****

**Department of Computer Science Engineering**

**Amrita School of Engineering**

**Coimbatore- 641 112, Tamil Nadu, India**

**Celebrity Voting dApp**

***A project submitted***

***in partial fulfillment of the requirements for the degree of***

***Masters of Technology in Computer Science Engineering***

**By**

**Sirajuddin Khan**

**(CB.SC.P2CSE23011)**

**Patil Sanket Shashikant**

**(CB.SC.P2CSE23020)**

**Saurav Kumar Topo**

**(CB.SC.P2CSE23028)**

**Patel Heet Sureshbhai**

**(CB.SC.P2CSE23022)**

**Supervised by:**

**Dr. Bagavathi C.**

**24CS721**

**BLOCKCHAIN TECHNOLOGY FOR ENTERPRISE GRADE APPLICATION DEVELOPMENT**

|  |  |  |
| --- | --- | --- |
| **Roll No** | **Name** | **Official Email ID** |
| CB.SC.P2CSE23011 | Sirajuddin Khan | cb.sc.p2cse23011@cb.students.amrita.edu |
| CB.SC.P2CSE23020 | Patil Sanket Shashikant | cb.sc.p2cse23020@cb.students.amrita.edu |
| CB.SC.P2CSE23028 | Saurav Kumar Topo | cb.sc.p2cse23028@cb.students.amrita.edu |
| CB.SC.P2CSE23022 | Patel Heet Sureshbhai | cb.sc.p2cse23022@cb.students.amrita.edu |

**GitHub URL of the project page:**

<https://github.com/The-Block-Dodgers/updated_single_page_webapp.git>

**APPLICATION**

**Celebrity Voting dApp**

**Description:** This project aims to develop a machine learning model to predict heart attack risk based on health attributes and medical history. The dataset includes features like age, gender, cholesterol levels, and more, with a binary target variable indicating heart attack occurrence. The goal is to accurately classify individuals into high or low-risk groups, aiding healthcare professionals in early intervention. Success is measured by the model's accuracy, performance, and insights into heart attack risk factors obtained through dataset analysis.

**Technologies Used:**

|  |  |
| --- | --- |
| Front End | React |
| Back End | Casper Wallet |
| Editor | Visual Studio Code |
| Language | Javascript and Rust |

**React:** React is a popular JavaScript library developed by Facebook for building user interfaces, particularly for single-page applications. It enables developers to create reusable UI components, which can manage their own state and be composed to create complex UIs. React uses a virtual DOM to efficiently update and render components, improving performance. Its declarative nature makes the code more predictable and easier to debug. React is widely used in web development due to its flexibility, performance, and strong community support.

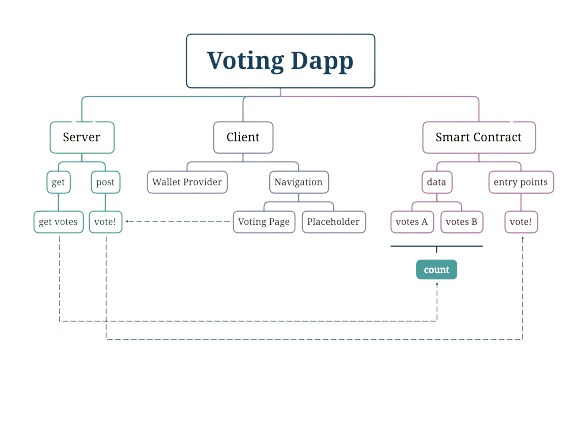
**Casper Wallet:** The Casper Wallet is a digital wallet designed for the Casper Network, a proof-of-stake blockchain that aims to provide enterprise-grade solutions with a focus on security, scalability, and decentralization. The wallet enables users to securely store, manage, and transfer CSPR tokens, which are native to the Casper Network. It also supports features like staking, allowing users to participate in network validation and earn rewards. The Casper Wallet offers a user-friendly interface, robust security measures, and seamless integration with Casper's ecosystem, making it an essential tool for interacting with the Casper blockchain.

**Visual Studio Code:** Visual Studio Code, often abbreviated as VS Code, is a free and open-source code editor developed by Microsoft. It's highly customizable, with support for various programming languages and extensions, making it a favorite among developers across different platforms. VS Code offers features like IntelliSense for smart code completion, debugging capabilities, built-in Git integration, and a powerful extension ecosystem that allows developers to tailor their coding environment to their specific needs. Its lightweight and fast performance, combined with its extensive feature set, have made it one of the most popular code editors in the developer community.

**JavaScript:** JavaScript is a versatile programming language primarily used for web development. It allows developers to add interactivity, dynamic content, and behavior to web pages. Originally created by Netscape, it has evolved into one of the most widely-used languages on the web, supported by all modern web browsers. JavaScript can be used both on the client-side (in the browser) and server-side (with platforms like Node.js), making it a full-stack language. It's dynamically typed and supports object-oriented, imperative, and functional programming paradigms. With its extensive ecosystem of libraries and frameworks, JavaScript is fundamental to modern web development.

**Rust:** Rust is a systems programming language that prioritizes safety, concurrency, and performance. Developed by Mozilla, it aims to provide a reliable alternative to C and C++ by preventing common programming errors like null pointer dereferences and buffer overflows through its strong type system and ownership model. Rust's fearless concurrency model allows developers to write concurrent code without fear of data races, thanks to its ownership and borrowing rules. Its performance is comparable to C and C++, making it suitable for low-level systems programming, as well as high-performance applications like game engines and web servers. Rust also has a growing ecosystem of libraries and tools, supported by an active and passionate community.

**MODEL DIAGRAM**



**CODE**

* **Home Page:**

import { useWallet } from '../service/CasperWallet';

import { Link } from 'react-router-dom';

export default function VotingPage() {

  const { activePublicKey, isLocked, fnConnect } = useWallet();

  return (

    <div>

      <div style={{

          backgroundImage: `url(https://s7d2.scene7.com/is/image/TWCNews/emmys-virtual-7-22-20jpg)`,

          backgroundRepeat: "no-repeat",

          backgroundSize: "cover"

          }}>

        <div className="mx-auto max-w-2xl py-32 sm:py-48 lg:py-56">

            <div className="text-center">

            <h1 style={{ fontFamily: 'PT Sans, sans-serif', color: '#FFFFFF' }} className="text-4xl font-bold tracking-tight text-yellow-200 sm:text-6xl pb-5">

                The BLOCK-DOGERS<br/>Network

              </h1>

              <div className="mb-8 whitespace-normal break-all">

                  <div style={{ fontFamily: 'PT Sans, sans-serif', backgroundColor: 'silver', color: 'black', border: '2px solid blue' }} className="relative rounded-full px-3 py-1 text-sm sm:text-base lg:text-lg leading-6 text-pink-600 ring-1 ring-white-900/10 hover:ring-white">

                    Account: {activePublicKey === undefined ? "Not connected!" : activePublicKey}

                  </div>

              </div>

              <div className="mt-6 text-lg leading-8 text-gray-600">

                {isLocked && (<button onClick={fnConnect} className="rounded-md bg-white px-3.5 py-2.5 text-sm font-semibold text-red-900 shadow-sm hover:bg-gray-100 focus-visible:outline focus-visible:outline-2 focus-visible:outline-offset-2 focus-visible:outline-white">Unlock</button>)}

                <br />

              </div>

              <div className="mt-10 items-center justify-center gap-x-6">

              {isLocked === false && activePublicKey === undefined ? (

                <button

                  onClick={fnConnect}

                  className="rounded-md bg-white px-3.5 py-2.5 text-sm font-semibold text-gray-900 shadow-sm hover:bg-gray-100 focus-visible:outline focus-visible:outline-2 focus-visible:outline-offset-2 focus-visible:outline-white"

                >

                  Connect Wallet

                </button>

              ): (<div></div>)}

               <div className="mt-6 text-lg leading-8 text-gray-600">

                  {isLocked === false && (

                    <Link to="./VotingPage" className="rounded-md bg-white px-3.5 py-2.5 text-sm font-semibold text-red-900 shadow-sm hover:bg-gray-100 focus-visible:outline focus-visible:outline-2 focus-visible:outline-offset-2 focus-visible:outline-white">

                            Click here To Vote

                    </Link>

                  )}

                  <br />

                </div>

                <a href="https://github.com/sauravkumartopo" className="text-sm font-semibold leading-6 text-white">

                  Explore Github <span aria-hidden="true">→</span>

                </a>

              </div>

            </div>

          </div>

        </div>

        {/\* <div className="relative bg-gray-800 px-6 py-16 sm:px-12 sm:py-20 lg:px-16">

          <div className="absolute inset-0 overflow-hidden">

            <img

              src="https://e1.pxfuel.com/desktop-wallpaper/23/263/desktop-wallpaper-minimalist-digital-art-simple-sunset-digital.jpg"

              alt=""

              className="h-full w-full object-cover object-center"

            />

          </div>

        <div aria-hidden="true" className="absolute inset-0 bg-gray-900 bg-opacity-50" />

          <div className="relative mx-auto flex max-w-3xl flex-col items-center text-center">

            <h2 className="text-3xl font-bold tracking-tight text-white sm:text-4xl pb-10">Blockchain Voting</h2>

            <div className="flex justify-center space-x-10">

              <button

                onClick={() => {vote(activePublicKey, provider, "choice\_A")}}

                className="rounded-md bg-white px-3.5 py-2.5 text-sm font-semibold text-gray-900 shadow-sm hover:bg-gray-100 focus-visible:outline focus-visible:outline-2 focus-visible:outline-offset-2 focus-visible:outline-white"

              >

                Choice "A"

              </button>

              <button

                onClick={() => vote(activePublicKey, provider, "choice\_B")}

                className="rounded-md bg-white px-3.5 py-2.5 text-sm font-semibold text-gray-900 shadow-sm hover:bg-gray-100 focus-visible:outline focus-visible:outline-2 focus-visible:outline-offset-2 focus-visible:outline-white"

              >

                Choice "B"

              </button>

              <button

                onClick={() => getVotes(setVotesA, setVotesB)}

                className="rounded-md bg-white px-3.5 py-2.5 text-sm font-semibold text-gray-900 shadow-sm hover:bg-gray-100 focus-visible:outline focus-visible:outline-2 focus-visible:outline-offset-2 focus-visible:outline-white"

              >

                Update

              </button>

            </div>

            <div className='pt-10'>

              <div className="bg-white rounded shadow p-4 w-80">

                {options.map((option) => (

                  <div key={option.id} className="mb-4">

                    <div className="flex items-center justify-between">

                      <span className="text-lg">{option.text}</span>

                      <span className="text-gray-500 text-sm">

                        {((option.votes / totalVotes) \* 100).toFixed(2)}%

                      </span>

                    </div>

                    <div className="h-2 bg-gray-200 rounded-full">

                      <div

                        className="h-full bg-blue-500 rounded-full"

                        style={{ width: `${(option.votes / totalVotes) \* 100}%` }}

                      />

                    </div>

                    <div className="text-gray-500 text-xs mt-1">

                      {option.votes} vote{option.votes !== 1 ? 's' : ''}

                    </div>

                  </div>

                ))}

              </div>

            </div>

          </div>

        </div> \*/}

      </div>

)}

* **Voting Page:**

import { useEffect, useState } from 'react';

import { useWallet } from '../service/CasperWallet';

import { vote, getVotes } from '../service/controller';

export default function VotingPage() {

  const {  provider, activePublicKey } = useWallet();

  const [ votesA, setVotesA] = useState(0);

  const [ votesB, setVotesB] = useState(0);

  const options = [

    {id: 1, text: "Option A", votes : parseInt(votesA, 16)},

    {id: 2, text: "Option B", votes: parseInt(votesB, 16)}

  ];

  const totalVotes = parseInt(votesA, 16) + parseInt(votesB, 16);

  useEffect(() => {

    getVotes(setVotesA, setVotesB);

  }, [])

  return (

        <div className="relative bg-gray px-6 py-16 sm:px-20 sm:py-30 lg:px-16">

          <div className="absolute inset-0 overflow-hidden bg-opacity-50">

            <img

              src="https://t4.ftcdn.net/jpg/06/31/27/93/360\_F\_631279307\_zm3rUBFO2BUNYsPYHdeyxcoveER2Zeo8.jpg"

              alt=""

              className="h-full w-full object-cover object-center"

            />

          </div>

        <div aria-hidden="true" className="absolute inset-0 bg-gray bg-opacity-50" />

          <div className="relative mx-auto flex max-w-3xl flex-col items-center text-center">

            <h2 className="text-3xl font-bold tracking-tight text-white sm:text-4xl pb-10">Casper Voting </h2>

            <h2 className="text-3xl font-bold tracking-tight text-white sm:text-4xl pb-10">Vote for your favourite Actor</h2>

            <div className="container mx-auto">

            <div className="flex justify-center space-x-10">

              <img

                src='https://www.livemint.com/lm-img/img/2023/11/02/1600x900/TOPSHOT---Bollywood-actor-Shah-Rukh-Khan-greets-fa\_1698890202689\_1698890213014.jpg'

                alt='Shahrukh Khan'

                className="w-80 h-150"

              />

              <img

                src='https://m.media-amazon.com/images/M/MV5BZDE4ZGM2YzEtZDYwNC00OGZhLTk3ZjItN2JiNmFjMzhhNTA0XkEyXkFqcGdeQXVyNjkwOTg4MTA@.\_V1\_.jpg'

                alt='Amitabh Bachchan'

                className="w-80 h-150"

              />

            </div>

          </div>

            <div>

              <button

                onClick={() => {vote(activePublicKey, provider, "choice\_A")}}

                className="rounded-md bg-white px-3.5 py-2.5 text-sm font-semibold text-gray-900 shadow-sm hover:bg-gray-100 focus-visible:outline focus-visible:outline-2 focus-visible:outline-offset-2 focus-visible:outline-white mx-20 mt-4"

              >

                ShahRukh Khan

              </button>

              <button

                onClick={() => vote(activePublicKey, provider, "choice\_B")}

                className="rounded-md bg-white px-3.5 py-2.5 text-sm font-semibold text-gray-900 shadow-sm hover:bg-gray-100 focus-visible:outline focus-visible:outline-2 focus-visible:outline-offset-2 focus-visible:outline-white mx-20 mt-4"

              >

                Amitabh Bachchan

              </button>

            </div>

           <br/>

            <div className="flex justify-center space-x-10 w-5">

            <button

                onClick={() => getVotes(setVotesA, setVotesB)}

                className="rounded-md bg-white px-3.5 py-2.5 text-sm font-semibold text-gray-900 shadow-sm hover:bg-gray-100 focus-visible:outline focus-visible:outline-2 focus-visible:outline-offset-2 focus-visible:outline-white mt-10"

              >

                Update

              </button>

              </div>

            <div className='pt-10'>

              <div className="bg-white rounded shadow p-4 w-80">

                {options.map((option) => (

                  <div key={option.id} className="mb-4">

                    <div className="flex items-center justify-between">

                      <span className="text-lg">{option.text}</span>

                      <span className="text-gray-500 text-sm">

                        {((option.votes / totalVotes) \* 100).toFixed(2)}%

                      </span>

                    </div>

                    <div className="h-2 bg-gray-200 rounded-full">

                      <div

                        className="h-full bg-blue-500 rounded-full"

                        style={{ width: `${(option.votes / totalVotes) \* 100}%` }}

                      />

                    </div>

                    <div className="text-gray-500 text-xs mt-1">

                      {option.votes} vote{option.votes !== 1 ? 's' : ''}

                    </div>

                  </div>

                ))}

              </div>

            </div>

          </div>

        </div>

)}

* **main.rs:**

#![no\_std]

#![no\_main]

#[cfg(not(target\_arch = "wasm32"))]

compile\_error!("target arch should be wasm32: compile with '--target wasm32-unknown-unknown'");

extern crate alloc;

use alloc::{

    string::{String, ToString},

    vec,

    vec::Vec,

};

use casper\_contract::{

    contract\_api::{runtime, storage},

    unwrap\_or\_revert::UnwrapOrRevert,

};

use casper\_types::{contracts::NamedKeys, ApiError, Key, URef, EntryPoints, EntryPoint, Parameter, RuntimeArgs, EntryPointAccess, EntryPointType, U256, CLType};

const VOTES\_DICT\_NAME: &str = "votes";

const CHOICE\_A: &str = "choice\_A";

const CHOICE\_B: &str = "choice\_B";

#[no\_mangle]

pub extern "C" fn vote(){

    let choice: String = runtime::get\_named\_arg("choice");

    let A\_uref: URef = match runtime::get\_key(CHOICE\_A){

        Some(key) => key,

        None => runtime::revert(ApiError::MissingKey)

    }.into\_uref().unwrap\_or\_revert();

    let B\_uref: URef = match runtime::get\_key(CHOICE\_B){

        Some(key) => key,

        None => runtime::revert(ApiError::MissingKey)

    }.into\_uref().unwrap\_or\_revert();

    if choice == CHOICE\_A{

        let mut A\_votes: U256 = storage::read\_or\_revert(A\_uref);

        A\_votes += U256::from(1);

        storage::write(A\_uref, A\_votes);

    }

    else if choice == CHOICE\_B{

        let mut B\_votes: U256 = storage::read\_or\_revert(B\_uref);

        B\_votes += U256::from(1);

        storage::write(B\_uref, B\_votes)

    }

    else{

        runtime::revert(ApiError::Unhandled)

    }

}

#[no\_mangle]

pub extern "C" fn call() {

    let mut entry\_points = EntryPoints::new();

    let vote = EntryPoint::new(

        "vote",

        vec![Parameter::new("choice", CLType::String)],

        CLType::Unit,

        EntryPointAccess::Public,

        EntryPointType::Contract

    );

    entry\_points.add\_entry\_point(vote);

    let named\_keys = {

        let mut named\_keys = NamedKeys::new();

        // For now - 2 simple options

        let A\_uref = storage::new\_uref(CHOICE\_A);

        let B\_uref = storage::new\_uref(CHOICE\_B);

        named\_keys.insert(CHOICE\_A.to\_string(), A\_uref.into());

        named\_keys.insert(CHOICE\_B.to\_string(), B\_uref.into());

        storage::write(A\_uref, U256::from(0));

        storage::write(B\_uref, U256::from(0));

        named\_keys

    };

    storage::new\_contract(

        entry\_points,

        Some(named\_keys),

        Some("Voting contract".to\_string()),

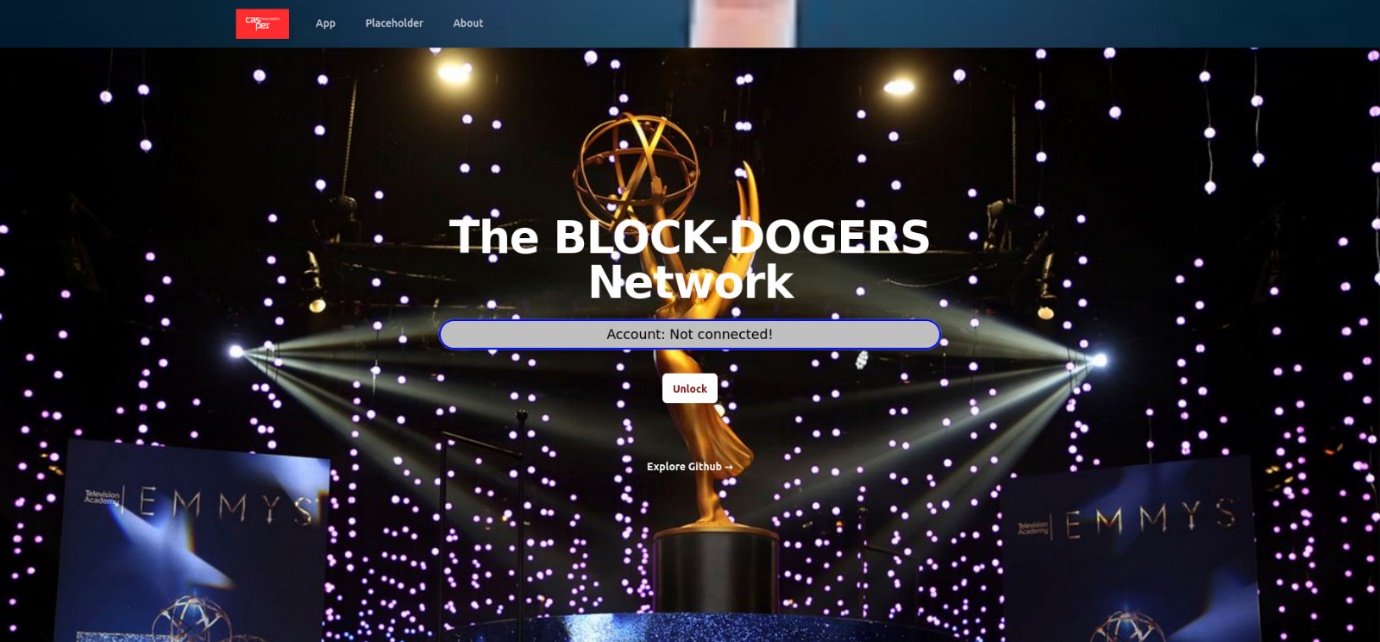
        Some("Voting access key".to\_string())

    );

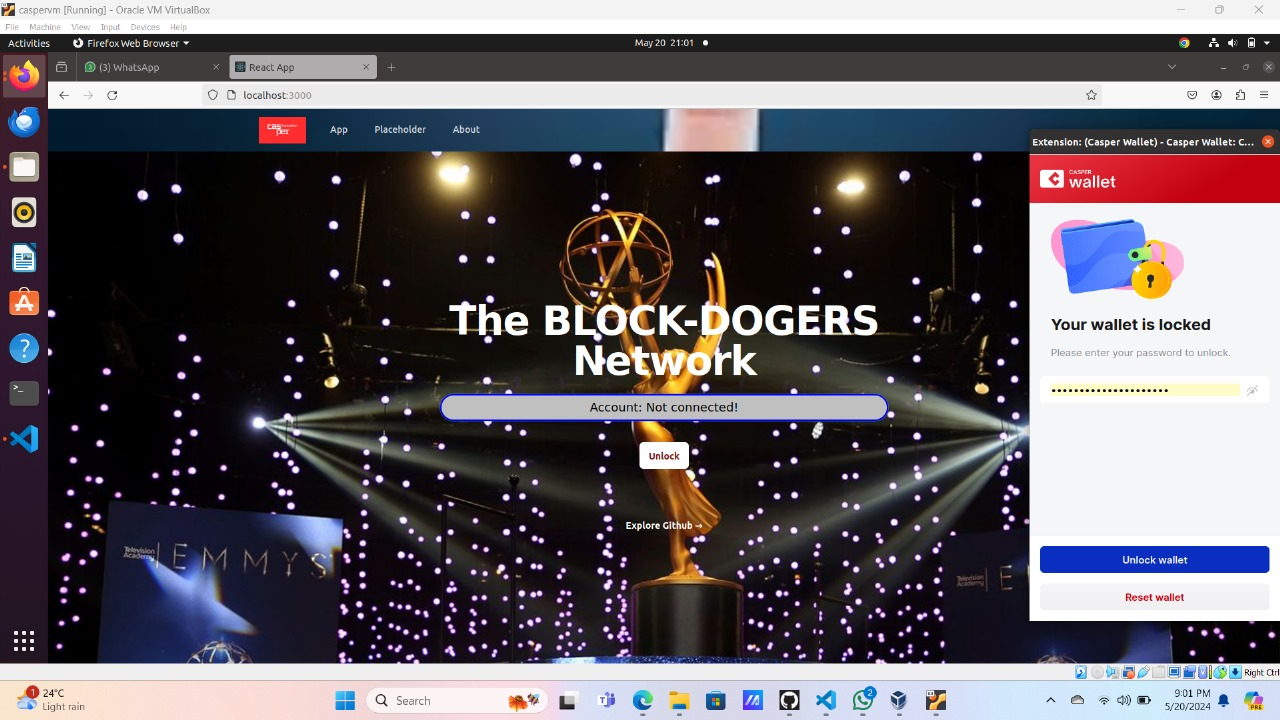
}

**OUTPUT SCREENSHOTS**

* **Home Page:**

****

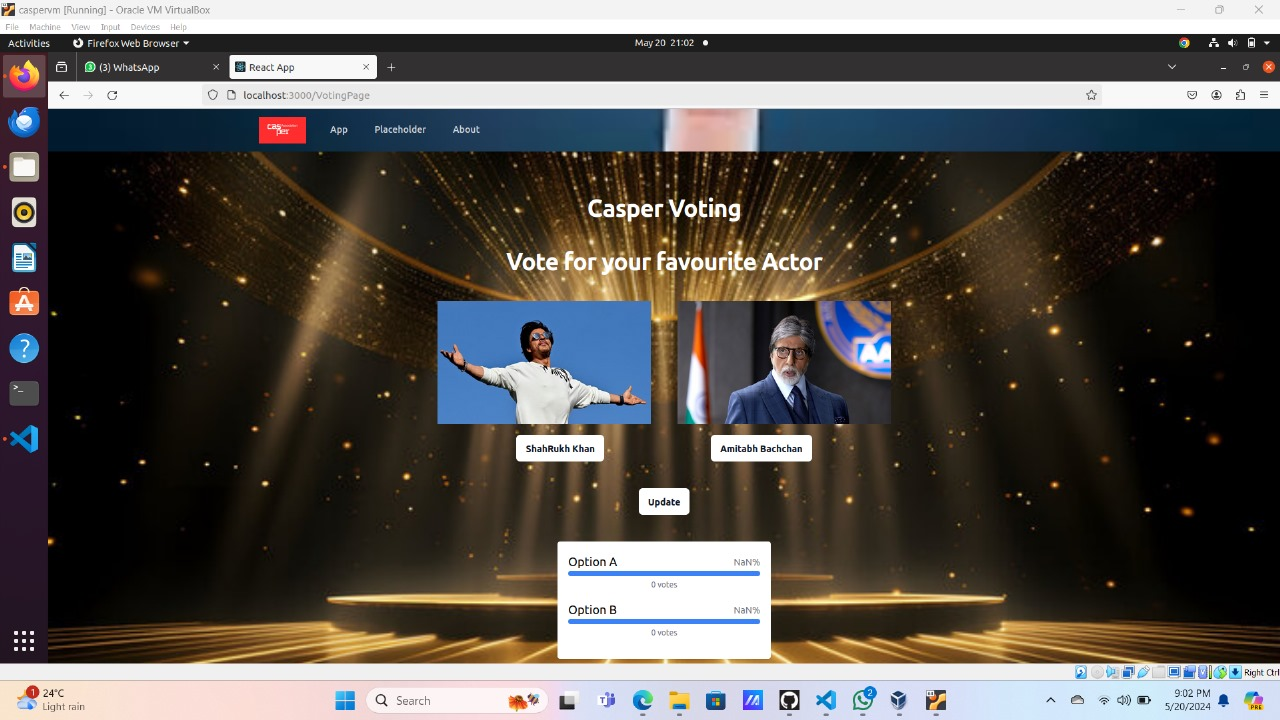
* **Wallet Account Login:**

****

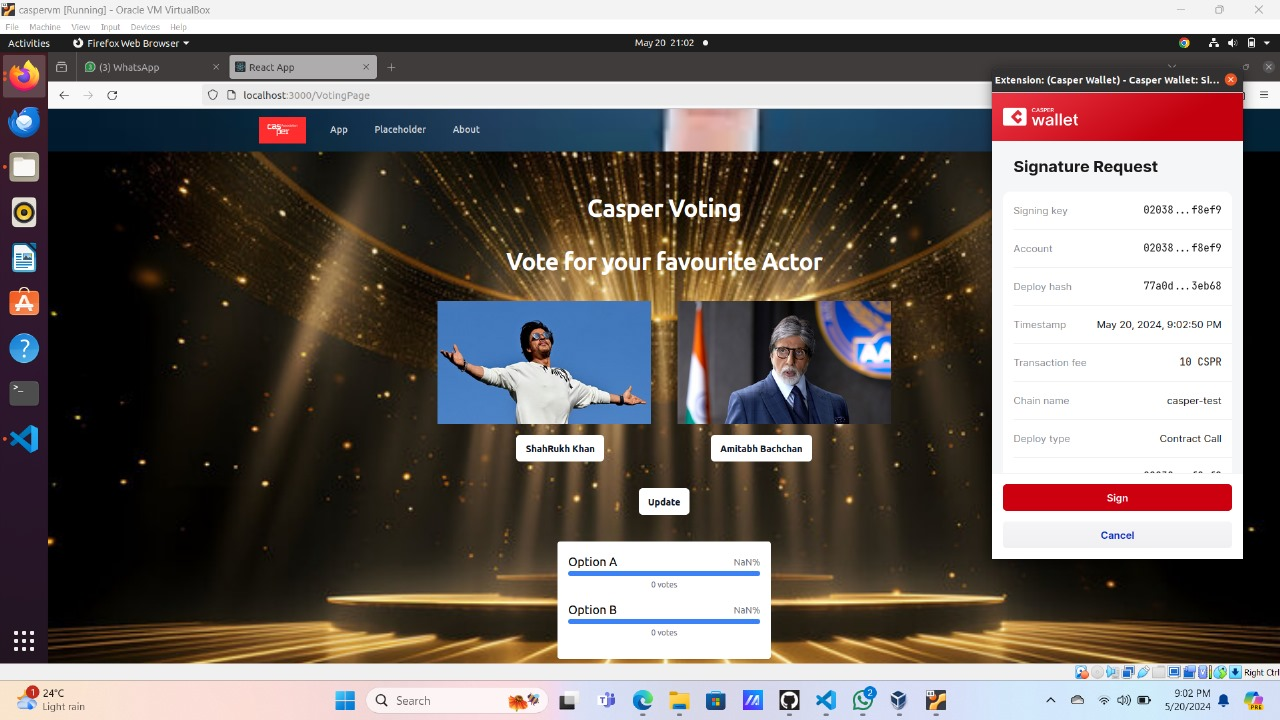
* **Home Page after Login:**

****

* **Voting Page:**

****

* **Hash Generation Page:**

****