Decentralized Event Management dApp: ticketbase.base.eth

Overview

Welcome to the **Decentralized Event Management dApp**! This project is built to decentralize event creation, ticketing, and attendance tracking. By leveraging blockchain technology, smart contracts, and DAO governance, this platform ensures a transparent, secure, and scalable ecosystem for event management. The system also aligns with **Sustainable Development Goals (SDGs)** to drive **public good**, **inclusivity**, and **sustainability**.

Key Features

- **Decentralized Event Creation and Management**: Create, manage, and host public/private events with blockchain transparency.
- **Smart Contract-driven Ticketing**: Mint, sell, and transfer tickets via smart contracts as NFTs.
- **Peer-to-Peer Ticket Marketplace**: List and resell tickets securely without intermediaries.
- **Liquidity Pool for Event Funding**: Crowdfund events using decentralized liquidity pools.
- **Decentralized Attendance Verification**: Track attendance through blockchain-based verification systems.
- **Green Event Certifications**: Track and offset event carbon footprints.

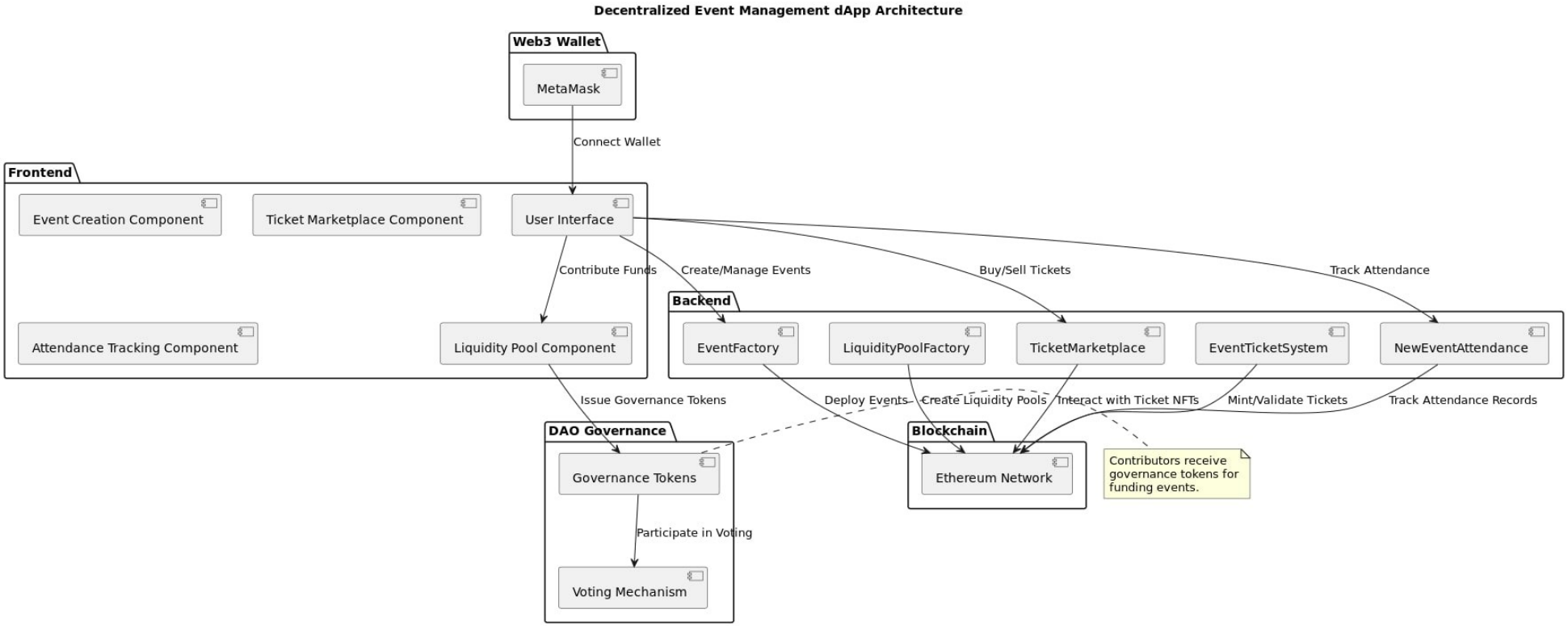
Project Structure

1. Frontend:

The frontend is built using **React.js**, creating a seamless interface for interacting with the blockchain backend.

2. Backend (Smart Contracts):

The backend comprises **five key smart contracts**, each performing a distinct role in event management, ticketing, attendance tracking, and funding.


Decentralized Event Management dApp Architecture
Web3 Wallet
MetaMask
Connect Wallet
Frontend
Event Creation Component
Ticket Marketplace Component
User Interface
Attendance Tracking Component
Liquidity Pool Component
Contribute Funds
Create/Manage Events
Buy/Sell Tickets
Track Attendance
Backend
EventFactory
LiquidityPoolFactory
TicketMarketplace
EventTicketSystem
NewEventAttendance
Issue Governance Tokens
Deploy Events
Create Liquidity Pools
Interact with Ticket NFTs
Mint/Validate Tickets
Track Attendance Records
DAO Governance
Governance Tokens
Participate in Voting
Voting Mechanism
Blockchain
Ethereum Network
Contributors receive governance tokens for funding events.

Installation Guide

Prerequisites

To run the dApp locally, you'll need:

- **Node.js** (v16+)
- **npm** or **yarn**
- **MetaMask** or any Web3 wallet
- **Ganache** or **Hardhat** for blockchain development

1. Clone the Repository

```
git clone https://github.com/your-repo/decentralized-event-management.git
cd decentralized-event-management
```

2. Install Dependencies

```
npm install
# or
yarn install
```

3. Start Local Blockchain

- **Using Ganache** or **Hardhat** to simulate a local Ethereum environment.

4. Compile & Deploy Smart Contracts

```
npx hardhat compile
npx hardhat run scripts/deploy.js --network localhost
```

5. Configure Frontend

Update the smart contract addresses in the frontend configuration (`contractAddresses.json`) with the newly deployed contract addresses.

6. Start the Frontend

```
npm start
# or
yarn start
```

Key Smart Contracts and Features

1. LiquidityPoolFactory

The **LiquidityPoolFactory** contract allows event organizers to create liquidity pools to raise funds for their events. Contributors can invest in events and receive governance tokens in return.

- **createLiquidityPool(eventId, fundingGoal)**: Creates a liquidity pool for a specific event with a funding target.
- **contribute(eventId, amount)**: Allows users to contribute funds to an event's pool.
- **withdrawFunds(eventId)**: After successful funding, organizers can withdraw funds.

Features:

- **Decentralized Crowdfunding**: Participants can invest in events they believe in and earn rewards.
- **DAO-Managed Funding**: The use of DAO ensures that funds are handled transparently, and only approved events receive funding.

---

2. TicketMarketplace

The **TicketMarketplace** is a decentralized platform for buying, selling, and transferring event tickets. All tickets are minted as NFTs (ERC-721/1155), providing a transparent and secure trading environment.

- **listTicket(eventId, ticketId, price)**: List a ticket for sale in the marketplace.
- **buyTicket(eventId, ticketId)**: Allows users to purchase tickets from the marketplace.
- **cancelListing(ticketId)**: Enables sellers to remove a ticket from the marketplace.
- **transferTicket(ticketId, to)**: Transfer ownership of a ticket to another user directly.

Features:

- **Secure P2P Ticket Trading**: Users can buy or sell tickets without intermediaries.
- **NFT-based Ownership**: Tickets are represented as NFTs, ensuring authenticity and verifiable ownership.

---

3. EventFactory

The **EventFactory** contract handles the creation and management of events. It allows users to create decentralized events with features such as ticket pricing, capacity, and more.

- **createEvent(name, date, ticketPrice, maxTickets)**: Allows event organizers to create a new event.
- **updateEvent(eventId, newDate, newPrice)**: Enables organizers to modify event details.
- **closeEvent(eventId)**: Closes the event, preventing further ticket sales.

Features:

- **Decentralized Event Creation**: Anyone can create an event, with all details recorded immutably on the blockchain.
- **Flexibility for Organizers**: Event parameters such as date, price, and capacity can be modified by organizers before the event takes place.

4. EventTicketSystem

The **EventTicketSystem** manages the issuance and validation of event tickets. It connects to the TicketMarketplace and handles the minting of tickets as NFTs.

- **mintTicket(eventId, buyer)**: Mints a new NFT-based ticket for a specific event and assigns it to the buyer.
- **validateTicket(ticketId)**: Validates the authenticity and ownership of a ticket, ensuring it is legitimate for entry to the event.

Features:

- **NFT-based Ticketing**: Tickets are minted as NFTs, providing proof of ownership, preventing fraud, and enabling easy transfers.
- **Integrated with Marketplace**: Tickets can be sold or transferred on the marketplace seamlessly.

---

5. NewEventAttendance

The **NewEventAttendance** contract tracks attendance for events, ensuring that only valid ticket holders are marked as attendees. It uses blockchain immutability to verify participation transparently.

- **markAttendance(ticketId)**: Marks a ticket as attended for a specific event.
- **getAttendance(eventId, ticketId)**: Verifies whether a ticket was used for attendance at an event.

Features:

- **Blockchain-verified Attendance**: Ensures transparent and tamper-proof attendance records.
- **Proof-of-Attendance NFTs**: Attendees can receive NFTs as proof of participation, unlocking potential rewards.

Usage Guide

1. Create an Event

1. **Connect Wallet**: Users connect their MetaMask or Web3 wallet.
2. **Fill Event Details**: Input event details like name, date, price, and capacity.
3. **Submit**: The event is created on-chain via the **EventFactory** contract.

2. Buy Tickets

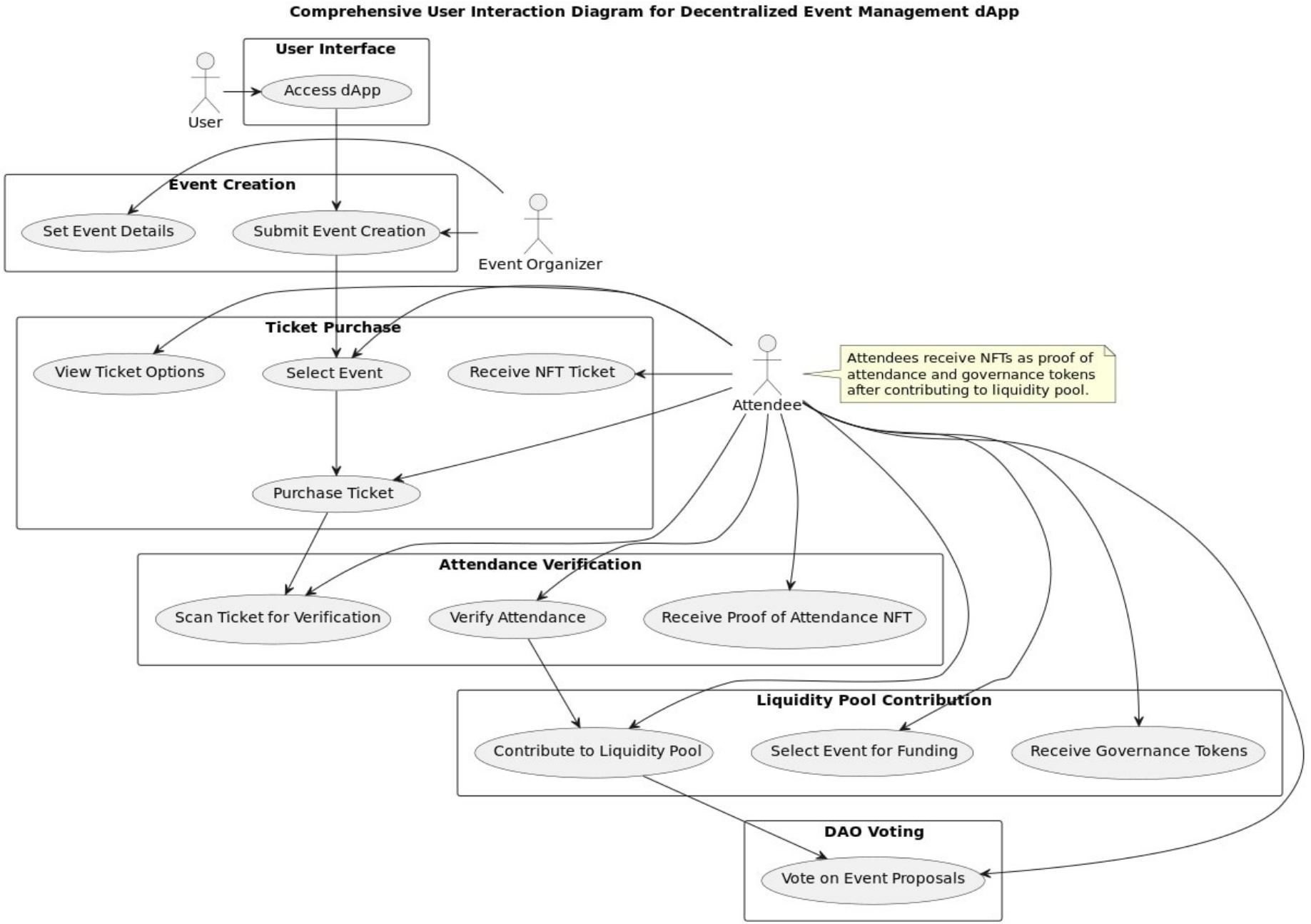
1. **Select Event**: Choose an event from the marketplace.
2. **Purchase Ticket**: Complete the purchase using the **TicketMarketplace** contract. MetaMask will confirm the transaction.
3. **Receive NFT Ticket**: The ticket will be minted and sent to the buyer’s wallet.

3. Track Attendance

1. **At the Event**: Use the **NewEventAttendance** contract to verify and mark ticket holders as attended.
2. **Get Proof of Attendance**: Users may receive a special NFT for attending the event, verified via the smart contract.

4. Contribute to Liquidity Pool

1. **Select Event**: Choose an event to fund through the **LiquidityPoolFactory**.
2. **Contribute**: Send funds to the liquidity pool in return for governance tokens.
3. **DAO Voting**: Use governance tokens to vote on event proposals or other platform changes.


Comprehensive User Interaction Diagram for Decentralized Event Management dApp
User Interface
Access dApp
User
Event Creation
Set Event Details
Submit Event Creation
Event Organizer
Ticket Purchase
View Ticket Options
Select Event
Receive NFT Ticket
Purchase Ticket
Attendee
Attendees receive NFTs as proof of attendance and governance tokens after contributing to liquidity pool.
Attendance Verification
Scan Ticket for Verification
Verify Attendance
Receive Proof of Attendance NFT
Liquidity Pool Contribution
Contribute to Liquidity Pool
Select Event for Funding
Receive Governance Tokens
DAO Voting
Vote on Event Proposals

Future-Proof Features

1. Sustainable Event Management

- Track the **carbon footprint** of events and purchase carbon credits directly via smart contracts.
- Reward attendees who reduce their carbon impact (e.g., using public transportation).

2. Decentralized Crowdfunding

4. Raise funds for events via **liquidity pools**.
5. Contributors are rewarded with governance tokens, allowing them to vote on event-related proposals.

3. NFT-based Proof of Attendance

4. Attendees receive NFTs as proof of participation, which could unlock rewards or access to future events.

4. Inclusive Ticketing

3. Special **subsidized tickets** for marginalized communities using blockchain transparency to ensure fairness.

Security Considerations

- **Smart Contract Audits**: Ensure all contracts undergo a professional audit to mitigate vulnerabilities.
- **Reentrancy Protections**: All contracts are protected against reentrancy attacks.
- **Multi-Signature Wallets**: For critical contract functions, we recommend using multi-signature wallets to ensure secure access.

Contributing

We welcome contributions! Follow these steps:

1. **Fork** the repo.
2. Create a new **branch** (`git checkout -b feature-name`).
3. **Commit** your changes (`git commit -m "add feature"`).
4. **Push** to the **branch** (`git push origin feature-name`).
5. Create a **pull request**.

License

This project is licensed under the **MIT License**.

Thank you for supporting the **Decentralized Event Management dApp**! Feel free to raise any issues or contribute to the project.