# ASSIGNMENT – 6

## (Blockchain Technology)


**Submitted to: Mr. Shashikant**

**Name: Ishit Singh**

**Class: 3NC1**

**Roll No.: 102115023**

**Semester: Jan’24-May’24**

## Problem Statement :

Create a simple user interface that will allow to interact with Ethereum smart contract using Ethereum TestRPC along with Web3.js.

## MY APPLICATION :

I created an **Token Tracker** that tracks the tokens Sent and Received to and from various accounts. It has interactions with Ethereum smart contract using Ethereum TestRPC and also interacts with a user-friendly front end as well as the MetaMask wallet using Web3.js.

## IMAGES :

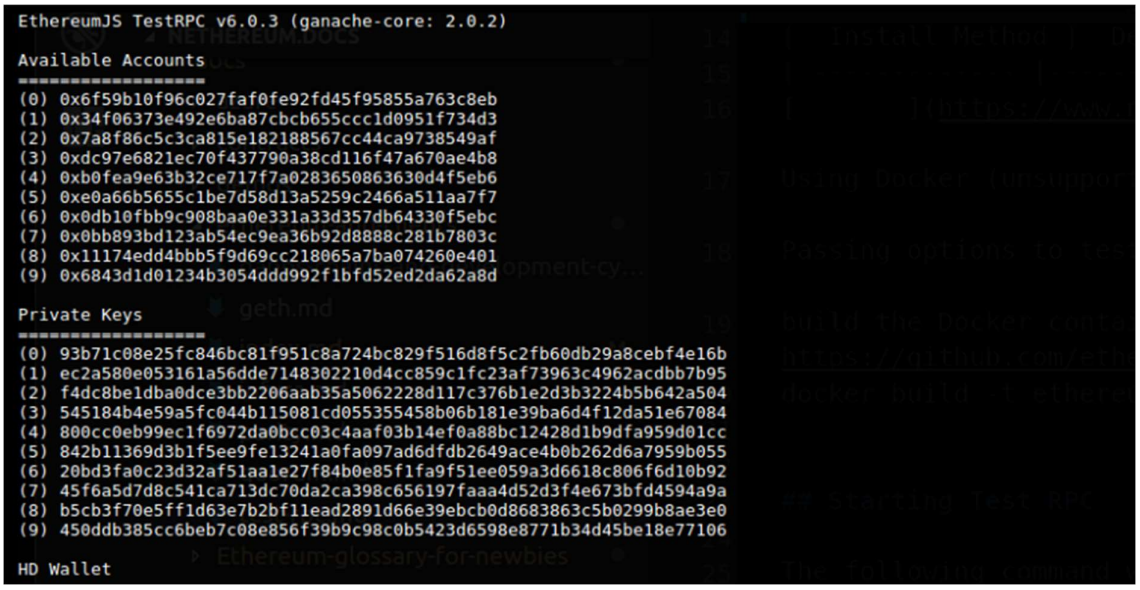


Fig a) Using Ethereum Test RPC to get account IDs and private keys for testing

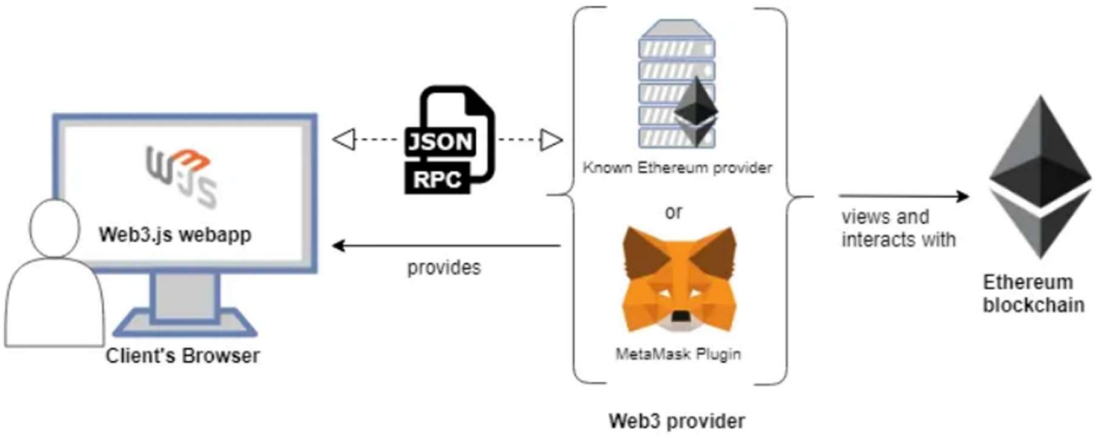


Fig b) Connection of application using Web3.js Library

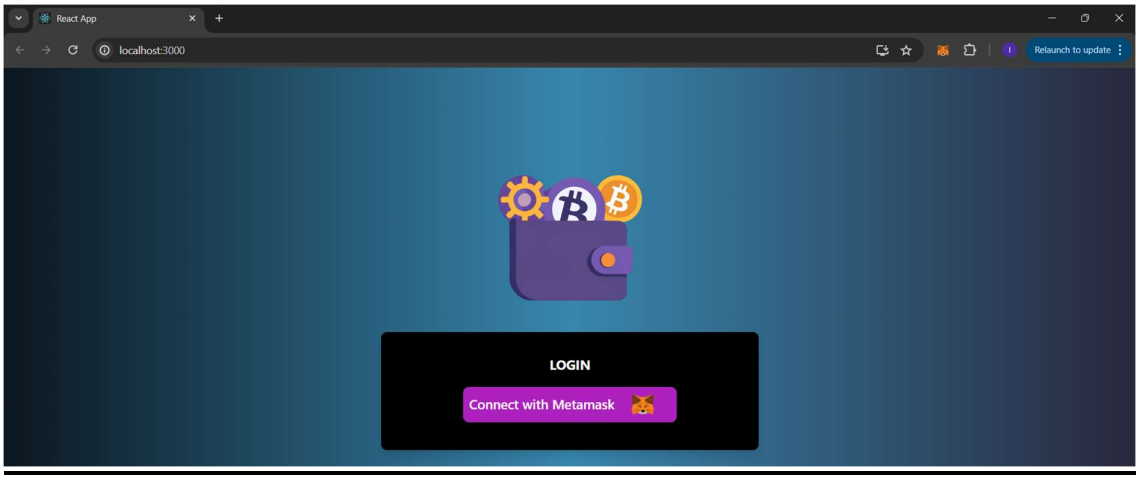


Fig c) Login Page of my “Token Tracker” application

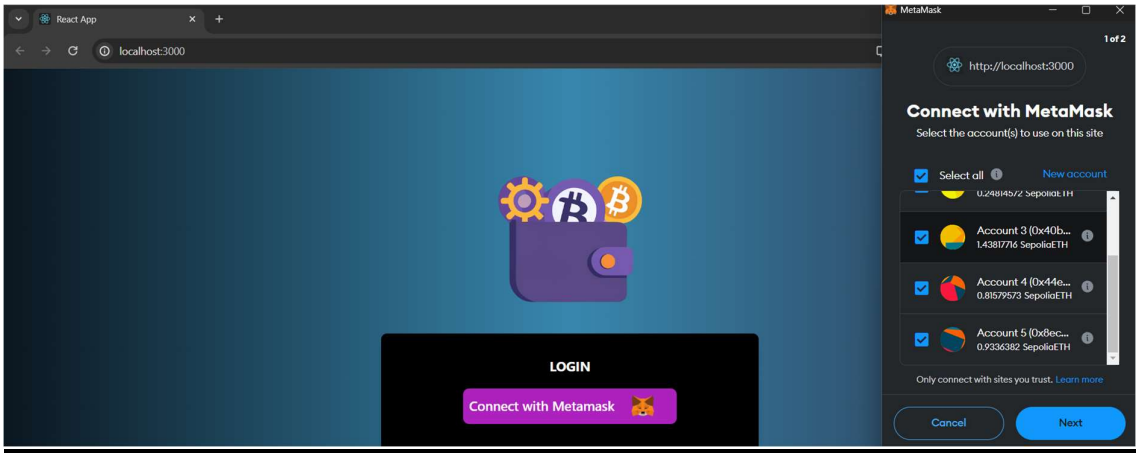


Fig d) Connecting all my MetaMask accounts with the application


Fig e) “Token Tracker” allows you to track tokens in various blockchain networks

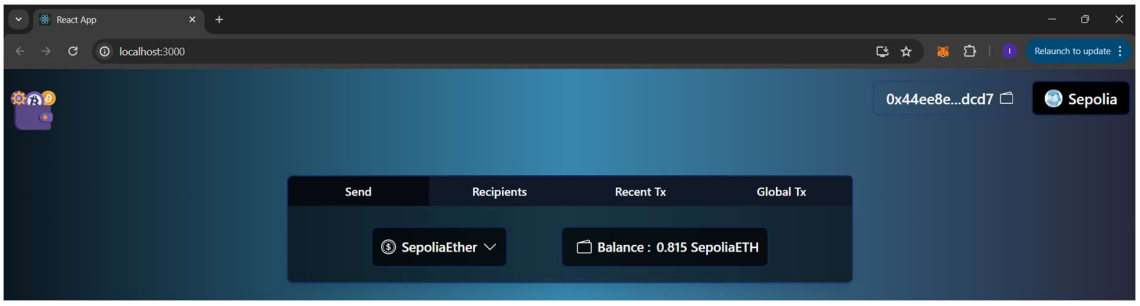


Fig f) Home Page showing the account address, balance and network(here SepoliaEther)

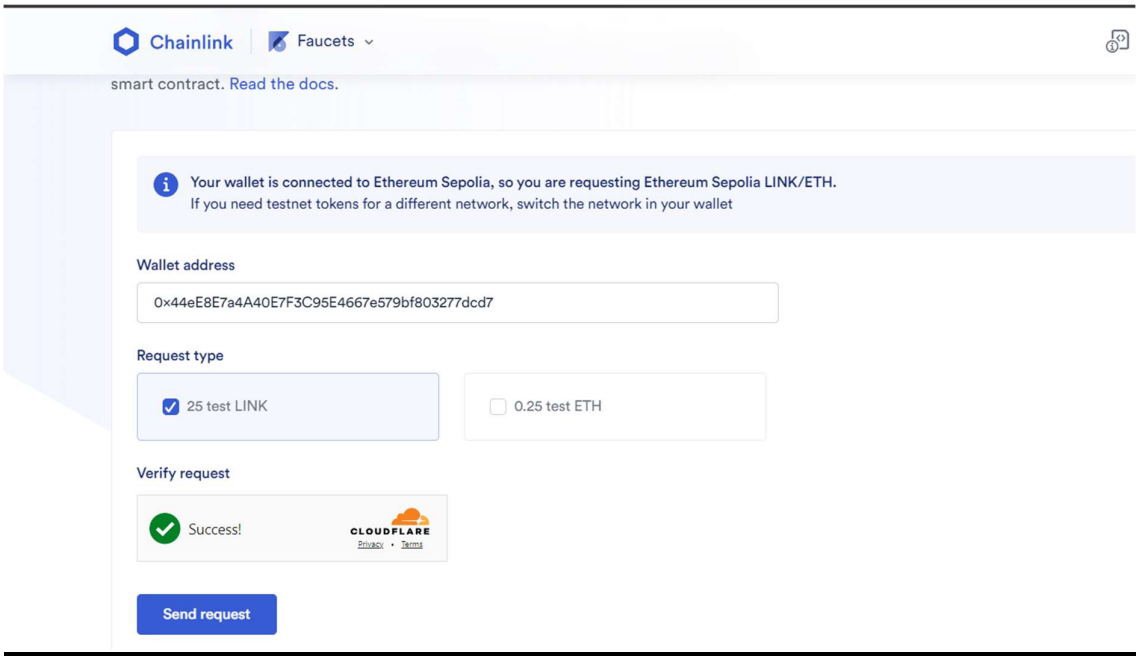


Fig g) Transferring some 25 test LINK from the account to another account through ChainLink

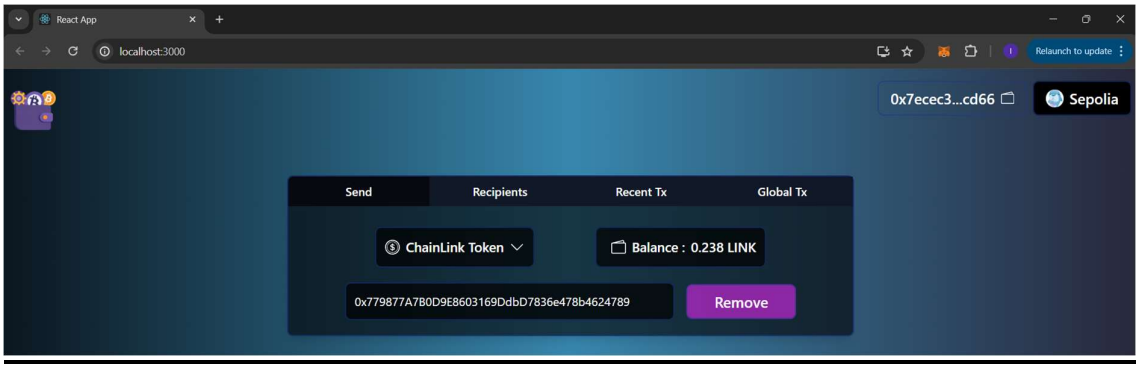


Fig h) Entering the token address to get the type of TOKEN transferred and the balance left

## CODES :

1. Token Tracker contract

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.13;

contract paypal{
    event transactions(address indexed  from, address to, uint amount, string
symbol);
    event recipients(address indexed recipientOf, address recipient, string
recipientName);

    function _transfer(address payable _to, string memory symbol) public
payable{
        _to.transfer(msg.value);
        emit transactions(msg.sender, _to, msg.value, symbol);
    }

    function saveTx(address from, address to, uint amount, string memory
symbol) public{
        emit transactions(from, to, amount, symbol);
    }

    function addRecipient(address recipient, string memory name) public {
        emit recipients(msg.sender, recipient, name);
    }
}
```

---------- Building the Frontend ----------

2. APP.JS

```
import { useState, createContext, useEffect } from "react";
import Main from "./components/Main";
import Header from "./components/Header";
import Login from "./components/Login";
import { ethers } from "ethers";

const AppState = createContext();

function App() {
  const { ethereum } = window;
  const [login, setLogin] = useState(false);
  const [address, setAddress] = useState("");
  const [chain, setChain] = useState("");
  const [symbol, setSymbol] = useState("");
```

```
const [balance, setBalance] = useState("");
const [currency, setCurrency] = useState("");

async function getBal() {
  const provider = new ethers.providers.Web3Provider(ethereum);
  const signer = provider.getSigner();
  const balance = await signer.getBalance();
  setBalance(ethers.utils.formatEther(balance));
}

useEffect(() => {
  ethereum.on("chainChanged", async (chainId) => {
    if (chainId == "0xaa36a7") {
      setChain("Sepolia");
      setSymbol("SepoliaETH");
      setCurrency("SepoliaEther");
    } else if (chainId == "0xe704") {
      setChain("Linea Goerli");
      setSymbol("LineaGoerliETH");
      setCurrency("LineaGoerliEther");
    } else if (chainId == "0xe705") {
      setChain("Linea Sepolia");
      setCurrency("LineaSepoliaEther");
      setSymbol("LineaSepoliaETH");
    } else {
      setLogin(false);
    }
    getBal();
  });

  ethereum.on("accountsChanged", async (accounts) => {
    setAddress(accounts[0]);
    getBal();
  });
}, []);

return (
  <AppState.Provider
    value={{
      login,
      setLogin,
      address,
      setAddress,
      chain,
      setChain,
      symbol,
      setSymbol,
      balance,
```

```
        setBalance,
        currency,
        setCurrency,
        getBal,
      }}
    >
      <div className="min-w-full h-screen">
        {login ? (
          <div className="min-w-full min-h-full">
            {/* Main Application */}
            <Header />
            <Main />
          </div>
        ) : (
          <Login />
        )}
      </div>
    </AppState.Provider>
  );
}

export default App;
export { AppState };
```

3. LOGIN.JS

```
import React, { useState, useContext } from 'react'
import { AppState } from '../App';

const Login = () => {
  const {ethereum} = window;
  const [error,setError] = useState('');
  const App = useContext(AppState);

  const LoginWallet = async() => {
    try{
      // To request Multiple Metamask accounts to connect
      await ethereum.request({
        method: "wallet_requestPermissions",
        params: [{eth_accounts:{}}]
      })
      // To request Single Metamask account to connect
      const accounts = await window.ethereum.request({
                          method: "eth_requestAccounts",
                          params: []
                        });
```

```
      App.setAddress(accounts[0]);

      const chainId = await ethereum.request({
        method: "eth_chainId"
      });

      App.getBal();

      if(chainId == '0xaa36a7'){
        App.setChain("Sepolia");
        App.setLogin(true);
        App.setCurrency('SepoliaEther');
        App.setSymbol('SepoliaETH');
      }
      else if(chainId == '0xe704'){
        App.setChain("Linea Goerli");
        App.setLogin(true);
        App.setCurrency('LineaGoerliEther');
        App.setSymbol('LineaGoerliETH');
      }
      else if(chainId == '0xe705'){
        App.setChain("Linea Sepolia");
        App.setLogin(true);
        App.setCurrency('LineaSepoliaEther');
        App.setSymbol('LineaSepoliaETH');
      }
      else{
        setError("Can only access with Sepolia, Linea Goerli, Linea Sepolia");
        App.setLogin(false);
      }

    }catch(error){
      setError(`"${error.message}"`);
    }
  }

  return (
    <div className='min-w-full h-4/5 flex justify-center flex-col items-
center'>
        <img className='w-1/6 h-21 m-0 p-0' src='walletPay.png' />
        <div className=  ' bg-black w-1/3 h-40 m-0 p-0 bg-opacity-200 p-2
rounded-lg shadow-lg border-opacity-40 border-0 flex flex-col justify-center
items-center'>
          <h1 className='text-white text-lg  font-bold text-center'>LOGIN</h1>
          {
            ethereum !== undefined ?
            // if "ethereum" is NOT undefined ---> DEFINED
```

```
            <div onClick={LoginWallet} className='flex border-opacity-60 bg-
opacity-90 text-lg font-medium border-0 cursor-pointer bg-fuchsia-600 text-
white mt-4 rounded-lg justify-center
              items-center py-1 px-2'>
              Connect with Metamask
              <img className='h-10' src='metamask.png'/>
            </div>
            :
            <div className='flex flex-col justify-center items-center'>
              {/* if "ethereum" is UNDEFINED --> Install Metamask*/}
              <a target={"_blank"}
href='https://chromewebstore.google.com/detail/metamask/nkbihfbeogaeaoehlefnko
dbefgpgknn'>
                <div className='flex border-opacity-60 bg-opacity-90 text-lg
font-medium border-2
                  border-blue-800 cursor-pointer bg-green-800 text-white mt-4
rounded-lg justify-center
                  items-center py-1 px-2'>
                  Install Metamask
                  <img className='h-10' src='metamask.png'/>
                </div>
              </a>
              <p className='text-red-600 text-lg mt-2'>Login Required Metamask
Extension</p>
            </div>
          }
          <p className='text-red-600 text-lg mt-2'>{error}</p>
        </div>
    </div>
  )
}

export default Login
```

4. MAIN.JS

```
import React, { useState } from "react";
import Send from "./Send";
import GlobalTx from "./GlobalTx";
import RecentTx from "./RecentTx";
import Recipients from "./Recipients";

const Main = () => {
  const [route, setRoute] = useState("send");

  return (
    <div className="w-full mt-12 flex flex-col justify-center items-center">
```

```
      {/* BUTTONS - send, recipients, recent tx, global tx */}
      <div className="flex justify-around text-log font-medium items-center
bg-gray-900 border-2 border-b-0 text-white border-opacity-50 border-blue-800
rounded-t-lg w-1/2">
        {/* send */}
        <li
          onClick={() => setRoute("send")}
          className={`list-none cursor-pointer py-2 w-1/4 ${
            route === "send" ? "bg-black bg-opacity-60" : "bg-gray-900"
          } text-center rounded-tl-lg hover:bg-black hover:bg-opacity-60`}
        >
          Send
        </li>
        {/* Recipients */}
        <li
          onClick={() => setRoute("recipients")}
          className={`list-none cursor-pointer py-2 w-1/4 ${
            route === "recipients" ? "bg-black bg-opacity-60" : "bg-gray-900"
          } text-center rounded-tl-lg hover:bg-black hover:bg-opacity-60`}
        >
          Recipients
        </li>
        {/* Recent Tx */}
        <li
          onClick={() => setRoute("recent_tx")}
          className={`list-none cursor-pointer py-2 w-1/4 ${
            route === "recent_tx" ? "bg-black bg-opacity-60" : "bg-gray-900"
          } text-center rounded-tl-lg hover:bg-black hover:bg-opacity-60`}
        >
          Recent Tx
        </li>
        {/* Global Tx */}
        <li
          onClick={() => setRoute("global_tx")}
          className={`list-none cursor-pointer py-2 w-1/4 ${
            route === "global_tx" ? "bg-black bg-opacity-60" : "bg-gray-900"
          } text-center rounded-tl-lg hover:bg-black hover:bg-opacity-60`}
        >
          Global Tx
        </li>
      </div>

      {/* SWITCHING */}
      <div className="bg-black bg-opacity-60 pb-5 overflow-y-auto border-2
border-t-0 shadow-lg border-opacity-50 border-blue-800 rounded-b-lg w-1/2">
        {(() => {
          if (route === "send") {
```

```
            return <Send />;
          } else if (route === "recipients") {
            return <Recipients />;
          } else if (route === "recent_tx") {
            return <RecentTx />;
          } else if (route === "global_tx") {
            return <GlobalTx />;
          }
        })()}
      </div>
    </div>
  );
};

export default Main;
```

5. SEND.JS

```
import React, { useContext, useState } from "react";
import { TailSpin } from "react-loader-spinner";
import { AppState } from "../App";
import { ethers } from "ethers";

const Send = () => {
  const {ethereum} = window;
  const App = useContext(AppState);

  const [showErc, setShowErc] = useState(false);
  const [ercLoading, setErcLoading] = useState(false);
  const [tokenChanged, setTokenChanged] = useState(false);
  const [ercTokenAddress, setErcTokenAddress] = useState("");

  const provider = new ethers.providers.Web3Provider(ethereum);
  const ERCABI = [
    "function balanceOf(address) view returns (uint)",
    "function transfer(address to, uint amount) returns (bool)",
    "function symbol()external view returns (string memory)",
    "function name()external view returns (string memory)"
  ]

  const ERCContract = new ethers.Contract(ercTokenAddress,ERCABI,provider);

  const selectToken = async() => {
    setErcLoading(true);
    const name = await ERCContract.name();
    const balance = await ERCContract.balanceOf(App.address);
```

```
    const symbol = await ERCContract.symbol();

    App.setBalance(ethers.utils.formatEther(balance))
    App.setSymbol(symbol)
    App.setCurrency(name);
    setTokenChanged(true);
    setErcLoading(false);
  }


  return (
    <div className="flex flex-col justify-center items-center text-white">
      {/* Balance */}
      <div
        onClick={() => setShowErc(showErc ? false : true)}
        className="flex w-4/5 justify-around items-center mt-7"
      >
        <div className="flex cursor-pointer justify-center items-center
border-2 border-blue-900 border-opacity-60 p-3 bg-black bg-opacity-70 rounded-
lg">
          <svg
            xmlns="http://www.w3.org/2000/svg"
            width="20"
            height="20"
            fill="currentColor"
            class="bi bi-coin"
            viewBox="0 0 16 16"
          >
            <path d="M5.5 9.511c.076.954.83 1.697 2.182 1.785V12h.6v-.709c1.4-
.098 2.218-.846 2.218-1.932 0-.987-.626-1.496-1.745-1.76l-.473-
.112V5.57c.6.068.982.396 1.074.85h1.052c-.076-.919-.864-1.638-2.126-1.716V4h-
.6v.719c-1.195.117-2.01.836-2.01 1.853 0 .9.606 1.472 1.613
1.707l.397.098v2.034c-.615-.093-1.022-.43-1.114-.9H5.5zm2.177-2.166c-.59-.137-
.91-.416-.91-.836 0-.47.345-.822.915-.925v1.76h-.005zm.692 1.193c.717.166
1.048.435 1.048.91 0 .542-.412.914-1.135.982V8.518l.087.02z" />
            <path d="M8 15A7 7 0 1 1 8 1a7 7 0 0 1 0 14zm0 1A8 8 0 1 0 8 0a8 8
0 0 0 0 16z" />
            <path d="M8 13.5a5.5 5.5 0 1 1 0-11 5.5 5.5 0 0 1 0 11zm0 .5A6 6 0
1 0 8 2a6 6 0 0 0 0 12z" />
          </svg>

          <h1 className="ml-2 text-lg font-medium">{App.currency}</h1>

          <svg
            xmlns="http://www.w3.org/2000/svg"
            width="20"
            height="20"
            fill="currentColor"
            class="ml-2 bi bi-chevron-down"
```

```
            viewBox="0 0 16 16"
          >
            <path
              fill-rule="evenodd"
              d="M1.646 4.646a.5.5 0 0 1 .708 0L8 10.293l5.646-5.647a.5.5 0 0
1 .708.708l-6 6a.5.5 0 0 1-.708 0l-6-6a.5.5 0 0 1 0-.708z"
            />
          </svg>
        </div>
        <div className="flex items-center border-2 border-blue-900 border-
opacity-60 p-3 bg-black rounded-lg bg-opacity-70">
          <svg
            xmlns="http://www.w3.org/2000/svg"
            width="20"
            height="20"
            fill="currentColor"
            class="ml-2 bi bi-wallet2"
            viewBox="0 0 16 16"
          >
            <path d="M12.136.326A1.5 1.5 0 0 1 14 1.78V3h.5A1.5 1.5 0 0 1 16
4.5v9a1.5 1.5 0 0 1-1.5 1.5h-13A1.5 1.5 0 0 1 0 13.5v-9a1.5 1.5 0 0 1 1.432-
1.499L12.136.326zM5.562 3H13V1.78a.5.5 0 0 0-.621-.484L5.562 3zM1.5 4a.5.5 0 0
0-.5.5v9a.5.5 0 0 0 .5.5h13a.5.5 0 0 0 .5-.5v-9a.5.5 0 0 0-.5-.5h-13z" />
          </svg>
          <h1 className="ml-2 text-lg font-medium">Balance :</h1>
          <h1 className="ml-2 text-lg font-medium">
            {App.balance.slice(0, 5)} {App.symbol}
          </h1>
        </div>
      </div>

      {/* ERC20 Address */}
      {showErc ? (
        <div className="flex w-4/5 justify-between items-center mt-5">
          <input
            onChange={(e) => setErcTokenAddress(e.target.value)}
            value={ercTokenAddress}
            className="w-3/4 p-3 bg-black border-2 border-blue-900 border-
opacity-60 bg-opacity-70 outline-none rounded-lg"
            placeholder="Paste ERC20 Token Address"
          />
          {ercLoading ? (
            <div className="flex p-2 cursor-pointer justify-around items-
center w-1/4 ml-4 bg-blue-800 bg-opacity-70 border-2 border-blue-900 border-
opacity-60 text-xl font-medium rounded-lg">
              <TailSpin width={28} height={28} color={"white"} />
            </div>
          ) : tokenChanged ? (
```

```
            <div className="flex cursor-pointer justify-around items-center w-
1/4 p-2 ml-4 bg-fuchsia-600 bg-opacity-70 border-2 border-blue-900 border-
opacity-60 text-xl font-medium rounded-lg">
              Remove
            </div>
          ) : (
            <div onClick={selectToken} className="flex cursor-pointer justify-
around items-center w-1/4 p-2 ml-4 bg-fuchsia-600 bg-opacity-70 border-2
border-blue-900 border-opacity-60 text-xl font-medium rounded-lg">
              Select
            </div>
          )}
        </div>
      ) : (
        <div></div>
      )}
    </div>
  );
};

export default Send;
```

6. INDEX.CSS

```
@tailwind base;
@tailwind components;
@tailwind utilities;

body {
  background-image: linear-gradient(to right, #0b161f, #3887ae, #28283c);
}
```