Laporan Tugas Besar II4035
Blockchain

Platform Tiket Konser Terdesentralisasi Berbasis Blockchain dan NFT

Dosen Pengampu :

Ir. Yudistira Dwi Wardhana Asnar, S.T, Ph.D.

Dr. Phil. Eng. Hari Purnama, S.Si., M.Si.

Disusun Oleh :

Frankie Huang	13521092
Fawwaz Abrial Saffa	18221067
Abraham Megantoro Samudra	18221123
Lie, Kevin Sebastian S. T.	18221143

Program Studi Sistem dan Teknologi Informasi

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 4013

DAFTAR ISI

I DESKRIPSI PROGRAM

Program ini adalah platform tiket berbasis *blockchain* untuk penjualan tiket konser dan acara *fan meeting*. Platform ini menggunakan *smart contract* yang berjalan di blockchain Ethereum Virtual Machine (EVM) dan mendukung token NFT (Non-Fungible Token) berbasis standar ERC-721. NFT digunakan untuk merepresentasikan setiap tiket, sehingga memastikan transparansi, keamanan, dan kepemilikan unik.

Fitur utama platform ini mencakup:

1. Pembuatan Tiket

   Tiket yang dihasilkan sebagai NFT dengan metadata yang mencakup informasi acara, lokasi tempat duduk, tanggal, dan harga tiket.
2. Pasar Sekunder

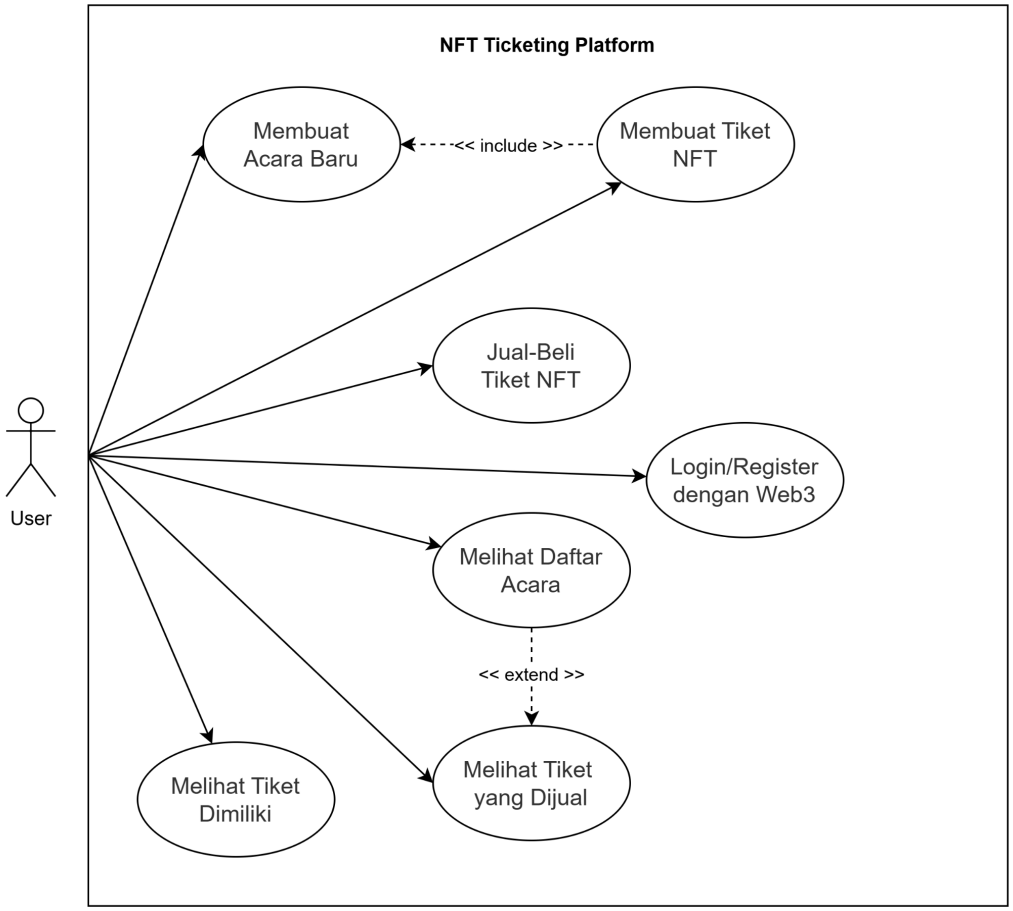
   Tiket dapat diperjualbelikan kembali melalui *smart contract* yang menjamin transparansi dan keamanan transaksi.
3. Pengelolaan Acara

   Pengguna dapat membuat dan mengelola acara melalui antarmuka web.

Platform ini juga memiliki beberapa keunggulan yang dapat digunakan untuk menyelesaikan beberapa permasalahan dalam penjualan tiket, yaitu:

1. Menghilangkan pemalsuan tiket dengan teknologi tanda tangan digital.
2. Memberikan transparansi penuh melalui pencatatan transaksi di blockchain.
3. Memungkinkan pembatasan dan pembagian kuota tiket untuk berbagai kelompok, seperti sponsor dan VIP.

Platform ini merupakan solusi inovatif untuk mengatasi masalah klasik dalam distribusi tiket, seperti kelebihan pemesanan, pemalsuan, dan ketidakadilan distribusi. Adapun *use case* dari sistem ini yang akan digambarkan pada gambar di bawah ini.



I.1 Platform Blockchain

Ethereum dipilih sebagai platform *blockchain* untuk proyek ini karena ekosistemnya yang matang, fleksibilitas, dan dukungan luas terhadap pengembangan aplikasi terdesentralisasi (DApp). Ethereum adalah platform *blockchain* yang pertama kali memperkenalkan konsep *smart contract*, memungkinkan pengembang untuk membuat aplikasi yang aman dengan memanfaatkan bahasa pemrograman Solidity. Standar ERC-721 yang digunakan untuk membuat Non-Fungible Token (NFT) juga berasal dari Ethereum, sehingga menjadikannya platform yang ideal untuk aplikasi berbasis NFT seperti *marketplace* tiket ini.

Selain itu, penggunaan Kurtosis memungkinkan pengaturan *private network* Ethereum secara cepat dan efisien. Kurtosis menyediakan lingkungan kontainer yang mempermudah simulasi jaringan *blockchain* untuk pengujian skala kecil hingga besar tanpa memerlukan biaya besar seperti di jaringan publik. Dengan Kurtosis, pengembang dapat mengontrol penuh konfigurasi node, konsensus, dan mekanisme transaksi dalam jaringan blockchain pribadi mereka.

I.2 Tech Stack

Dalam pengembangan proyek ini, terdapat berbagai *tech stack* yang digunakan berdasarkan komponen yang dikembangkan.

I.2.1 Smart Contract

Solidity digunakan untuk mengembangkan *smart contract* karena merupakan bahasa standar yang didukung oleh Ethereum dan kompatibel dengan EVM (Ethereum Virtual Machine). Bahasa ini dirancang untuk membuat aplikasi desentralisasi yang aman, efisien, dan dapat diandalkan. Solidity juga memungkinkan integrasi dengan berbagai library seperti OpenZeppelin untuk mempercepat pengembangan *smart contract*.

Selain itu, Hardhat digunakan sebagai *framework* pengembangan untuk mempermudah penulisan, pengujian, dan deployment *smart contract*.

I.2.2 Oracle

Oracle adalah perantara yang menghubungkan *smart contract* dengan data eksternal atau *off-chain*. JavaScript digunakan untuk pengembangan Oracle yang berfungsi untuk menghubungkan blockchain dengan data eksternal secara aman.

I.2.3 Client

Next.js digunakan untuk pengembangan antarmuka pengguna serta implementasi REST API untuk menghubungkan aplikasi ke database menggunakan Drizzle ORM dan PostgreSQL.

I.3 High Level Architecture

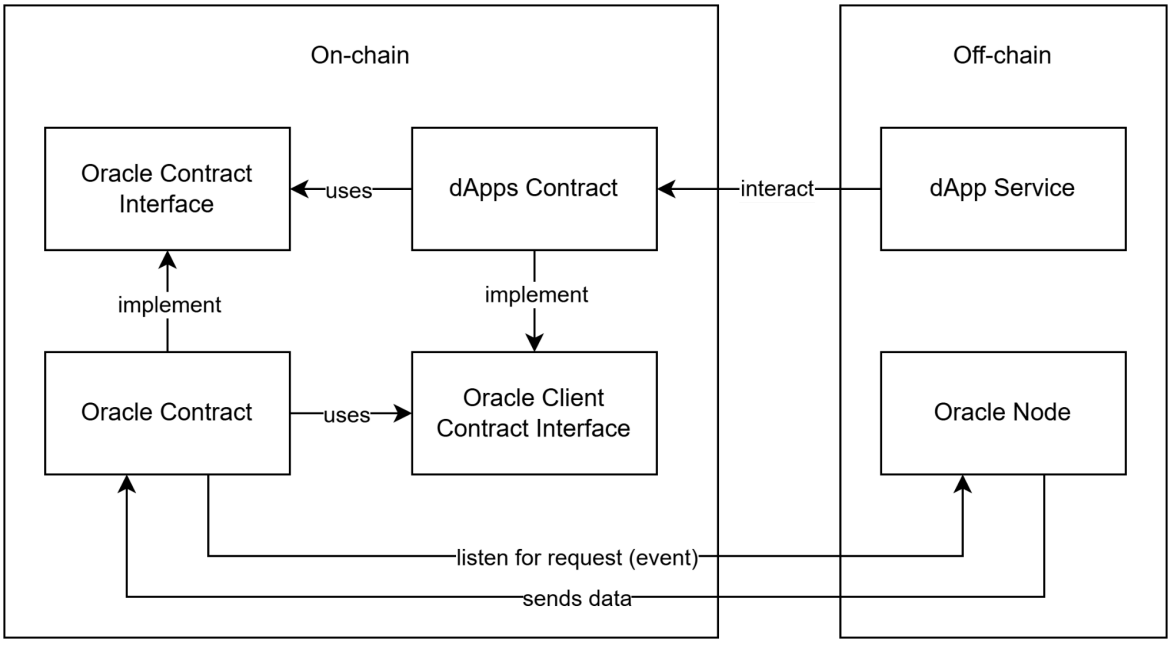


Diagram diatas menggambarkan arsitektur sistem yang mengintegrasikan komponen *on-chain* dan *off-chain* melalui mekanisme Oracle. Pada sisi *on-chain*, terdapat beberapa kontrak pintar, yaitu dApps Contract yang menjadi inti aplikasi terdesentralisasi, Oracle Client Contract Interface yang mendefinisikan standar komunikasi dengan oracle, Oracle Contract Interface sebagai dasar implementasi oracle, dan Oracle Contract yang menangani permintaan data eksternal. dApps Contract mengirim permintaan data melalui Oracle Client Contract Interface ke Oracle Contract, yang kemudian memicu event untuk memberi tahu komponen *off-chain* tentang permintaan tersebut. Di sisi *off-chain*, Oracle Node mendengarkan *event* ini, mengambil data dari sumber eksternal, dan mengirimkannya kembali ke Oracle Contract. Data ini kemudian diteruskan ke dApps Contract untuk digunakan dalam aplikasi blockchain. Selain itu, dApp Service juga dapat berinteraksi dengan dApps Contract untuk pengolahan data tambahan. Arsitektur ini memungkinkan blockchain, yang bersifat deterministik, untuk mengakses data eksternal secara aman dan terpercaya melalui oracle.

I.4 System Requirement

Dalam mengimplementasikan sebuah sistem, penting untuk mengidentifikasi kebutuhan sistem yang mencakup kebutuhan fungsional dan non fungsional. Pada bagian ini, akan dijelaskan secara rinci terkait kebutuhan fungsional dan non fungsional.

ID	Kebutuhan Fungsional	Deskripsi
FR-1	Sistem dapat membuat acara baru	Sistem memungkinkan penyelenggara acara untuk membuat acara baru melalui antarmuka web. Penyelenggara dapat menginput beberapa informasi terkait acara.
FR-2	Sistem dapat membuat tiket NFT	Sistem memungkinkan penyelenggara acara untuk membuat tiket dalam bentuk NFT berbasis standar ERC-721, mencakup beberapa informasi terkait tiket.
FR-3	Sistem dapat mendukung transaksi jual-beli tiket NFT	Sistem harus memungkinkan pengguna untuk menjual dan membeli tiket melalui marketplace menggunakan smart contract.
FR-4	Sistem dapat menampilkan semua acara yang ada beserta informasi detailnya	Sistem menyediakan halaman daftar acara yang menampilkan semua acara yang tersedia beserta informasi detail untuk setiap acara

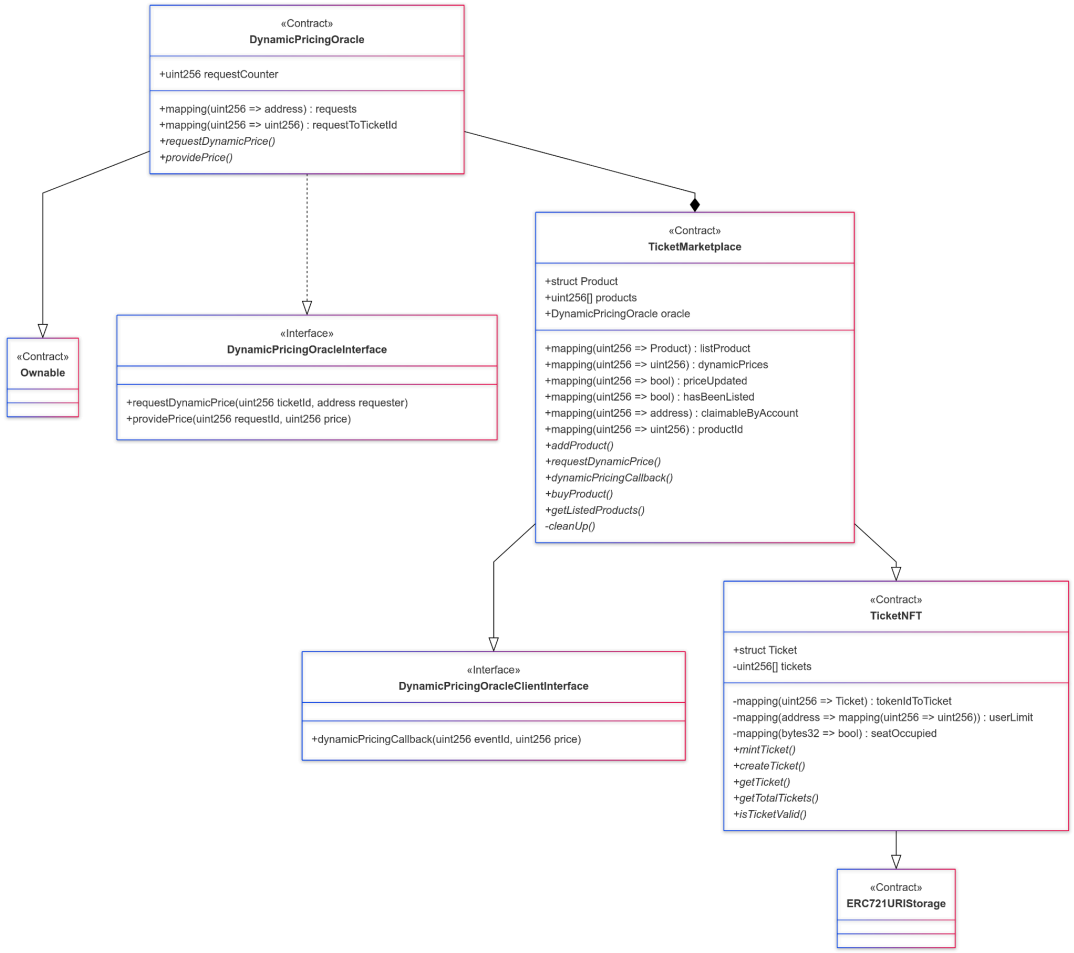
FR-5	Sistem dapat melakukan login dan register	Sistem menyediakan fitur login dan register untuk pengguna. Autentikasi berbasis Web3, memungkinkan pengguna menggunakan dompet kripto (seperti MetaMask) untuk login.
FR-6	Sistem dapat menampilkan tiket yang dimiliki	Sistem menampilkan daftar tiket yang dimiliki pengguna dalam dompet kripto mereka dan informasi terkait masing-masing tiket.
FR-7	Sistem dapat menampilkan tiket yang sedang dijual dari sebuah acara	Sistem menampilkan daftar tiket dari acara tertentu yang sedang dijual di pasar sekunder. Pengguna dapat membeli tiket langsung dari daftar ini.

ID	Kebutuhan Non-Fungsional	Deskripsi
NR-1	Keamanan	Menggunakan protokol keamanan blockchain standar untuk melindungi transaksi dan data pengguna.
NR-2	Skalabilitas	Sistem dirancang untuk mendukung banyak acara dan transaksi secara paralel tanpa penurunan performa.
NR-3	Kompatibilitas	Mendukung standar ERC-721 dan kompatibel dengan dompet Ethereum populer.
NR-4	Auditabilitas	Seluruh aktivitas tercatat di blockchain dan dapat diaudit secara publik.

II IMPLEMENTASI

II.1 Smart Contract

Berikut adalah deskripsi dari *smart contract* yang dirancang berdasarkan *class diagram* yang diberikan. Sistem ini terdiri dari beberapa komponen utama, yaitu DynamicPricingOracle, TicketMarketplace, TicketNFT, dan beberapa *interface* pendukung yang memastikan modularitas antar komponen.



Smart contract DynamicPricingOracle bertanggung jawab sebagai Oracle untuk menyediakan harga dinamis bagi produk yang ada di *marketplace*. Oracle ini mengimplementasikan *interface* DynamicPricingOracleInterface, yang mendefinisikan fungsi utama seperti requestDynamicPrice untuk menerima permintaan harga dinamis dan providePrice untuk memberikan harga tersebut. Contract ini juga mewarisi Ownable, yang memberikan kontrol administratif hanya kepada pemilik *contract*.

Di sisi *marketplace*, TicketMarketplace bertindak sebagai platform jual beli tiket. Marketplace ini menggunakan DynamicPricingOracle untuk mendapatkan harga tiket secara dinamis berdasarkan parameter tertentu. Hubungan antara *marketplace* dan Oracle diatur menggunakan interface DynamicPricingOracleClientInterface, yang memungkinkan komunikasi dua arah melalui fungsi *callback* seperti dynamicPricingCallback. Dalam marketplace, terdapat struktur data Product untuk menyimpan informasi setiap produk, termasuk harga, status ketersediaan, dan pemetaan produk yang telah terdaftar.

Selanjutnya, *contract* TicketNFT mengelola tiket dalam bentuk Non-Fungible Token (NFT). Tiket-tiket ini dibuat menggunakan fungsi mintTicket dan disimpan dalam format

yang sesuai dengan standar ERC-721. *Contract* ini mewarisi ERC721URIStorage, sehingga metadata tiket dapat disimpan dan dikelola dengan baik. TicketMarketplace memanfaatkan TicketNFT untuk mengelola proses *minting* dan validasi tiket yang akan dijual di platform.

Hubungan antar komponen dalam sistem ini memastikan bahwa setiap *contract* memiliki tanggung jawab spesifik dan dapat beroperasi secara modular. Oracle berfokus pada penentuan harga, marketplace bertanggung jawab atas pengelolaan produk dan interaksi pengguna, serta TicketNFT mengelola representasi digital tiket dalam bentuk NFT. Dengan desain ini, sistem memiliki fleksibilitas tinggi dan dapat diadaptasi untuk berbagai skenario penggunaan di dunia nyata.

II.2 Oracle

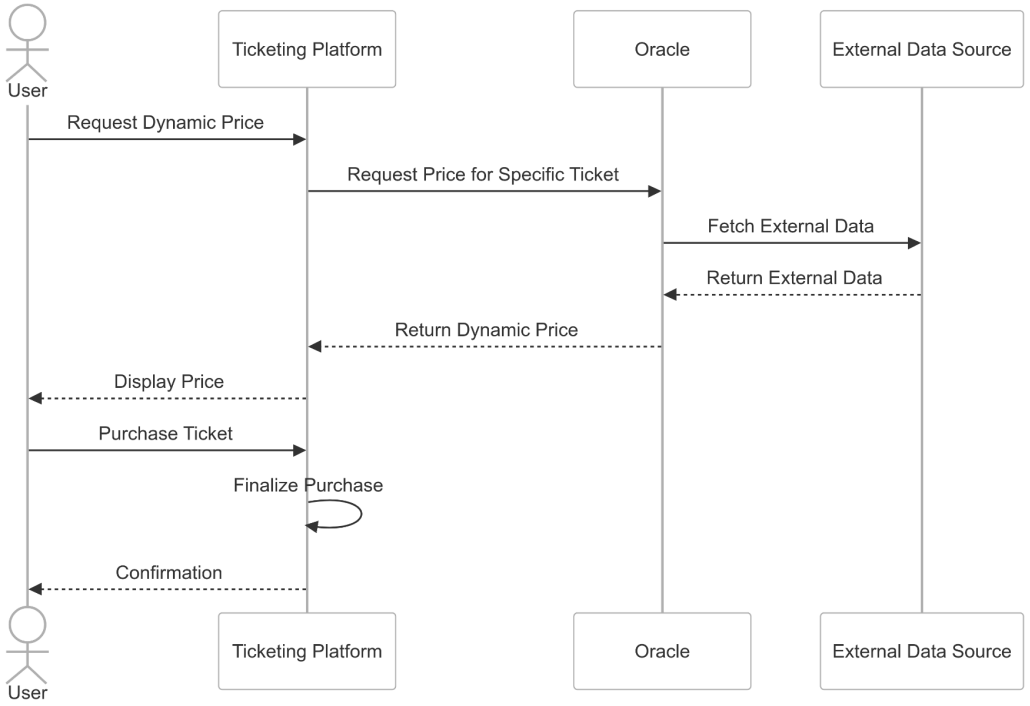
Dalam kasus platform ticketing, Oracle digunakan untuk mengambil harga dinamis tiket berdasarkan faktor-faktor seperti kurs mata uang, harga pasar, atau permintaan saat itu. Struktur data yang diambil dari oracle adalah ticketId dan price yang sudah disesuaikan dan dalam mata uang ether seperti ini

```
{
      "ticket_id": "TICKET001",
      "dynamic_price": 2.0
}
```

Dengan mengimplementasikan seperti itu, smart contract dapat mendapatkan harga secara dinamis berdasarkan kondisi yang terjadi

II.2.1 Mekanisme

Mekanisme komunikasi antara *smart contract* dan Oracle dimulai dengan *smart contract* mengirim permintaan data ke Oracle, lalu Oracle mengirimkan permintaan HTTP/HTTPS ke API eksternal untuk mendapatkan harga terkini. Oracle memvalidasi data yang diterima dari API. Terakhir, Oracle mengirimkan data ke *smart contract* melalui fungsi *callback*. Proses tersebut dapat digambarkan di dalam sequence diagram seperti ini :



II.2.2 Risiko Kegagalan

Terdapat beberapa skenario buruk yang perlu diantisipasi, seperti *downtime* pada API eksternal, manipulasi data, atau keterlambatan respons Oracle. Untuk mengatasi hal ini, beberapa solusi diterapkan, seperti mekanisme *fallback* untuk menggunakan data *price* terakhir jika API gagal merespons atau menggunakan harga tiket asli. Selain itu, implementasi mekanisme *timeout* memastikan bahwa *smart contract* tidak tergantung pada respons Oracle yang lambat. Dengan solusi ini, fitur dynamic pricing menjadi lebih andal dan aman untuk digunakan pada platform ticketing.

II.3 Client

Implementasi pada *client* dibagi menjadi dua komponen, yaitu *frontend* dan *backend.*

II.3.1 Frontend

Aplikasi frontend menggunakan Next.js untuk mendukung berbagai URL berikut:

URL	Deskripsi
/	Menampilkan semua event yang sedang berlangsung (halaman home).
/register	Halaman untuk proses registrasi pengguna.
/profile	Halaman profil pengguna yang menampilkan informasi pribadi dan daftar tiket yang dimiliki.
/events/create	Halaman untuk membuat event baru.

/events/[id]	Halaman detail event yang menampilkan deskripsi lengkap event, daftar seller, dan tempat pembelian tiket.

II.3.2 Backend

Frontend berinteraksi dengan backend melalui REST API yang dirancang untuk berbagai operasi:

Endpoint	Method	Deskripsi
/api/auth/register	POST	Melakukan registrasi pengguna baru
/api/auth/login	POST	Melakukan autentikasi pengguna untuk login.
/api/users/[id]	GET	Mengambil data pengguna berdasarkan ID.
/api/events	GET	Mengambil semua event yang tersedia.
/api/events	POST	Membuat event baru.
/api/events/[id]	GET	Mengambil detail satu event berdasarkan ID.
/api/tickets	GET	Menampilkan semua tiket yang tersedia di marketplace.
/api/tickets	POST	Membuat tiket baru setelah permintaan selesai diproses.
/api/tickets	PATCH	Mengganti kepemilikan tiket berdasarkan transaksi.

II.4 Design Pattern

Terdapat beberapa *design pattern* yang digunakan dalam mengimplementasikan *smart contract*.

II.4.1 Factory Pattern

Factory pattern digunakan untuk membuat *instance* baru dari kontrak atau objek tertentu. Pada platform ini, *factory pattern* digunakan pada fungsi *minting* dan *create* tiket NFT dalam kontrak TicketNFT untuk membuat tiket baru yang direpresentasikan sebagai NFT. Setiap kali fungsi ini dipanggil, tiket baru dicetak dengan detail tertentu dan disimpan dalam penyimpanan.

II.4.2 Oracle Pattern

Oracle pattern digunakan untuk menghubungkan *smart contract* dengan sumber data eksternal. Pada DynamicPricingOracle, *event* dan *callback* (dynamicPricingCallback) digunakan untuk menerima data harga dinamis dari Oracle eksternal.

II.4.3 Observer Pattern

Observer pattern digunakan pada hubungan antara DynamicPricingOracle dan DynamicPricingOracleClientInterface. DynamicPricingOracle bertindak sebagai pengirim notifikasi dengan menyediakan harga dinamis. TicketMarketplace yang mengimplementasikan DynamicPricingOracleClientInterface adalah penerima notifikasi (*observer*). Ketika perubahan harga tersedia melalui fungsi providePrice, DynamicPricingOracle memanggil dynamicPricingCallback untuk memberi tahu observer

III OPTIMASI

Optimasi yang dilakukan adalah menggunakan *modularization* dengan *interface* dalam *smart contract*, seperti yang terdapat pada *file* DynamicPricingOracleInterface.sol dan DynamicPricingOracleClientInterface.sol, memberikan fleksibilitas, efisiensi, dan keterbacaan kode yang lebih baik. Interface di Solidity memungkinkan deklarasi fungsi tanpa implementasi, sehingga memisahkan spesifikasi dan logika implementasi. Hal ini memudahkan pengelolaan kode karena dapat memperbarui atau mengganti logika di kontrak implementasi tanpa mempengaruhi interface selama spesifikasinya tetap sama. Modularisasi ini memastikan standar komunikasi antara kontrak Oracle, seperti DynamicPricingOracle, dengan kliennya, seperti TicketNFT atau TicketMarketplace.

Interface DynamicPricingOracleInterface mendefinisikan fungsi utama seperti getDynamicPrice, yang memberikan harga dinamis berdasarkan faktor-faktor seperti permintaan atau waktu acara. Dengan pendekatan ini, kontrak klien dapat mengakses harga tanpa perlu memahami algoritma yang digunakan di dalam Oracle. Di sisi lain, interface DynamicPricingOracleClientInterface memungkinkan Oracle untuk memberi tahu klien secara otomatis tentang perubahan harga melalui fungsi seperti updatePrice. Hal ini mendukung arsitektur berbasis event-driven yang lebih responsif, di mana harga tiket pada marketplace atau NFT dapat diperbarui secara *real-time*.

IV PEMBAGIAN TUGAS

Berikut merupakan pembagian tugas pada pengerjaan tugas besar

Nama	NIM	Tugas
Frankie Huang	13521092	- Frontend - Backend
Fawwaz Abrial Saffa	18221067	- Frontend - Backend
Abraham Megantoro Samudra	18221123	- Smart Contract - Deployment Smart Contract
Lie, Kevin Sebastian S. T.	18221143	- Smart Contract - Oracle

REFERENSI

Sombat, P., & Ratanaworachan, P. (2023). A blockchain-based ticket sales platform. *2023 27th International Computer Science and Engineering Conference (ICSEC)*. https://doi.org/10.1109/icsec59635.2023.10329682