BLOCKCHAIN PROJECT

Team: The “Definitely CS Majors”

Project Software Engineers

Isaiah Mumaw - imumaw@nd.edu

Peter Ainsworth - painswor@nd.edu

Ed Stifter - estifter@nd.edu

Adam Mazurek - amazure2@nd.edu

Project Manager:

Julia Buckley - jbuckle2@nd.edu

Project Director:

Matt Morrison - Assistant Teaching Professor - University of Notre Dame - matt.morrison@nd.edu

Table of Contents

Section 1: Cerebration Sessions, Brainstorming, and Initial Design

I. Identified Unmet needs

As the popularity of cryptocurrency grows, an increasing number of people are asking questions about the underlying mechanics of blockchains, their level of security, and other processes related to them.

- It is an extremely new technology – most people first learned about it in the last few years, most notably during a major spike in the price of bitcoin in early 2021.
- People normally interact with blockchain in very abstracted ways. For example we see the buying and selling of a coin or NFT, but none of the underlying processes or code that make it happen.
- Blockchain is becoming increasingly relevant in the modern era due to the growing reach of cryptocurrency as a result of reduced trust in fiat currency and classical economic systems.
- As a new technology blockchain needs to gain the public's trust and the best way to do this is increase public understanding. Especially in areas like finance or healthcare public trust is vital for success.

II. Thought questions

- In what ways can we teach others about blockchain implementation and usage? How can we demonstrate the underlying computational process to someone unfamiliar with it?
- What is the minimum baseline knowledge we can assume others will have? How do we account for this in our documentation and visualization?
- How can this be successfully implemented in a way that is sufficiently thorough, yet within the constraints of this course?
- How can we implement an example blockchain for demonstration? How can we tie this to a broader point about real-world applications?
- Are there any ways to visualize the code and concepts in blockchain?

III. Resource Availability and Restrictions

Available Resources

- Project Director (Professor Morrison)
- Project Manager (TA Julia Buckley)
- Fellow group members
- Internet access for conducting project research
- Student-level access to CS servers and other university resources
- Full access to Python modules
- Other TA's, other students

Restrictions on Resources

- No direct experience with the subject, or access to someone currently in the industry who has worked with the subject
- Incomplete information about the baseline level of knowledge possessed by a potential user
- Limited time remaining in semester for testing, optimization, or revision

IV. Identifying Risks and Alternatives

Risks	Alternatives/Solutions
Insufficient complexity in building a basic blockchain, may complete project too early	Expanding beyond just implementing a blockchain, adding the ability for multiple machines to view/update the chain
Users may have problems understanding the code and reasoning behind it	Provide thorough documentation throughout the process in order to bolster understanding for those not directly involved in the project
Incompatibility between machines	Store data in a universal file format

V. Discussion and Debate

- Initial conversation began with the idea of doing something crypto related. Other subject matter were discussed (for example, Isaiah has some background in basic physics simulations through his major), however we continued to return to the idea of simulating some of the processes involved in crypto
- The next issue addressed was what exactly we could do that would meet the project requirements. We could, for example, build a simple market and use random number generators to simulate buying and selling. However, the scope of that idea was deemed far too broad, and we instead turned our attention to blockchains. While blockchains are very simple to implement, especially in Python, there is a lot of room to build algorithms and create much deeper projects with them.
    - A blockchain is a type of linked list, which allows us to modify some of the structures we have built in this course during Lectures 7-8 in order to implement it. In addition, our past experience with hashing in Lecture 15 will be of value when implementing our own hashing algorithm.
    - There was some debate over which language to use, as Python has such a wide variety of modules that we may abstract beyond needing to build our own data structures at all. However, C++ is more complex to code in and we may lose a lot of time just getting the basic elements of our project put together. We tentatively have settled on Python, under the assumption that we will try to build as many of our own functions as possible within the constraints of the language
- There was some concern expressed about the requisite background knowledge, as none of us has a complete understanding of the subject of interest. However, we compiled a number of articles that explain the underlying mechanics and also conducted some individual research in order to make up for gaps in our understanding. Further research is anticipated throughout the project as we continue.
- At this stage we realized that we still did not fully know our end goal, and that simply building a blockchain would likely not give us enough work. While we may have to adapt this as the project progresses, we decided that our end goal should be to build a blockchain which all four of us can access on our own machines, into which we can input messages and also see previous messages. This will require some method of storage that is available to all of us.

VI. Opportunities for Proofs of Concept

- We are looking at constructing our project in two stages:

    - The first part will be building the blockchain software itself. This includes the blockchain code, any algorithms that we create and networking code in order to create the distributed system.
    - The second stage of our project will be creating materials to explain and teach how blockchain technology works. This section will involve some code to visualize the steps in the blockchain. As of right now our plan for this part will be something along the lines of printing out the blockchain ledger and the code used for each step side by side so you can see how the underlying process works.

- The following tasks can all serve as a proof of concept for our project and we will treat them as checkpoints along the way. At each one we plan to take a step back and reevaluate where we are on our project, adjusting our end goals and project itinerary as necessary and addressing any major concerns.
    - **Create a rudimentary blockchain that can run on one computer.** This is not a true blockchain ledger since it cannot communicate with the outside world, but it would function as a proof of concept demonstrating that the project is feasible.
    - **Create sockets that can send messages to each other**. This will demonstrate that the networking basis behind the distributed network we plan to build is feasible, and allow us to implement a true blockchain ledger across multiple devices.
    - **Make a model output and graphics**. This will bring us to our end goal of teaching how the blockchain functions through visualizations in the command line and/or Python visual libraries.

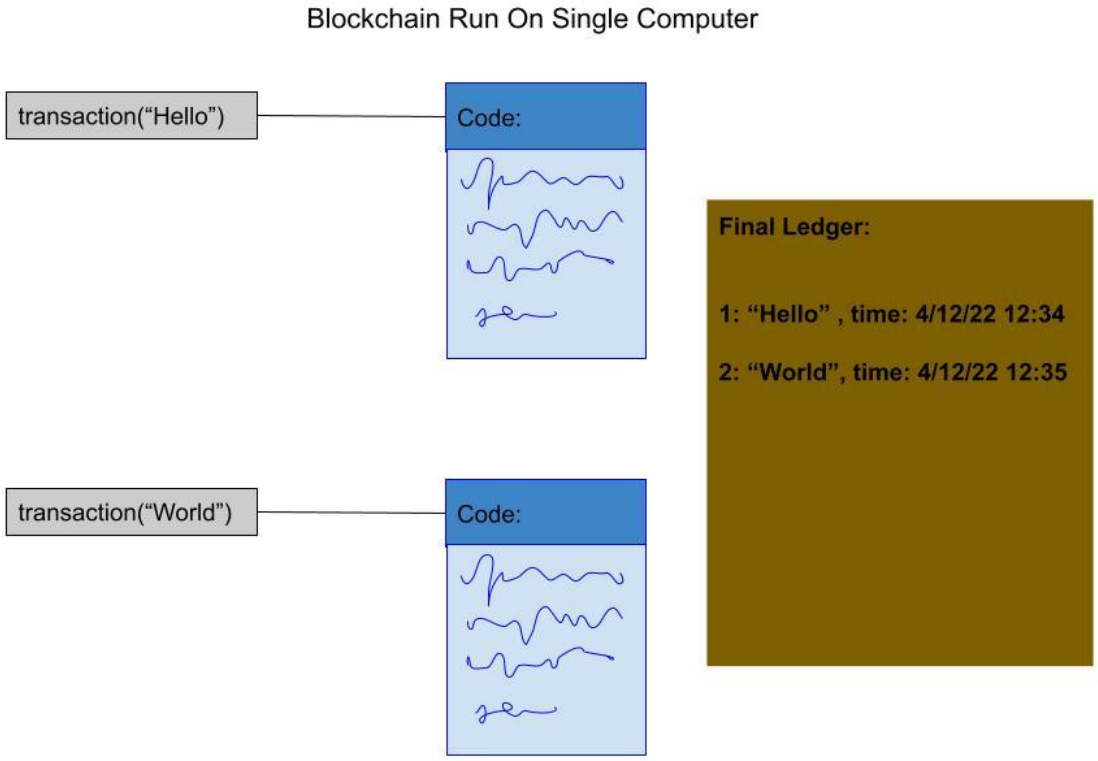

Figure: What we envision the output from our blockchain run on a single computer to look like.

Section 2: Key Terms and Definitions

Blockchain - a data structure comprising an ordered, reverse-linked list of **blocks**

Block - an element of the blockchain which contains a list of transactions and a reference to the **parent block**

Parent Block - previous block in the chain relative to current block

Hash - unique identifier for a single block. This hash is a compressed numeric value calculated by the **SHA256 hash algorithm**. Included in the hash is the hash of the parent block.

Proof of work method - Because the parent hash is included in the current hash, any changes to the parent block must be updated in the hash of the current block. With a block chain of hundreds of thousands of blocks, a change in one block, requires changes in all subsequent blocks. The immense computing power required to perform this ensures blocks remain immutable.

SHA256 Hash Algorithm - a one way function which generates a scrambled output of a constant size no matter the input size. The SHA256 hash algorithm is a part of the US National Security Agency's set of SHA-2 hash functions.

Genesis Block - first block in the block chain. This block is the common ancestor of all other blocks.

Mining - the process of confirming the validity of a new block through computational work

Section 3: Project Introduction Statement

The objective of our project is to build a simple blockchain from scratch to demonstrate the elements that make up a modern blockchain. Ultimately, we hope to make information about blockchain more accessible to lay people and those just beginning their programming careers. The explosion in popularity of cryptocurrencies and other blockchain-based technologies has led to greater awareness that blockchain exists, but not how it works.

We plan to address this ignorance by building and documenting a blockchain on which we can write messages that can be seen by every node with access to the chain. We will create tools and other functionality that will explain clearly what each step of the process of using a blockchain entails from creation of the chain to proof of work to adding nodes.

We will use Python as it is the most accessible language to lay people and beginning programmers alike, and our project will only be as valuable as it is understandable. We will use GitHub to make our code available to the public, and we will write extensive documentation so that those uninvolved in the design process will be able to get up to speed on our project quickly.

Section 4: Design Considerations

I. Assumptions

- We assume that users are able to navigate and interact with a basic web interface.
- We assume that users have no prior understanding of blockchain.
- We assume that there will be multiple nodes on the network so that the demonstration of the blockchain is not trivial.

II. Project Requirements

- This project will require the use of Python and some basic web technologies for pretty documentation and a simple interface.
- This project will require the use of a GitHub repository to share code between members and allow multiple developers to contribute to the project.
- This project will require us to distribute our blockchain design so that other nodes can join the network.
- The project will need some way to communicate over a network.
    - We can explore using cloud providers like Digital Ocean or Linode for cheap Linux server hosting, or we can work on a local network.
    - Remote hosting would allow us to operate our blockchain from anywhere.
    - Local hosting would allow us to demonstrate the project for free, but we would have to be on the same network.
- We will need to write very detailed documentation about the project, complete with code samples, diagrams, and adequate analogies for lay readers.
    - We will need to figure out how to produce pretty diagrams to illustrate our points (which will likely require learning about new Python libraries)

III. Specifications

- As explained above, different servers, whether it is a cloud provider or a local machine will need to communicate through each other using sockets
- These sockets will need to communicate with each other using json files. We have many options for what to apply the blockchain to and could make a few examples - perhaps one that passes fun messages to each other and another that does mock financial transactions.
- Each server will need to run a program that makes a cryptographic hash of the json data it receives and add it to its database.
- The ledger stored on each machine will need to use a data structure to store transactions. Because we want something that prioritizes being easy to search but also isn't too memory intensive, we will most likely use a tree structure.

- Lastly for the teaching aspect we need code that can run through the ledger and print out transaction history - this aspect will be written in python.

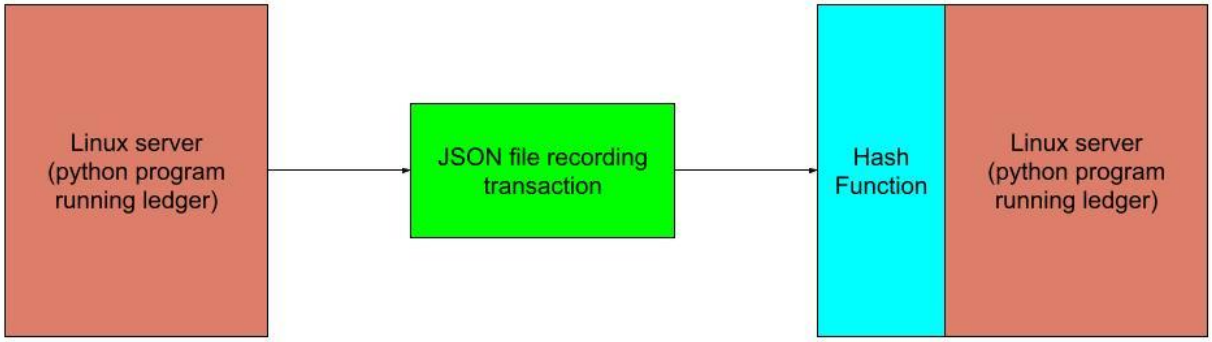

FIG 1. A hyper- simplified explanation of how the resources in our program will interact

IV. Initial RIsks and Alternatives

- Our biggest risk is finding and being able to properly use a cloud server to host our blockchain. If this function of our project fails we could use local machines to host the blockchain. This would obviously be a letdown since we want to make our blockchain as realistic as possible but our alternative would still allow us to demonstrate the concept.
- The second largest risk is there will be overwhelming our user with information on the blockchain. We need to make sure that we stick to a single narrative and do not get off track. This is more of a guiding goal as we work through the project.

V. Artifacts

- **Time Conflicts**
  - *Ed* – Out the week of 4/11 for surgery - will be available online and to work on project.
  - *Peter* – Project in Logic Design, Exam in Probability and Statistics.
  - *Isaiah* – Limited time on weekends, final paper and presentation for Advanced Physics Lab during last *full* week of class
  - *Adam* – Final Projects in Statistical Mechanics and Social Entrepreneurship, Work shifts in the afternoons.

- **Proof of Tools**
  - Coding software: Python is downloaded on the student machines and we all have github accounts already set up

    - Data storage space: Easy and cheap access to a linux server can be found at https://www.digitalocean.com/. This is something that we currently do not have but it can be easily obtained.

- **Miscellaneous requirements**
    - For this project our stockholders are people looking to learn about blockchain in their free time. Our target group will be college students. This is why we are trying to develop a front end report that is easy to use.
    - As explained in other parts of the project we have a clear picture of what resources we will need to complete the project and are confident in our ability to use them.

Section 5: Implementation of Data Structures Course Concepts

JSON Format

We will be coding our project in Python for its extensive hashing and distributed systems libraries and object oriented nature. Our code will be structured through a series of compounding data structures. First, each block will contain a JSON structure to store its data. This will include its index, timestamp, proof, and previous hash. Because this data is constant per block and does not need to be accessed repeatedly, a JSON will be used instead of a Python dictionary for its textual string format (which will enable hashing as discussed below). At the same time, the JSON provides key-value pairs for our data to be properly labeled.

Python Classes

Next, each block will be structured as a simple Python class. By creating blocks as their own objects, blocks will be able to interact with each other through the use of methods (like hash, add block, add transaction, and others) while storing their data as described above.

Hashing

To add the security of a blockchain, the data of each block will be computed as a hash. This will become the identity of the block. If the data of the block changes even a little bit, the resulting hash will be completely different. To perform this, we will utilize the SHA-256 hash function imported as a library. This algorithm performs a series of operations including And, Xor, Rot, Add, mod 2^{32}, Or, and Shr to produce a unique 256 bit output for any input. The nature of these operations makes the function strictly one-way, giving each block a secure identity.

Linked List

Because the hash of each block includes the hash of the previous block, all blocks are linked together as a chain. This acts similar to a linked list in which a pointer at each node points to the next. Here, however, each block points to the previous, allowing blocks to be concatenated to the end of the list as they are created. This allows traversal from the current block all the way to the starting block. Furthermore, this adds immutability to the chain as modifications in one block require re-hashing for all subsequent blocks.

Section 6: Goals and Timeline

Date	Goal
Thursday 4/7	All members - Complete and submit project proposal rough draft
Tuesday 4/12	All members - Proposed Project Manager Meeting All members - Revise and submit final project proposal
4/12 - 4/15	Week before easter break (Ed out for surgery) 4/12 - Meet to begin code and set expectations for CP1 Construct basic blockchain in python by CP1 Includes block classes and hashing functions 4/14 - Travel day
Friday 4/15	Checkpoint 1 Code Review and Memorandum
4/15 - 4/18	Easter Break Review comments from CP1 Brainstorm goals moving forwards (No intense work expected in observance of the Holiday)
Monday 4/18	Travel day: Return to campus
4/19 - 4/22	4/19 - Meet to review brainstorming ideas from over break for CP2 Build upon Proof of Work Algorithm Begin distributed systems model Brainstorm visual layout of output
Friday 4/22	Checkpoint 2 Code Review and Memorandum
4/22 - 4/26	Finalize distributed systems model Produce visual output for learning purposes All members - Work on final Report
4/26	Final Submission Due