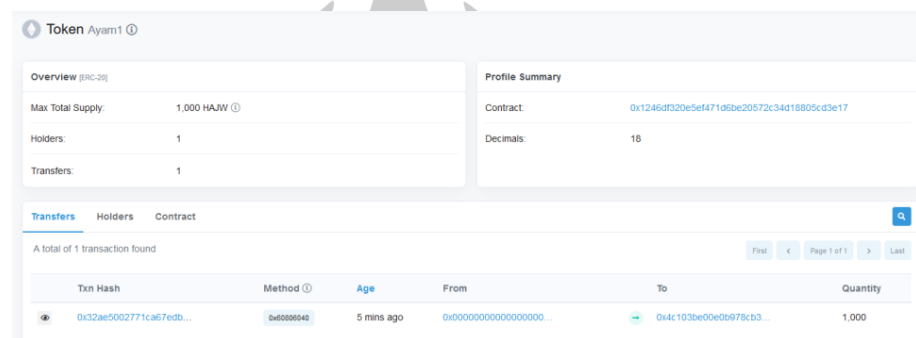
**Gambar 4.10** Detail *Contract Deployment*

Pada **Gambar 4.9** dan **Gambar 4.10** terlihat aktivitas *deployment smart contract* yang telah dibuat oleh peneliti. Untuk melihat data data blockchain silakan pilih View on block explorer

### 4.3.2 Token HAJW



**Gambar 4.11** Txn Deployment Smart Contract

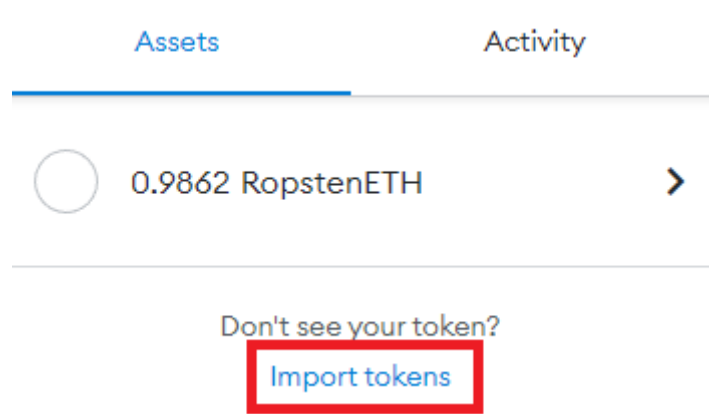


**Gambar 4.12** Token HAJW

Pada **Gambar 4.11** dan **Gambar 4.12** terlihat bahwa token HAJW memiliki alamat kontrak (0x1246df320E5EF471d6bE20572C34D18805cd3E17) untuk di-import ke MetaMask peneliti.



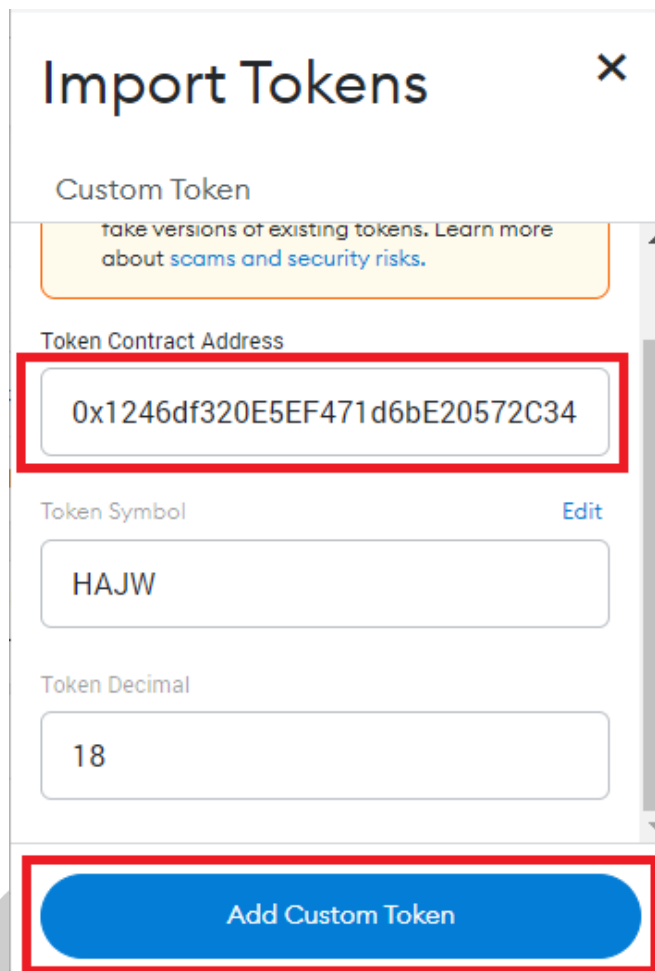
### 4.3.3 Import Token



**Gambar 4.13** Tampilan MetaMask

Pada **Gambar 4.13** pilih *Import tokens* untuk meng-*impor* token HAJW.





**Import Tokens** ×

Custom Token

take versions of existing tokens. Learn more about [scams](#) and [security risks](#).

Token Contract Address

0x1246df320E5EF471d6bE20572C34

Token Symbol Edit

HAJW

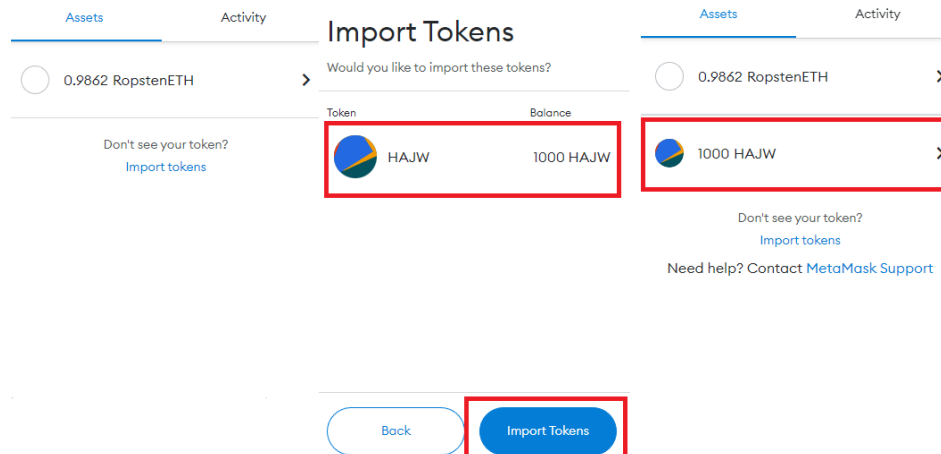
Token Decimal

18

Add Custom Token

**Gambar 4.14** Isi Token Contract Address

Pada **Gambar 4.14** peneliti menempelkan alamat kontrak token HAJW ke dalam kolom *Token Contract Address* setelah itu informasi mengenai token akan muncul.



**Gambar 4.15** Impor Token Berhasil

Pada **Gambar 4.15** terlihat bahwa alamat kontrak sesuai dengan token yang telah dibuat dari *smart contract* sebelumnya, lalu peneliti memilih *button Import Tokens* untuk mengimport token HAJW ke dalam akun MetaMask peneliti.




## 4.4 Input CMS


The image displays a CMS interface with three stacked input forms for a web page layout. Each form has an 'ADVANCED SETTINGS' toggle and a trash icon.


**Form 1: Responsive Grid Row**

- Responsive Grid Row
- Full width: ☐
- Content: +

**Form 2: Card**

- Card
- Image:  CLEAR CHOICE CHANGE IMAGE EDIT THIS IMAGE
- Title: Yakisoba
- Subtitle:
- Body:
 

From:  retail1.eth

To:  okaeri.eth
- Links: +

**Form 3: Button Link**

- Button Link
- Page link: CHOOSE A PAGE
- Document link: CHOOSE A DOCUMENT
- Other link: <https://ropsten.etherscan.io/tx/0x>
- Button Title: Txn
- Button Style: Outline Danger
- Button Size: Default

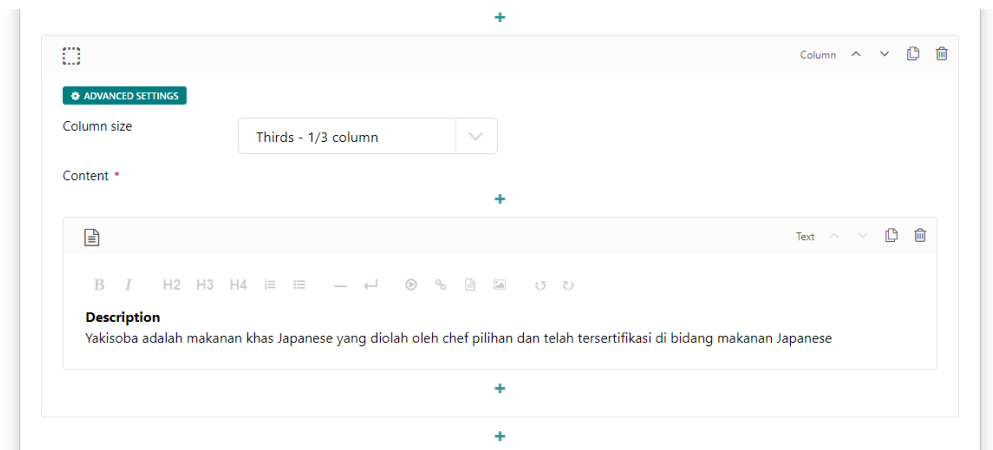
**Form 4: Button Link**

- Button Link
- Page link: CHOOSE A PAGE
- Document link: CHOOSE A DOCUMENT
- Other link: <http://34.124.208.72:8000/hajw-pr>
- Button Title: Asal Bahan Baku
- Button Style: Outline Success
- Button Size: Default

**Gambar 4.16** Kolom 1 Halaman Web

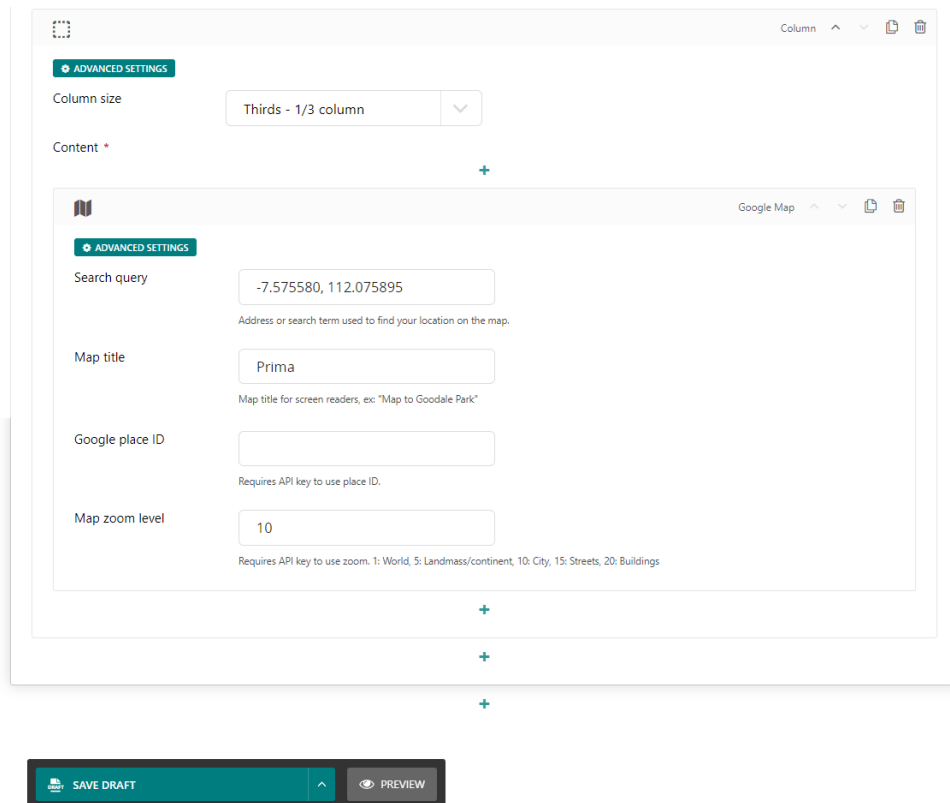
Pada **Gambar 4.16** merupakan isi dari kolom 1 yang dienkapsulasi dalam elemen card yang berisi

- Foto barang yang dibeli oleh pelanggan
- Nama barang yang dibeli oleh pelanggan
- Tautan blockchain pengirim dan penerima pelaku *supply chain*
- Tautan Txn blockchain
- Tautan referensi dari supply chain sebelumnya



**Gambar 4.17** Kolom 2 Halaman Web

Pada **Gambar 4.17** merupakan isi dari kolom 2 yang mendeskripsikan barang yang dibeli oleh pelanggan.



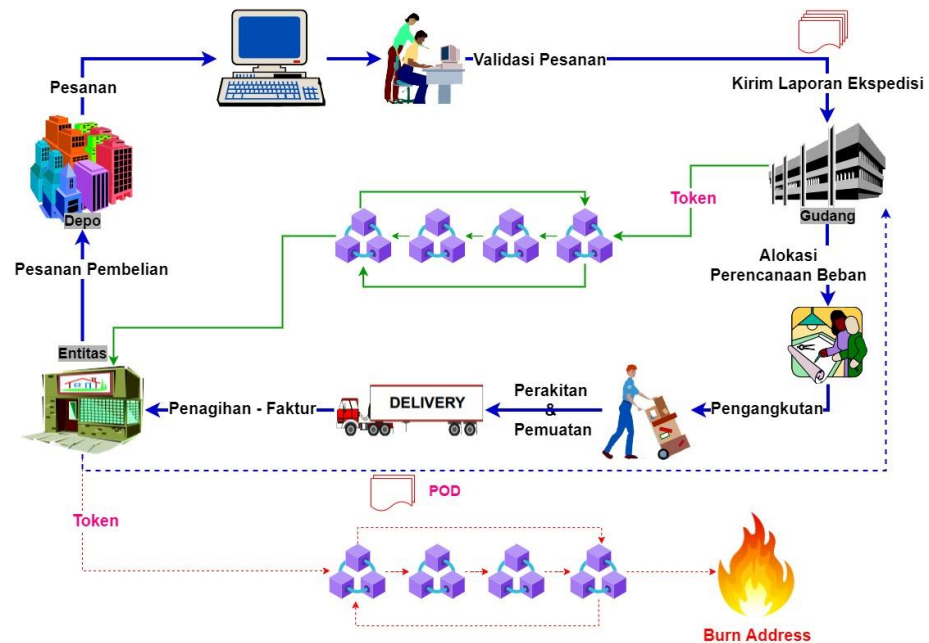
**Gambar 4.18** Kolom 3 Halaman Web

Pada **Gambar 4.18** merupakan isi dari kolom 3 yang menampilkan map dari entitas sebelumnya dalam supply chain.

Konten yang dimasukkan ke dalam halaman web merupakan hak dari masing-masing entitas, peneliti membuat isi konten seperti di atas dikarenakan isi konten tersebut sudah sangat transparan untuk bisa dipahami oleh pelanggan dan kepentingan auditing.

## 4.5 Proses Logistik dan Transport

### 4.5.1 Proses Rantai Pasok



**Gambar 4.19** Proses Rantai Pasok

**Gambar 4.19** merupakan proses rantai pasok dan mekanisme pengangkutan yang dipakai saat proses pengiriman barang dan validasi pesanan. Entitas melakukan Pesanan Pembelian (*Purchase Order*) ke pada Depo untuk memesan barang dan Depo melakukan validasi pesanan. Dari Depo mengirimkan dokumen ekspedisi ke gudang untuk mengatur pesanan sesuai dengan dokumen, lalu gudang akan mengalokasikan dan merencanakan beban barang sesuai dengan pesanan. Saat proses alokasi dan perencanaan beban pihak gudang dibantu oleh pihak ekspedisi untuk validasi pesanan saat proses pengangkutan di dalam kendaraan. Setelah barang diangkut di dalam kendaraan lalu diantarkan ke Entitas akan terjadi proses penerimaan barang di Entitas dan terjadi proses validasi pesanan apakah sudah sesuai atau belum. Jika tidak sesuai maka akan terjadi proses POD (Proof of Delivery) yang di mana pihak Entitas dan Depo harus mengisi formulir sesuai dengan kasus yang ada seperti barang rusak/hilang/tertukar, kurang dalam produk (*missed in product*) dan kasus lainnya. Jika saat proses POD ditemui

barang yang hilang dan lainnya seperti pada tabel 4.1 maka token harus dibakar sesuai dengan catatan kehilangan atau kerusakan barang yang ada pada dokumen POD, dan jika saat proses POD ditemui kelebihan barang maka barang harus dikembalikan sesuai dengan catatan POD.

#### 4.5.2 Aktivitas POD (*Proof of Delivery*)

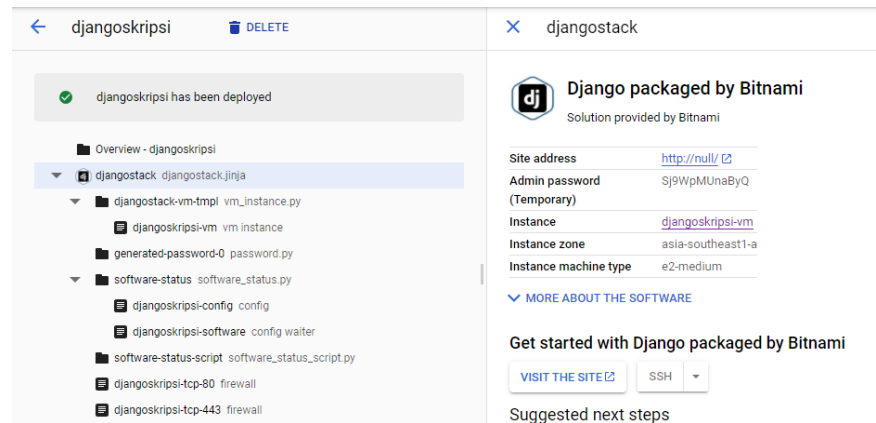
**Tabel 4.1** Aktivitas POD

No	Point	
<b>1</b>	<b>Lead Time Evaluation</b>	
	<b>a</b>	Gunakan form “Laporan Penerimaan”
	<b>b</b>	Kirim form melalui email ke Depo
	<b>c</b>	Kirim hardcopy ke Depo
<b>2</b>	<b>Barang Rusak/Hilang/Tertukar</b>	
	<b>a</b>	Menggunakan form “X1” yang harus ditanda tangani lengkap oleh ASM, Dist Gudang, Sopir
	<b>b</b>	Wajib menandatangani dan mengembalikan nota retur
<b>3</b>	<b>Kurang Dalam Karton</b>	
	<b>a</b>	Menggunakan form “X2”
	<b>b</b>	Wajib melampirkan sobekan kode produksi
	<b>c</b>	Pengecekan barang kurang dalam karton harus disaksikan oleh ASM langsung
	<b>d</b>	Wajib menandatangani dan mengembalikan nota retur

#### 4.6 Penggunaan Layanan GCP

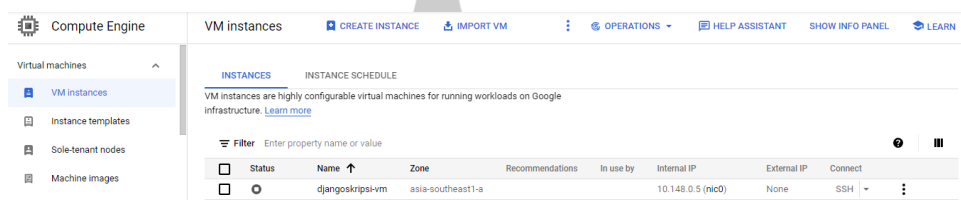
Pada penelitian ini peneliti menggunakan layanan cloud GCP untuk mendeploy CMS supaya server bersifat online. Peneliti menggunakan deployment manager Djangostack dari Bitnami untuk membangun environment server sebagai pendukung jalannya server CMS.





**Gambar 4.20** Tampilan Deployment Manager Djangostack

Environment tersebut menjalankan Instance server dan VPC guna mendukung apapun yang akan dilakukan oleh peneliti saat menjalankan Instance server.










**Gambar 4.21** Tampilan Instance Server CMS Peneliti

Terlihat bahwa di instance server terdapat

- Name merupakan nama instance compute engine milik peneliti yaitu djangostack-vm
- Zone merupakan letak zona wilayah instance compute engine milik peneliti berada di asia-southeast1-a (Zona wilayah asia tenggara)
- Internal IP merupakan IP yang digunakan untuk kegiatan internal mengatur instance server
- External IP merupakan bagian terpenting dalam penelitian ini dikarenakan external IP akan digunakan oleh user untuk mengakses CMS yang telah diatur oleh peneliti, dan konsumen untuk melihat halaman situs antar entitas.

e. Connect merupakan tempat untuk menghubungkan ke SSH server.

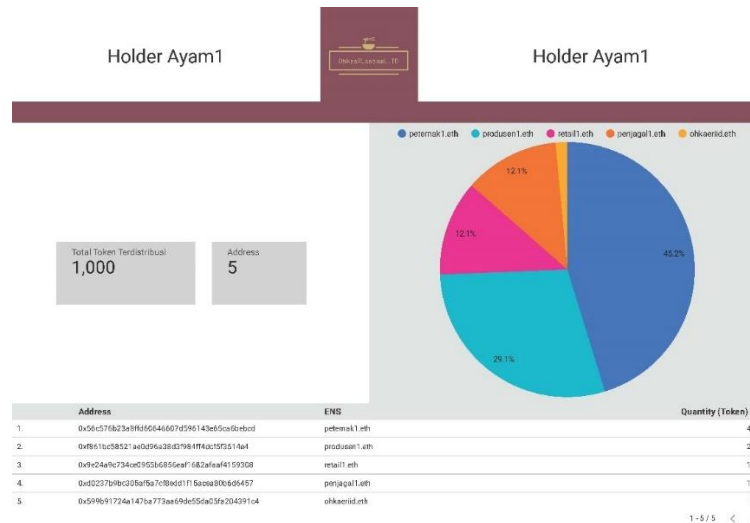
	Firewall	 Filter Enter property name or value					
	Routes	<input type="checkbox"/>	Name	Type	Targets	Filters	Protocols / ports
	VPC network peering	<input type="checkbox"/>	default-allow-http	Ingress	http-server	IP ranges: 0.0.0.0/0	tcp:80
	Shared VPC	<input type="checkbox"/>	djangoskripsi-tcp-443	Ingress	djangoskripsi-tcp-443	IP ranges: 0.0.0.0/0	tcp:443
	Serverless VPC access	<input type="checkbox"/>	djangoskripsi-tcp-80	Ingress	djangoskripsi-tcp-80	IP ranges: 0.0.0.0/0	tcp:80
	Packet mirroring	<input type="checkbox"/>	djangoskripsi-tcp8000	Ingress	djangoskripsi-tcp8000	IP ranges: 0.0.0.0/0	tcp:8000

**Gambar 4.22** Tampilan Pengaturan *Firewall*

Terlihat bahwa di VPC network terdapat pengaturan jaringan agar pengguna bisa mengakses external IP

- Name merupakan nama pengaturan firewall
- Type merupakan pilihan apakah jaringan berintegrasi dengan layanan jaringan GCP yang lain
- Targets merupakan sasaran manakah yang akan diintegrasikan dengan pengaturan firewall yang telah diatur.
- Filters berisi range IP yang akan digunakan oleh peneliti untuk menjalankan server
- Protocols/port berisi port angka yang fungsinya agar user lain bisa melihat tampilan situs CMS.


#### 4.7 Grafik Pemegang Token



**Gambar 4.23** Grafik Pemegang Token HAJW

**Gambar 4.23** merupakan gambaran dari grafik pemegang token yang digunakan di Tugas Akhir ini. Dengan adanya grafik ini diharapkan para pemangku kepentingan atau konsumen bisa melihat jumlah token yang sudah terdistribusi di dalam proses *Supply Chain* maupun untuk pengambilan keputusan lainnya.

#### 4.8 Cetak QR Code



Hangga Generator

Enter URL

File Name

**Gambar 4.24** Tampilan Sistem Generator QR Code

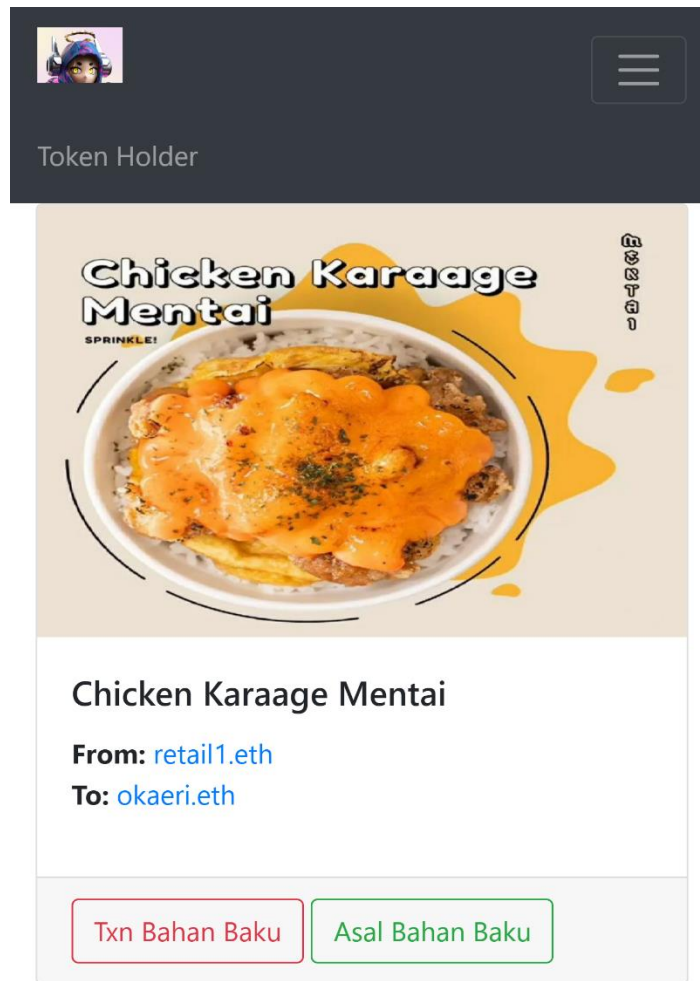
Pada **Gambar 4.24** merupakan tampilan sistem generator QR code yang berisi kolom URL dan Nama file untuk diisi oleh pengguna guna mendapatkan QR Code seperti pada **Gambar 4.25**



**Gambar 4.25** Hasil QR Code

Pada **Gambar 4.25** QR Code tersebut akan menampilkan tautan dan memindahkan pelanggan ke tautan yang sudah di-*input* lalu pelanggan bisa melihat proses supply chain mulai dari tempat peternak bahan baku ayam sampai diolah di restoran.

#### 4.9 Tampilan Website



##### Description

Chicken Karaage Mentai adalah makanan khas Japanese yang dibalut dengan saos mentai pilihan racikan dari juru masak terlatih



**Gambar 4.26** Tampilan Website Setelah User Memindai QR Code

Setelah user memindai QR Code maka akan muncul halaman web seperti **Gambar 4.26** yang berisi sesuai dengan apa yang telah di-input dalam CMS seperti foto barang yang dibeli, informasi dalam blockchain, deskripsi barang, peta tempat entitas supply chain sebelumnya. User bisa melihat informasi dalam blockchain saat memilih *button* Txn Bahan Baku untuk memastikan apakah memang benar antar alamat Ethereum pelaku supply chain saling bekerja sama.

