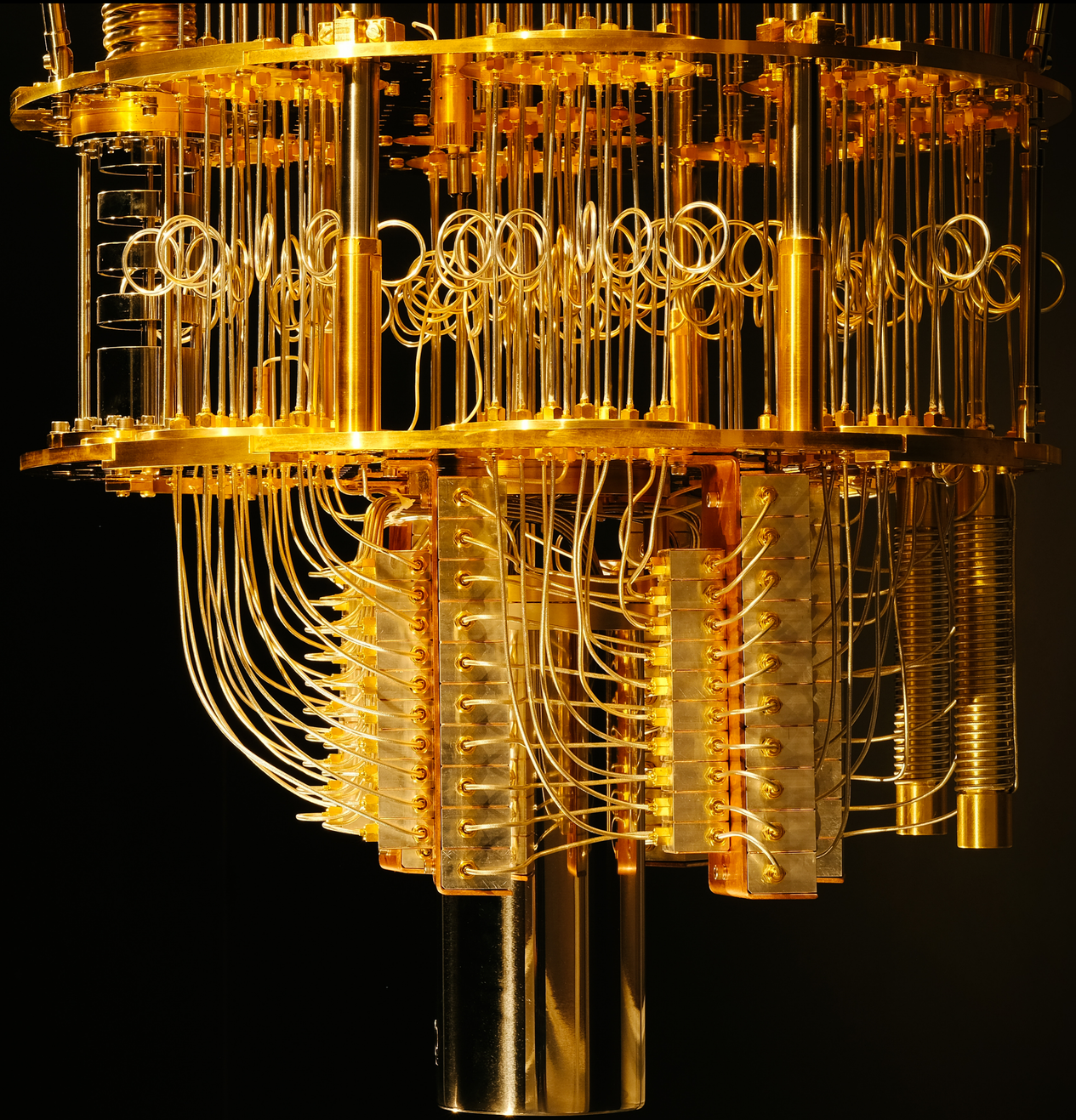
Quantum Oracles

Quantum computing as a service on-chain



Yash Goyal, Jan Ole Ernst, Jessica Pointing

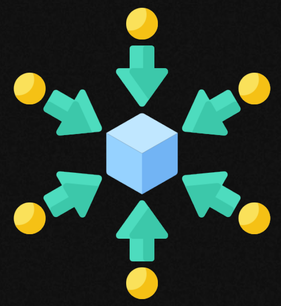
Why?

Quantum computing could solve the world’s hardest problems

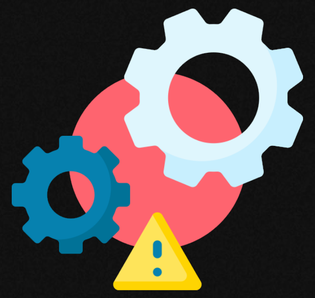
Quantum computing hardware & software development is accelerating

Major companies and governments are getting involved in quantum computing

Problem



Access is centralised and not democratised

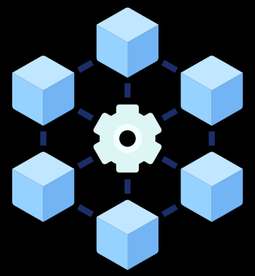


The same computations are rerun from scratch, which is inefficient

No ability to execute logic based on others runs

Our Solution

Quantum Oracles is decentralised quantum computing as-a-service on-chain. We allow users to execute quantum programs, called quantum circuits, on-chain. Therefore, we can change the industry by:



Decentralisation

We decentralise the execution of the program by running the same program on multiple quantum devices, reducing the need for trust and dependency on a single quantum service provider.

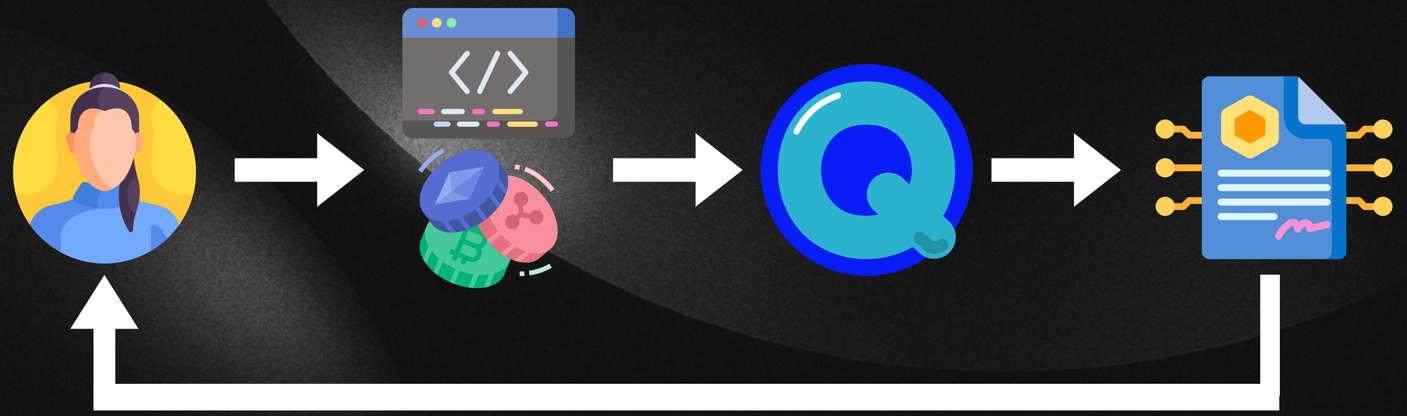
Storing results on-chain

This allows for efficient retrieval of results, so the program only needs to be executed once. Also these results can be used by other smart contracts to execute logic based on the results.

Seamless payments & user experience

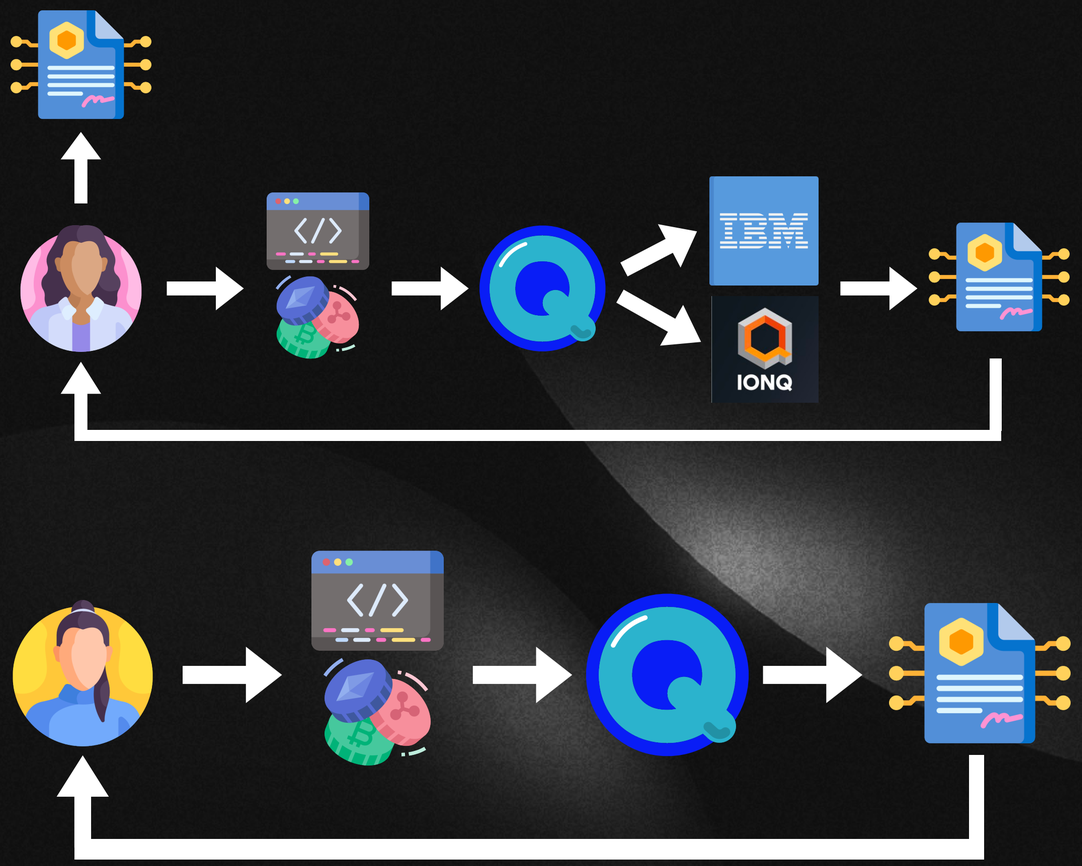
Users can pay in cryptocurrency and don’t need to go through the hassle of setting up accounts and payments on multiple quantum service providers.

Our Solution



If another user submits the same quantum program as someone else, they can receive the results immediately and we can bypass re-running the same computation on the quantum service providers

Benefits



1. **Hashing the circuit and results on-chain**
   Efficient retrieval of results and no duplicate computation; Privacy preserving

2. **Our management of IBM account**
   No need for the user to spend time and energy setting up IBM account and agreement to run on their quantum computer

3. **Storing results on chain**
   User can execute logic in another smart contract based on the quantum computation results

4. **Payment in crypto**
   Seamless payments

5. **Flexible backend options**
   User can choose any backend to run on (simulator, hardware, IBM, Google, etc…).
   Solution is portable to other specialised devices (e.g. GPUs, classical supercomputers…)

6. **Decentralisation**
   The same quantum program is sent to multiple quantum computers/simulators and the results are aggregated and stored on-chain

Tech Stack

flare

Fast and cheap blockchain, where we deployed our smart contract, for storing quantum computation results.

Scaffold-ETH

Used for building the front-end and integrating the front-end with the smart contract.

Google Cloud

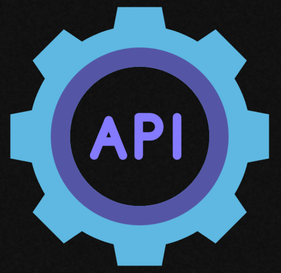
Deployed our quantum API and API3 Airnode on Google Cloud.

API3

Deployed an Airnode to call our quantum API and used the dAPI price feed for calculating the cost in cryptocurrency for running the quantum computation.

Components

User Interface:
- User uploads quantum program
- Price calculation
- Sign transaction
- Display results

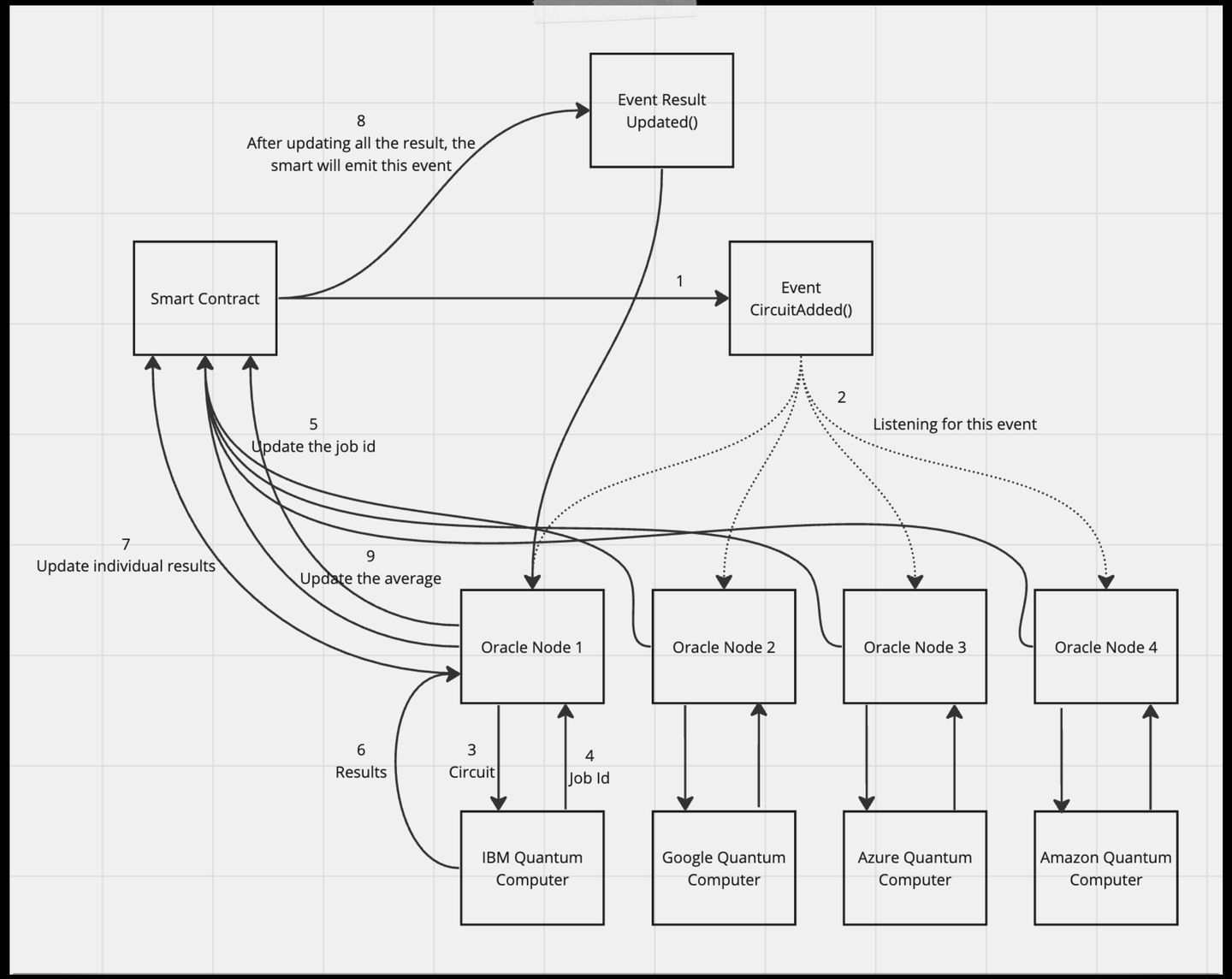


Quantum computing API
- Send a job to a quantum computer (can choose backend)
- Retrieve job result
- Draw diagram of quantum program
- Deployed at URL

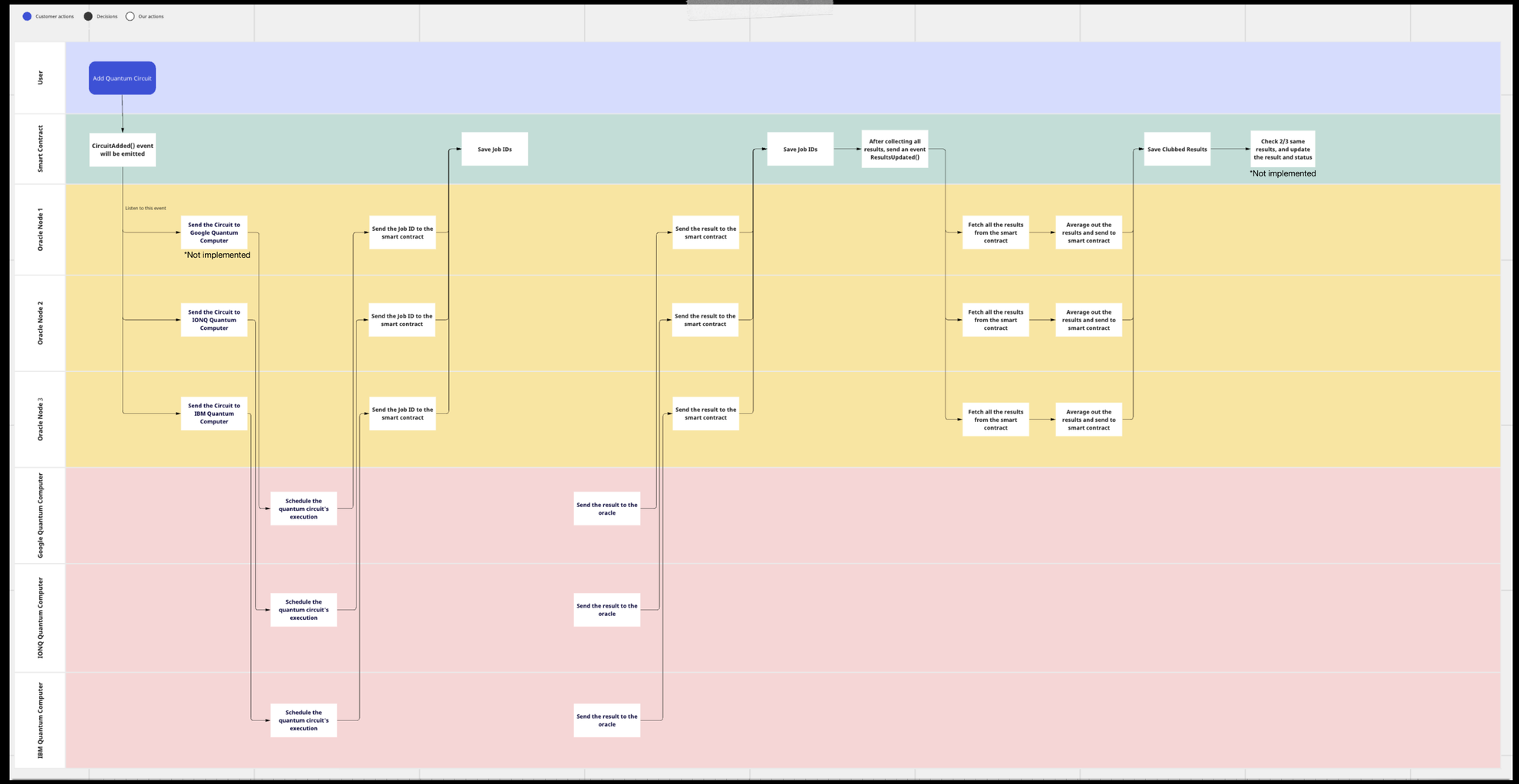
Smart Contract
- Send quantum program via our oracles to quantum service
- Store hash of circuit and quantum program results

Oracles
- Send quantum program to quantum computing service
- Retrieve results from quantum computing services

Architecture



Flow Diagram



https://miro.com/app/board/uXjVNVN3tMU=/

Upload

QOracle
Decentralised Oracle for your Quantum Circuits

Home | Upload Quantum Circuit | QOracle Computations | Debug Contracts

9.9142 CFLR
Coston

0X11...FF4C

Welcome to
QOracle

Upload your Quantum Circuit
Upload tab.

Find the results of your Quantum
Computations here

Fork me · Built with ♡ at BuidlGuidl · Support

Results

QOracle
Decentralised Oracle for your Quantum Circuits

Home | Upload Quantum Circuit | QOracle Computations | Debug Contracts

9.9142 CFLR
Coston

0X11...FF4C

Upload your Quantum Circuit

File input *

Upload your QASM file

Set your Circuit

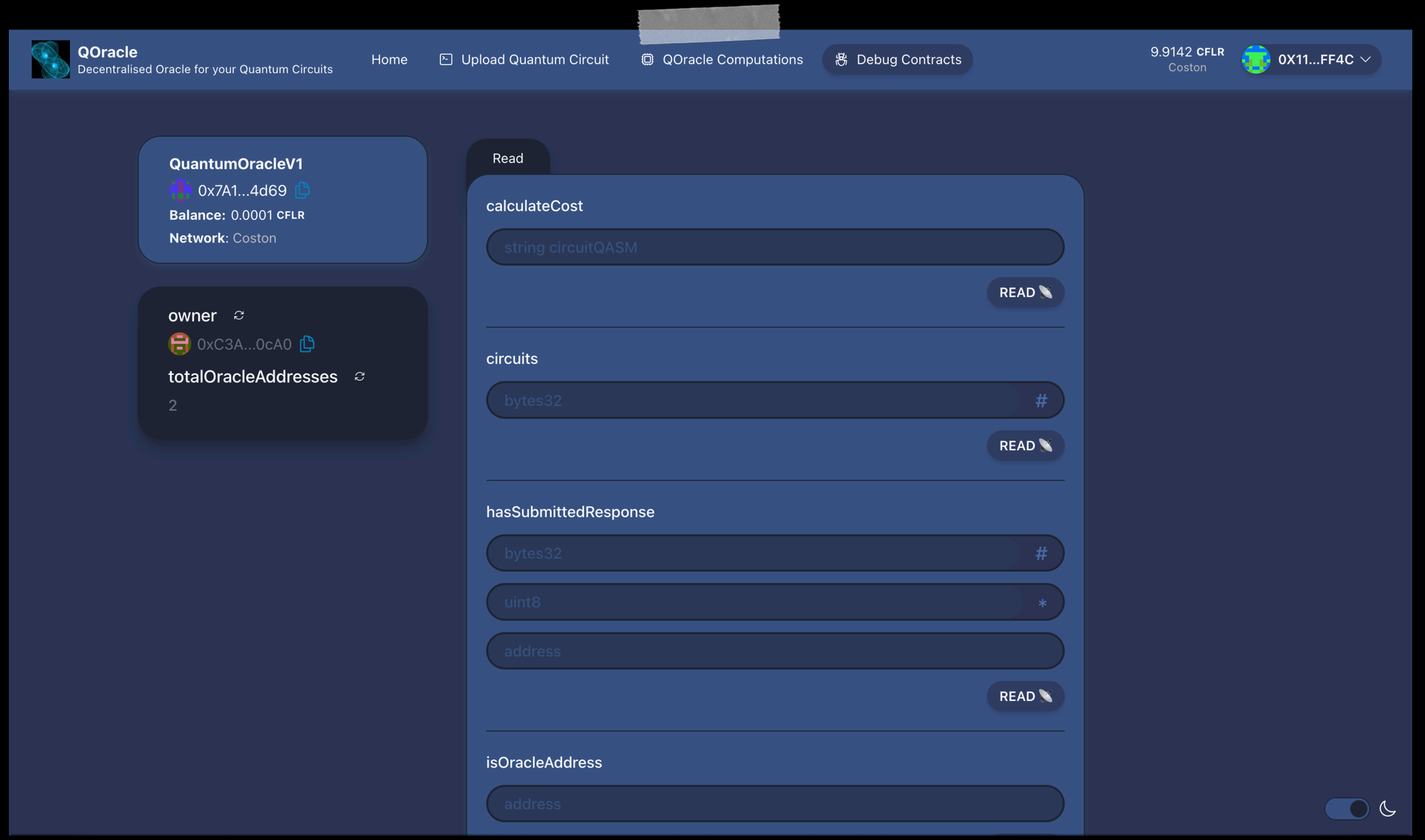
Display Circuit
Your Circuit

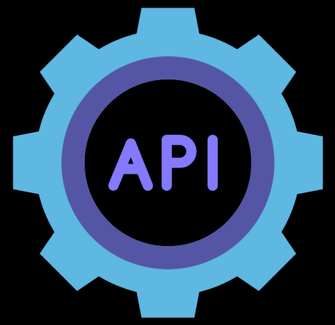
```
OPENQASM 2.0;
include "qelib1.inc";
qreg q[3];
creg meas[3];
h q[2];
x q[0];
y q[2];
h q[2];
h q[2];
cx q[0],q[1];
cx q[0],q[2];
barrier q[0],q[1],q[2];
measure q[0] -> meas[0];
measure q[1] -> meas[1];
measure q[2] -> meas[2];
```

Result: {"110":2502,"111":2522}

Fork me · Built with ♡ at BuidlGuidl · Support

Smart Contract Functions





Add circuit
Draw circuit
Retrieve results

FastAPI 0.1.0 OAS 3.1

/openapi.json

default

POST `/circuit` Create Circuit

POST `/draw` Draw Circuit

POST `/result` Get Result

Schemas

Body_create_circuit_circuit_post > Expand all object

Body_get_result_result_post > Expand all object

HTTPValidationError > Expand all object

ValidationError > Expand all object

How it began...

The idea was born at the Oxford Blockchain Hack a week ago. During that hackathon, we implemented:

- A simple frontend (not using Scaffold ETH)
- A quantum API with one quantum service provider (IBM)
- A smart contract with Chainlink functions deployed on Polygon
- We built these components individually - they were not integrated together

For this hackathon, we've created a new project based on the same concept. This new project includes new code implementation, new sponsor integration, and improvements to the concept. For this hackathon, we implemented:

- A new frontend using Scaffold ETH
- Other service providers to the quantum API in order to add decentralisation to our app
- Our own oracles to call the quantum API
- Deployment of the smart contract on Flare
- Deployment of the quantum API and API3 Airnode on Google Cloud
- Deployment of an API3 Airnode and integrating its dAPI price feed in our computational cost calculations

Team

Jan Ole Ernst
PhD in Physics

Specialising in experimental quantum control
Wealth of web3 experience

Yash Goyal
Digital Nomad

Specialised in blockchain development
and backend development

Jessica Pointing
PhD in Physics

Specialising in quantum neural networks
Web3 security researcher

+ Brandon Severin (emotional and slide show support)