Brief Overview of the EHR System

Our decentralized application (dApp) leverages blockchain technology to provide a secure, transparent, and decentralized platform for managing Electronic Health Records (EHRs). The system is designed to ensure the security and privacy of patient data while enabling seamless access and sharing among authorized parties, such as doctors and patients. The core components of the system include user authentication, role-based access control (RBAC), and authorization, all of which are implemented using blockchain's inherent properties and smart contracts.

Implementation of Authentication, Authorization, and Role-Based Access Control (RBAC)

Authentication

Authentication is the process of verifying the identity of users (patients and doctors) who access the system. In our dApp, we implement authentication using the following methods:

1. **Metamask Integration**:
    - Users log in to the dApp using their Metamask wallet, which serves as their digital identity.
    - Metamask provides a secure method for users to sign transactions and authenticate themselves without the need for traditional usernames and passwords.
    - The login process involves connecting the user's Metamask account to the dApp, which verifies the user's identity based on their wallet address.
2. **Smart Contract Deployment**:
    - The smart contract is compiled and deployed on the Ethereum blockchain.
    - The contract address and Application Binary Interface (ABI) are used to interact with the smart contract from the dApp.
    - The contract handles user registration, ensuring that each user is uniquely identified by their wallet address.

Authorization

Authorization determines what actions a user can perform within the system based on their role. Our dApp uses smart contracts to enforce authorization rules:

1. **Role Assignment**:
    - Users are assigned roles (e.g., patient, doctor) during the registration process.
    - The smart contract stores role information and enforces access control based on these roles.
2. **Permission Management**:
    - The smart contract defines specific permissions for each role.

    - For example, a doctor may have permission to access and update patient records, while a patient may only view their own records.
    - Smart contracts ensure that users can only perform actions they are authorized to do based on their role.

Role-Based Access Control (RBAC)

RBAC is implemented to manage access to resources based on the roles assigned to users. The following steps outline the RBAC implementation:

1. **Smart Contracts for Role Management**:
    - Smart contracts are used to define and manage roles and permissions.
    - Roles are stored on the blockchain, and access control logic is enforced by the smart contract.
2. **Access Control Logic**:
    - The smart contract includes functions to check user roles and permissions before allowing access to sensitive operations.
    - For example, a function `getPatientRecord(address patientAddress)` would check if the caller has the doctor role before providing access to a patient's record.
3. **Immutable Audit Trail**:
    - Blockchain's immutable ledger ensures that all transactions and access attempts are recorded transparently.
    - This audit trail provides accountability and helps detect unauthorized access attempts.

Detailed Workflow

1. **User Registration**:
    - Patients and doctors register on the dApp using their Metamask wallet.
    - The smart contract verifies the user's wallet address and stores their role (patient or doctor) on the blockchain.
2. **User Login**:
    - Users log in to the dApp by connecting their Metamask wallet.
    - The dApp retrieves the user's role from the smart contract to customize the user interface and access permissions.
3. **Accessing EHRs**:
    - When a doctor tries to access a patient's EHR, the smart contract checks the doctor's role and permissions.
    - If authorized, the smart contract provides access to the requested EHR.
    - Patients can view their own EHRs but cannot access others' records.
4. **Updating EHRs**:
    - Doctors can update patient records, which are stored as transactions on the blockchain.

    - Each update creates a new transaction, ensuring a complete and immutable history of changes.

My Role in Implementing RBAC

In this project, my primary responsibility was to design and implement the Role-Based Access Control (RBAC) system. This included defining roles, managing permissions, and ensuring secure access to EHRs based on user roles. Here's a detailed overview of my role and contributions:

Responsibilities:

1. **Designing the RBAC Framework**:
    - Defined the roles within the system, such as patients and doctors.
    - Determined the permissions associated with each role, such as viewing and updating EHRs.
2. **Developing Smart Contracts**:
    - Implemented smart contracts to enforce the RBAC rules.
    - Wrote functions to handle role assignment during user registration.
    - Developed functions to check user permissions before allowing access to specific actions, such as accessing or updating health records.
3. **Integration with Metamask**:
    - Integrated Metamask for user authentication, ensuring secure access to the dApp.
    - Ensured that the smart contract interacts correctly with Metamask to verify user identities and roles.
4. **Implementing Access Control Logic**:
    - Developed the logic to manage access based on user roles within the smart contract.
    - Ensured that only authorized users (e.g., doctors) could access or modify EHRs.
    - Implemented functions to grant and revoke access based on patient consent.
5. **Testing and Validation**:
    - Conducted extensive testing to ensure that the RBAC system works as intended.
    - Validated that the smart contracts correctly enforce access control rules and that unauthorized access attempts are appropriately blocked.
6. **Documentation and User Training**:
    - Created detailed documentation explaining the RBAC implementation.
    - Provided training to users (patients and healthcare providers) on how to interact with the system, manage permissions, and understand the access control mechanisms.

Key Contributions:

- **Secure User Authentication**: Ensured that users are authenticated securely using Metamask, leveraging blockchain for identity verification.
- **Role Management**: Developed robust smart contracts to manage user roles and permissions, ensuring that only authorized users can perform specific actions.
- **Data Privacy and Control**: Implemented mechanisms to give patients control over their data, allowing them to manage who can access their EHRs.
- **Seamless Integration**: Worked on integrating the RBAC system with existing healthcare workflows, ensuring minimal disruption and maximum security.

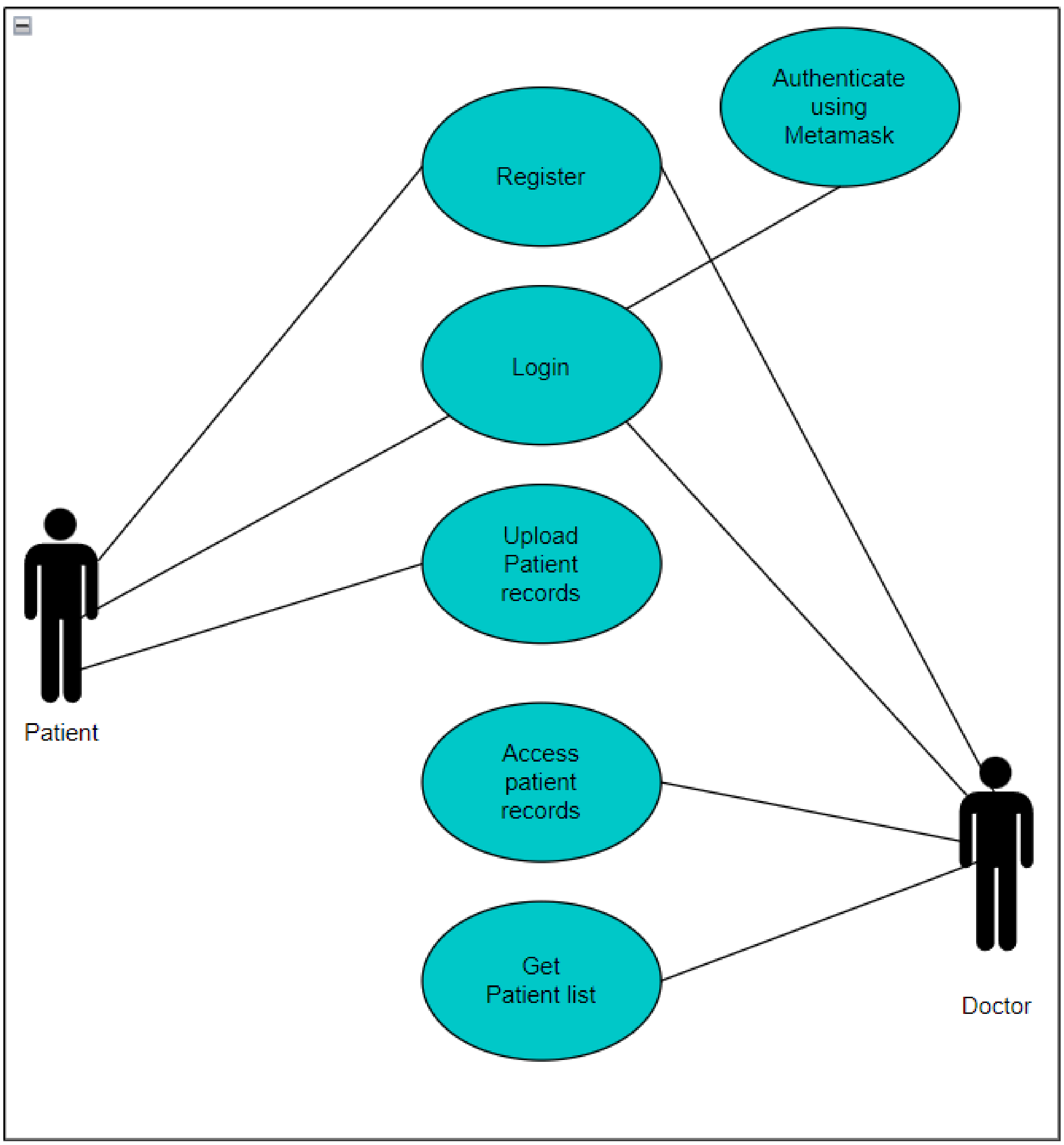


Fig.1: Use case of EHR

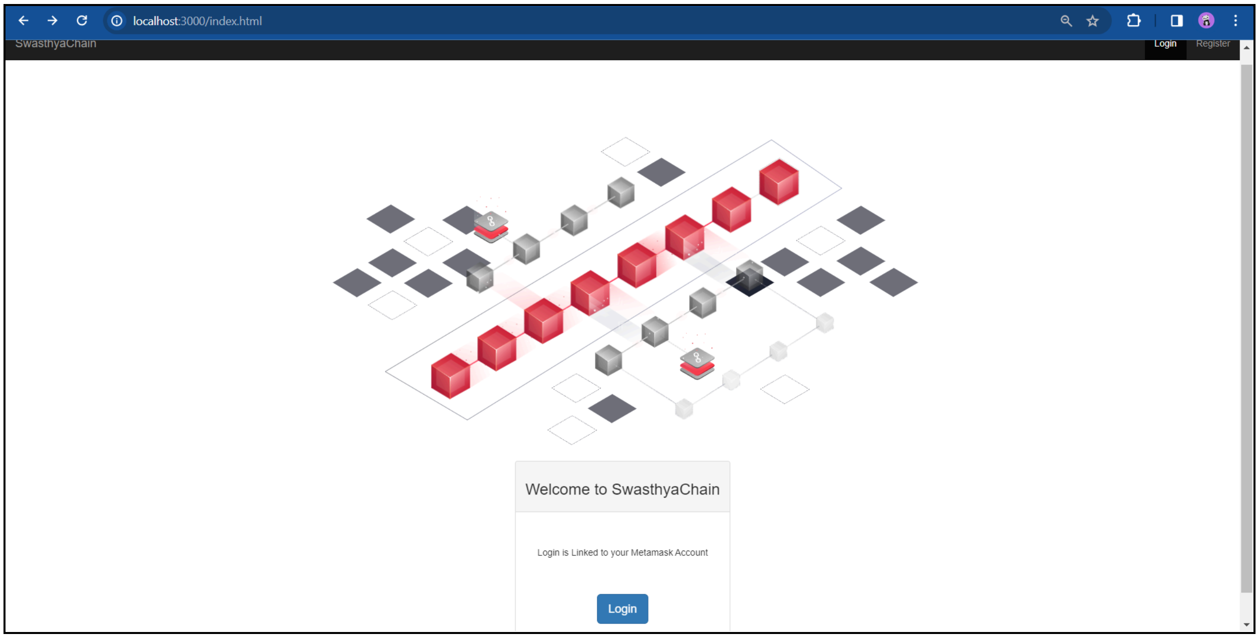


Fig 2: Home Page of the Website

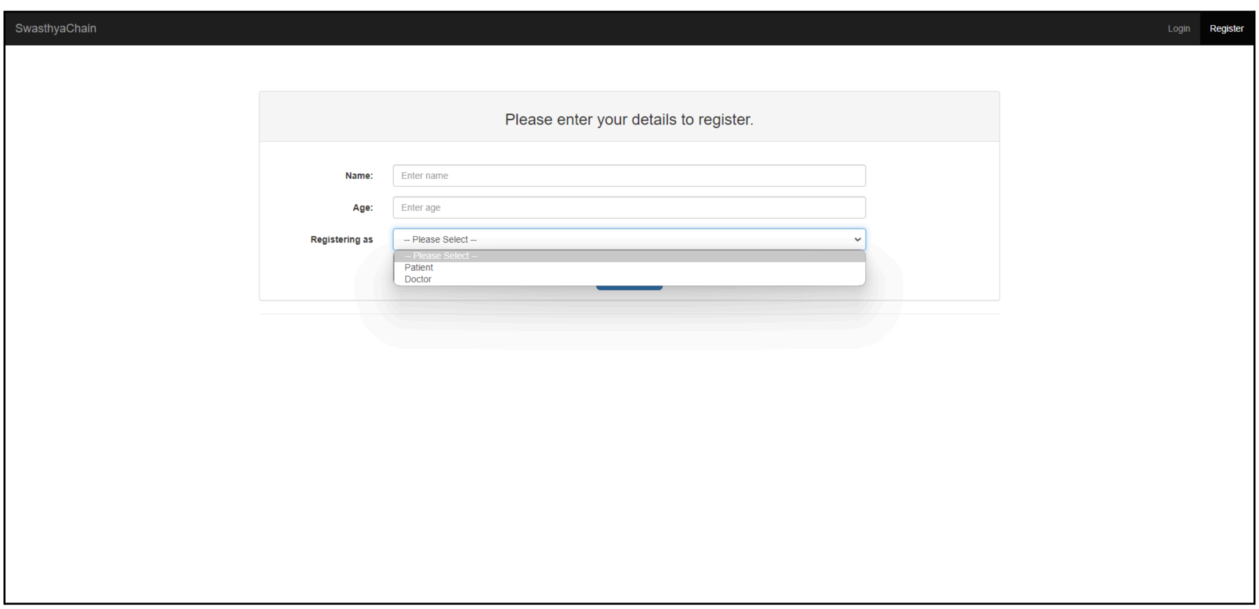


Fig 3: Registration Page of the Website

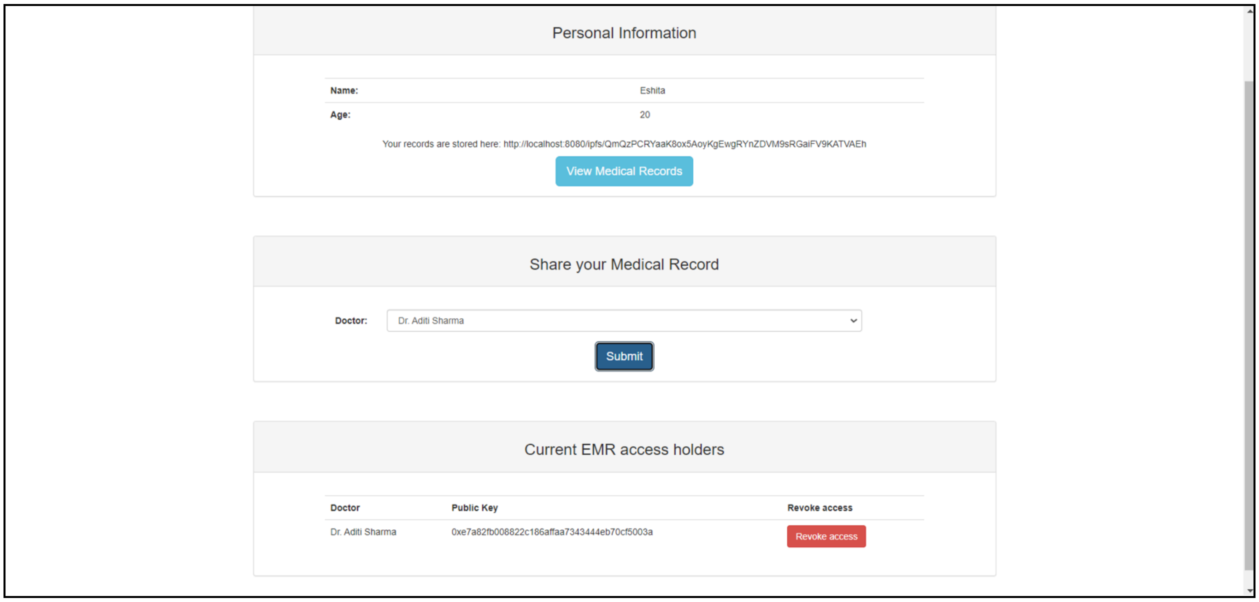


Fig 4: Patient's page of our application

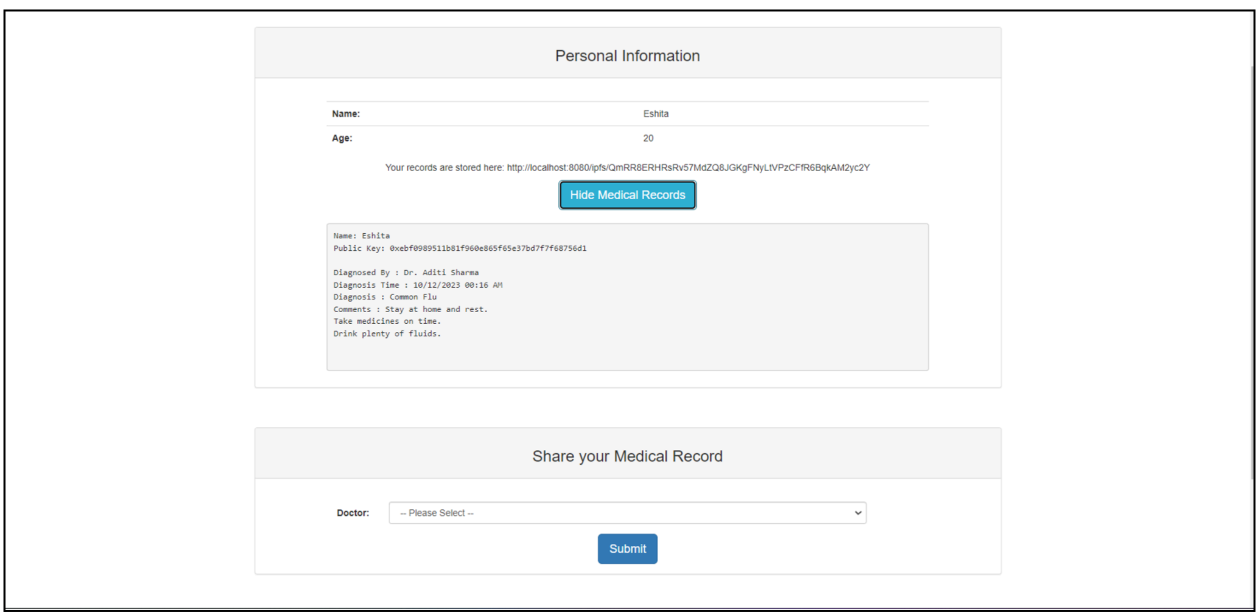


Fig 5: Personal medical record of patient

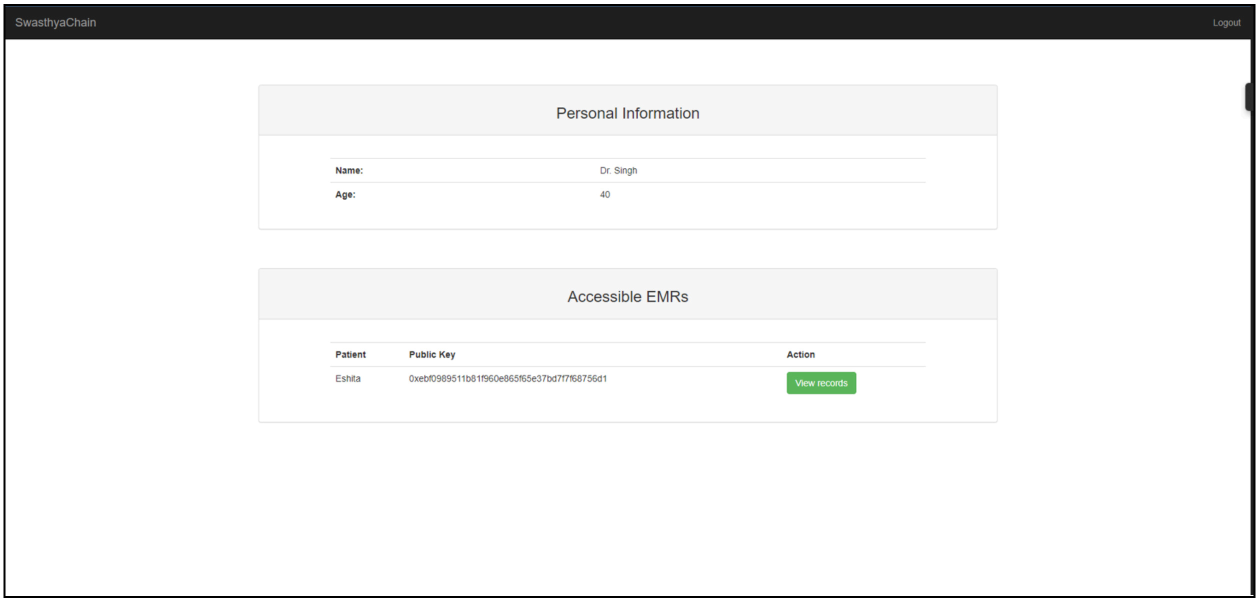


Fig 6: Doctor's page of our application

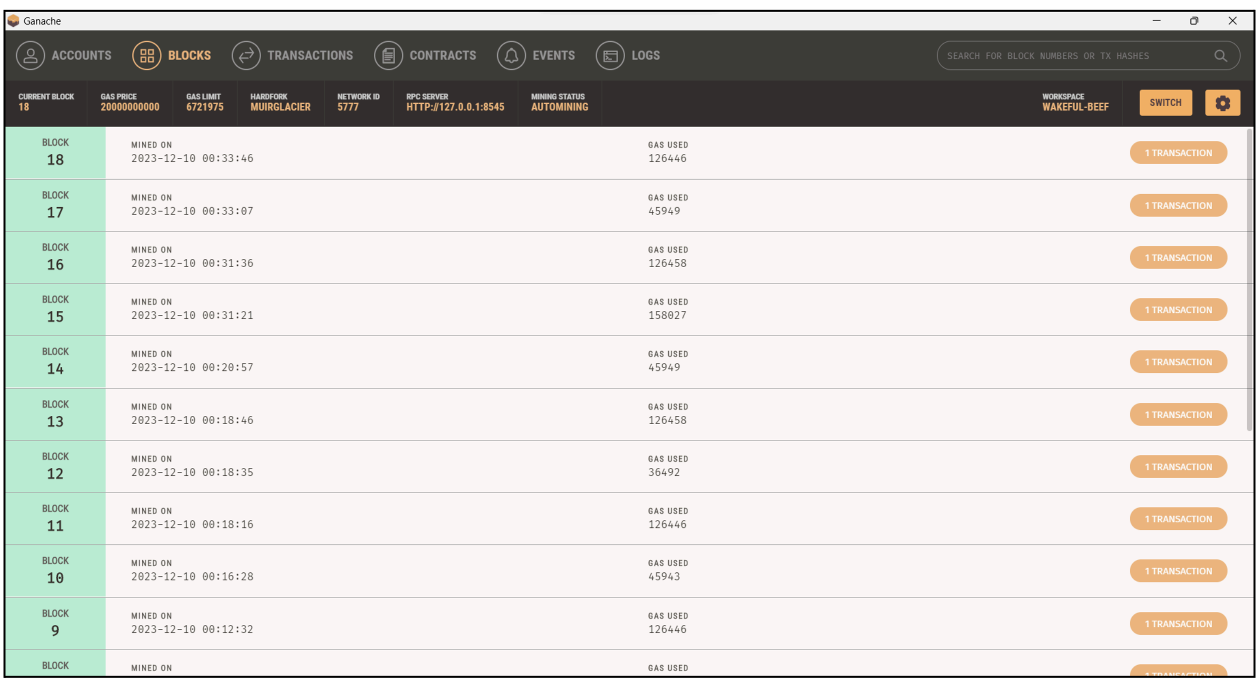
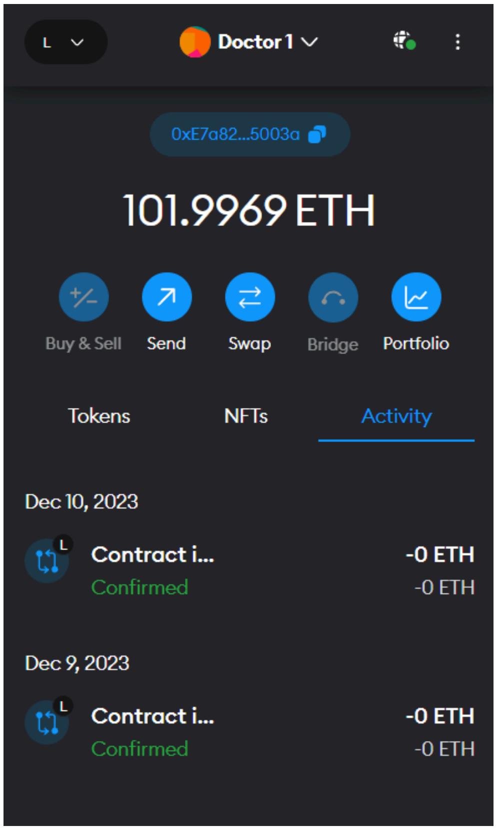


Fig 7: Transaction stored in Ganache



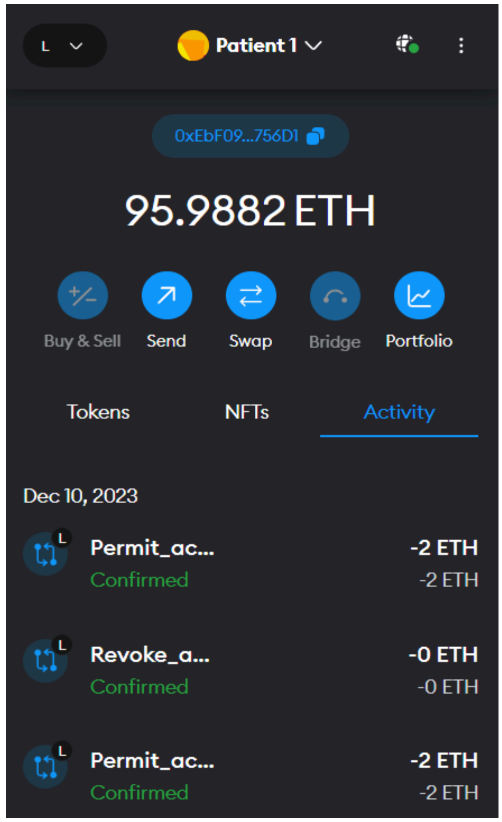


Fig 8: Doctor's metamask account **Fig 9: Patient's metamask account**

Conclusion

Implementing RBAC in the blockchain-based EHR system was crucial to ensuring secure and controlled access to sensitive health data. By designing a comprehensive RBAC framework and developing the necessary smart contracts, I contributed to creating a secure, efficient, and user-friendly EHR management system that leverages the strengths of blockchain technology to address the limitations of traditional EHR systems.