TEAM BLOCKBUSTERS

Theme Song: 'Who ya gonna call... Block Busters!

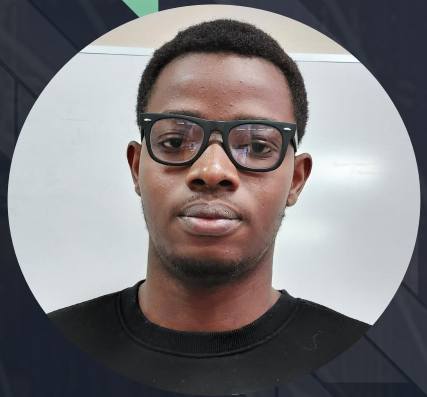
Dr. Keith Maull
Mentor

Agbeli Ameko
Mentor

Ibrahim Oluwajoba Adisa
Co-Mentor

Faiz Ikramulla
Student

Christopher Metellus
Student



Abdul Baqiy Diyaolu
Student

Team BlockBusters (Blockchain + IoT)

Goal: Our goal is to leverage city data stream(s) from IoT devices to build a reusable blockchain based application that supports smart contracts, traceability, and data sovereignty for the benefit and incentivization of city residents.

MVP: Add live data streaming into the bbserver.py , and have it call pysimplechain and ideally work as a flaskapp.

Github: https://github.com/rollercoaster111/block-busters

Team BlockBusters (Blockchain + IoT)

What is a Blockchain?

Blockchain is a shared, immutable ledger that facilitates the process of recording transactions and tracking assets (data, house, car, land) in a business network thereby reducing risk and cutting cost for citizens and governments (IBM, 2021).

Why would citizens St. Louis want to this?

Blockchain technology can provide a highly digitalized, advanced and sustainable urban management model for mobility, energy efficiency, waste treatment and citizen participation in the city of St. Louis.

According to Blockchain4Cities researchers, the most important benefits of blockchain for running cities include:

Increased transparency and connectivity, Direct communication, Information Integrity, and Efficient management

(https://www.iberdrola.com/innovation/blockchain-for-smart-cities-urban-management)

Implication for St. Louis

Blockchain technology can provide a highly digitalized, advanced and sustainable urban management model for mobility, energy efficiency, waste treatment and citizen participation in the city of St. Louis. Specifically, the implementation of this project will be beneficial to the city of St. Louis by enhancing the following:

Security: It will improve the protection of data across all sectors in St. Louis.

Energy: It will make it easier for St. Louis residents to to automatically trade surplus energy with other members of the grid.

Mobility: Government can know citizens that use their cars daily and offer incentives to encourage public transport and improve air quality.

Waste: It can provide real-time information on waste level of St. Louis residents and allow for prompt collection of waste

Participation: It will give every resident of St. Louis an opportunity to be a part of the community.

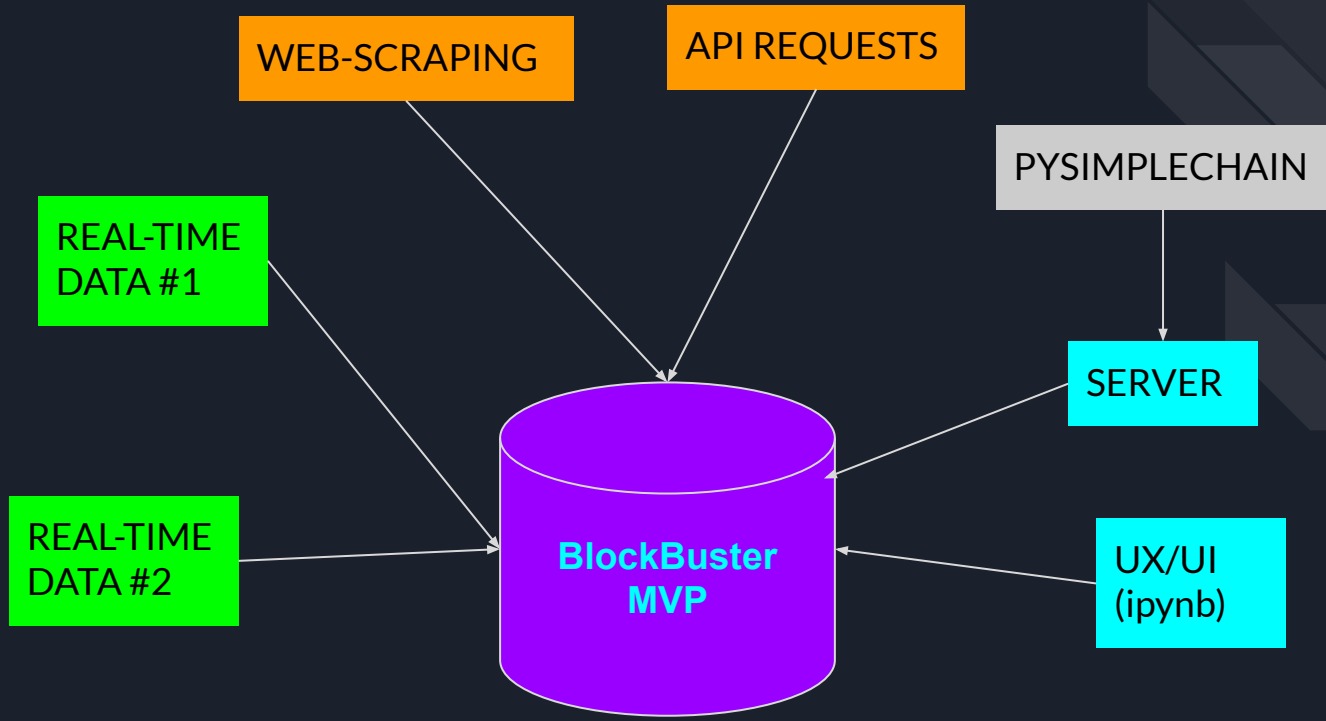
Project Timeline and Structure

Task	Status
1. Find existing repos of Blockhain applications/code/ and decide on the one to use.	Completed • PySimpleChain
2. Deciding on relevant data sources and begin cleaning	Completed • St. Louis Weather Data (AgEBB) • MesoWest Region Data • Static CSV Weather Data Files
3. Running and Testing of codes.	Completed *Demo Available*
4. Data Scraping and Cleaning	Completed Using Bs4 to scrape data from AgEBB Using Meso API to scrape data from MesoWest and clean data
5. Flask app & Server http:	Completed *Github Repo: 'Block Busters'*

Resources

- Google Colab
- Python
- Flask
- Google Cloud
- Meso API
- BeautifulSoup

BlockBuster MVP



PySimpleChain.py

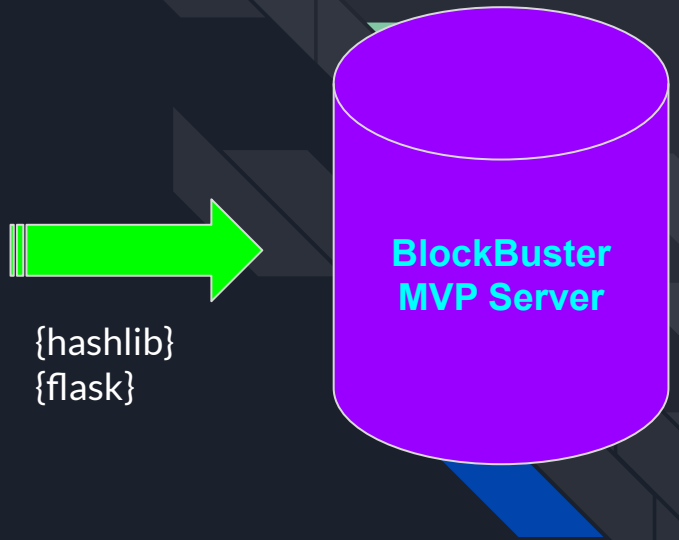
class Message : GET /add

class Block : POST /chain

class SimpleChain : POST /chain

class manager : GET /{block_hash}

```python
class Message:
    def __init__(self, data):
        self.hash = None
        self.prev_hash = None
        self.timestamp = time.time()
        self.size = len(data.encode('utf-8'))  # length in bytes
        self.data = data
        self.payload_hash = self._hash_payload()

    def _hash_payload(self):
        return hashlib.sha256(bytearray(str(self.timestamp) + str(self.data), "utf-8")).hexdigest()

    def _hash_message(self):
        return hashlib.sha256(bytearray(str(self.prev_hash) + self.payload_hash, "utf-8")).hexdigest()

    def link(self, message):
        """ Link the message to the previous one via hashes."""
        self.prev_hash = message.hash

    def seal(self):
        """ Get the message hash. """
        self.hash = self._hash_message()

    def validate(self):
        """ Check whether the message is valid or not. """
        if self.payload_hash != self._hash_payload():
            raise InvalidMessage("Invalid payload hash in message: " + str(self))
        if self.hash != self._hash_message():
            raise InvalidMessage("Invalid message hash in message: " + str(self))
```



{hashlib}
{flask}

Server

bbserver

Implementation of blockchain based on hypnopump's `pysimplechain`.

Run this server with the following:

```
python bbserver.py
```

api methods

[POST] `/add`

- : add a block to the chain
- input is a json ojbect of the form

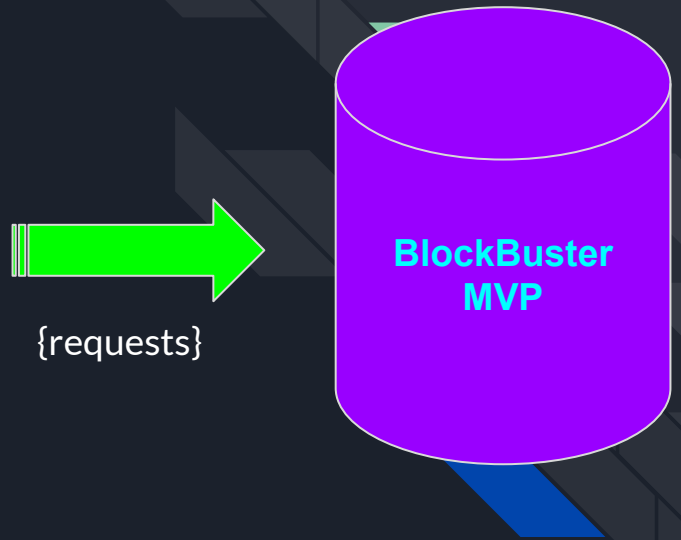
```
{
    "temp": 22.1,
    "rh": 55,
    "timestamp": "YYYY-MM-DDTHH:MM:SSZ"
}
```

[GET] `/chain`

- : show current chain

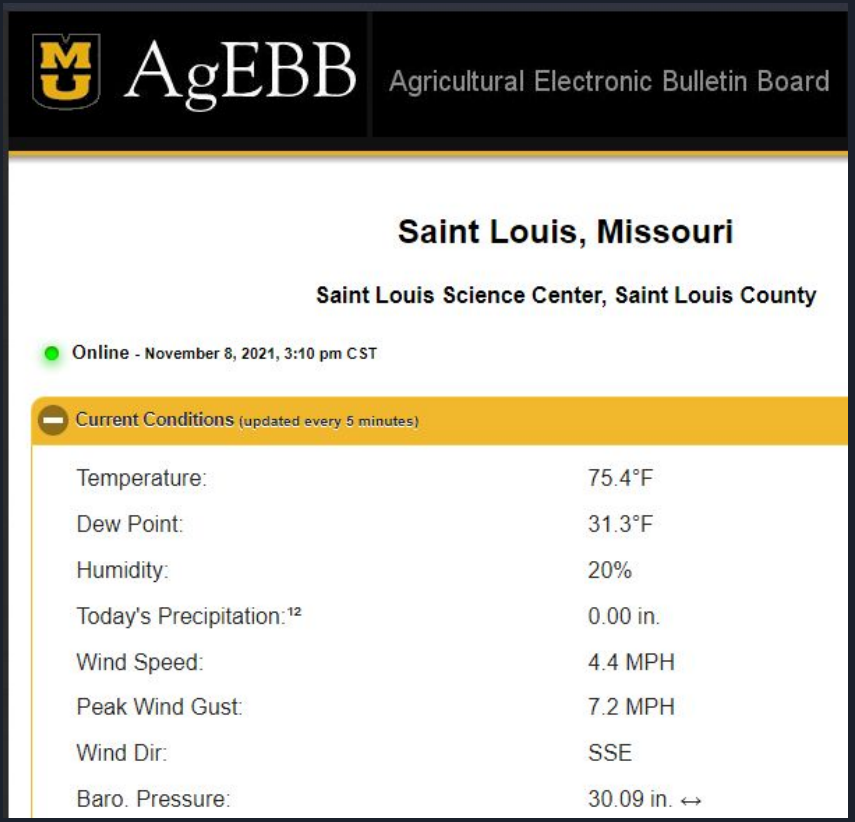
[GET] '/{block_hash}'

- : retrieve the data in a block

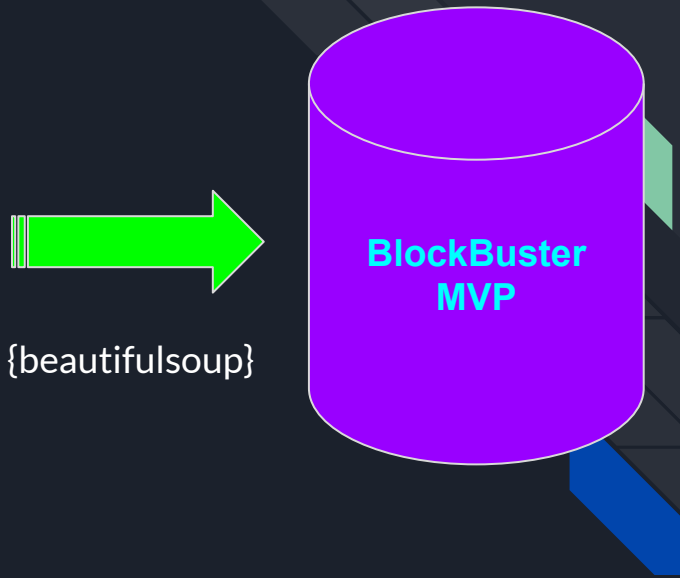


MVP Mode Demo #1

Demo #1: [Missouri AgEBB]

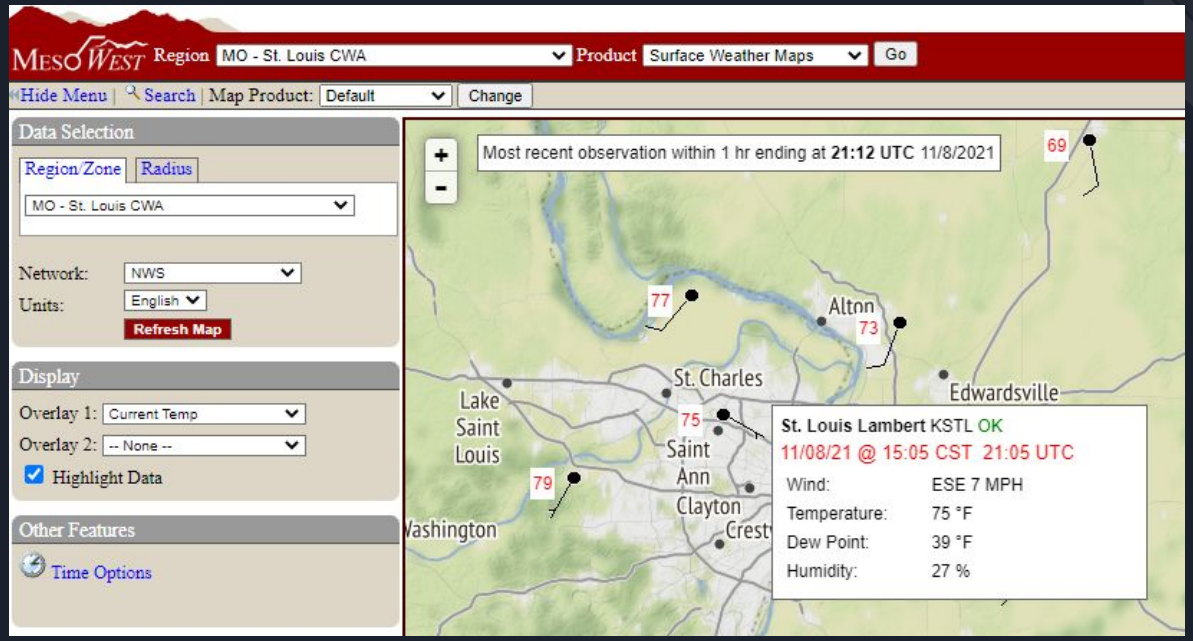


{beautifulsoup}

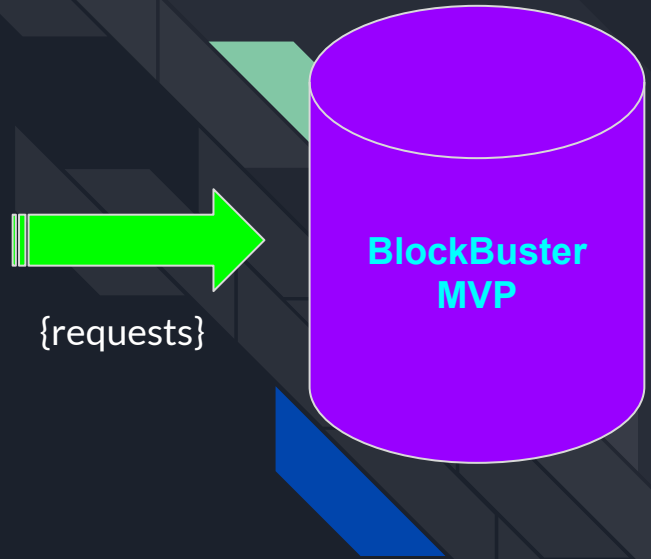


MVP Mode Demo #2

Demo #2: [MesoWest API]



{requests}



Status / Next Steps

STATUS

- Successful implementation of Blockchain for city weather data
- Built blockchain, server, automated data ingestion, validation tool

NEXT STEPS

- Scale Up to more data sources, points, types, blocks - (not just weather)
- Define incentive structure for citizens
- Integrate IoT device streaming data
- Test with citizens for feasibility / improvements

Conclusion


Thank You!


HACKHPC.ORG
INCREASING UNDERSTANDING BY HARNESSING THE RESOURCES, SKILLS, AND KNOWLEDGE FOUND IN THE HPC COMMUNITY