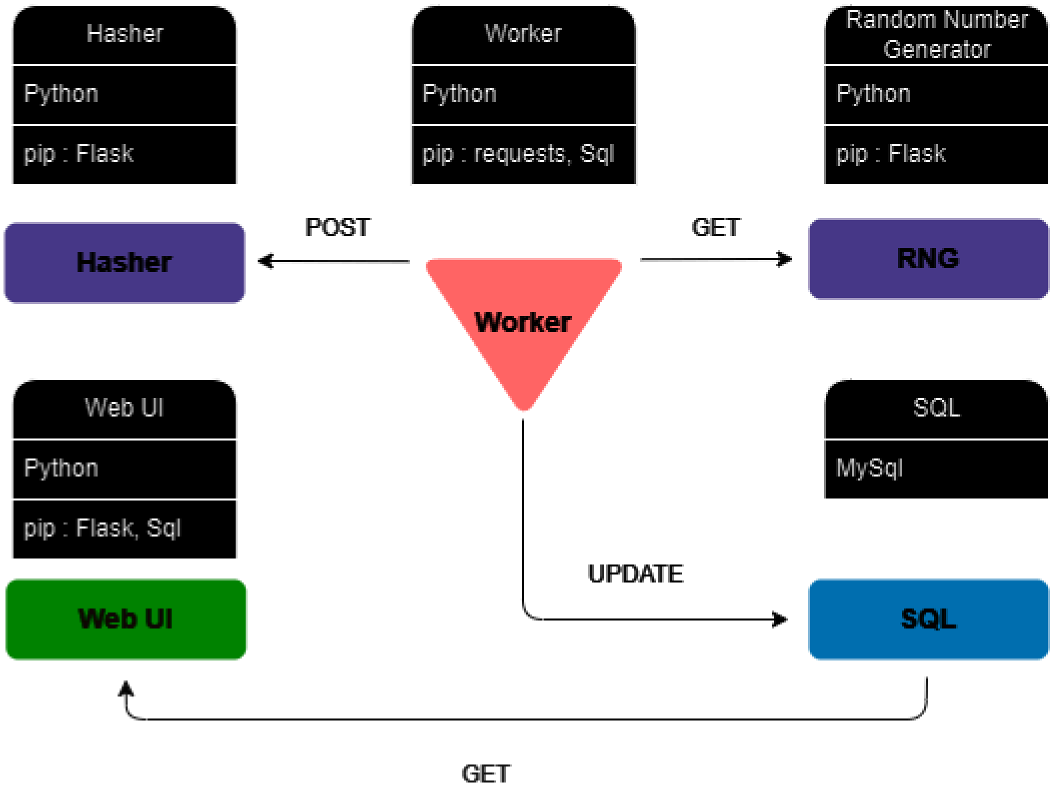
Group 9

Zachary Leopold, Sean Berlin, Kirtan Chavda, Jared Colletti

Coin Miner


Hasher
Python
pip : Flask
Worker
Python
pip : requests, Sql
Random Number Generator
Python
pip : Flask
Hasher
POST
Worker
GET
RNG
Web UI
Python
pip : Flask, Sql
SQL
MySql
UPDATE
Web UI
SQL
GET

Introduction

The Coin Miner Project is a distributed cryptocurrency mining system. The system comprises several components, each hosted on its own Linux machine in the cloud. These components include a Worker, a Random Number Generator (RNG), a Hasher, an SQL database, and a Web UI. The primary objective of the system is to mine cryptocurrency efficiently and provide a real-time mining rate to users.

Component Description

- Worker
  - The Worker component orchestrates the mining process.
  - Worker utilizes the requests library in Python to interact with Hasher and RNG.
  - Sends GET requests to the RNG to acquire random bytes.
  - Sends POST requests to the Hasher to hash the received bytes.
  - Updates the SQL database to record the number of iterations completed.

- Random Number Generator (RNG)
  - The RNG component continuously generates a stream of random bytes.
  - Flask is used to create an API endpoint.
  - Listens for GET requests from Worker and responds with a specified number of random bytes.

- Hasher
  - The Hasher component receives bytes from Worker and hashes them.
  - Utilizes Flask to create an API endpoint for receiving POST requests.
  - Hashes the received bytes and returns the resulting hash value to the requesting Worker.

- SQL Database
  - The SQL database stores mining-related data, including the number of iterations completed by Worker.

    - Workers update the database to record their progress in the mining process.

- Web UI
    - Queries the SQL database to retrieve the number of iterations completed.
    - Calculates the mining rate based on the retrieved data.
    - Displays the mining rate on the Web UI page in real-time for user monitoring.

Project Architecture

The Coin Miner Project follows a distributed architecture, where each component is hosted on separate Linux machines in the cloud. Communication between components is facilitated through API requests. Workers utilize the requests library to interact with the RNG and Hasher components, which are implemented using Flask. The number of iterations taking place in Worker is updated to the SQL Database. From there, the web UI will fetch the latest information from the database, compute the mining rate, and display it to the user.