Trabajo de Grado I - Modalidad asistentes

Presentado por

Jhonathan Arevalo

Presentado para

Yesid Diaz

Asignatura

Trabajo de Grado I

Universidad

Iberoamericana

Bogotá D.C

Contenido

Introducción

En el contexto de la evolución tecnológica y la creciente importancia de la seguridad y la integridad de los datos, el presente documento técnico explora el desarrollo de una aplicación en Python 8.2.0, aprovechando el poderoso framework Flask 2.1.1. El propósito central de esta aplicación radica en la implementación de un proceso de Blockchain en un documento de texto (.txt) que contiene datos de ejemplo. Este enfoque busca ofrecer una perspectiva completa de la arquitectura, las tecnologías utilizadas y la implementación práctica de la aplicación.

Contexto Tecnológico

En la era digital actual, la seguridad de la información y la confianza en la integridad de los datos son fundamentales. En este contexto, las tecnologías de cadena de bloques han emergido como una solución robusta para garantizar la inmutabilidad y la transparencia de los datos. Python, conocido por su versatilidad y legibilidad, se convierte en la elección idónea para el desarrollo de esta aplicación, mientras que Flask, un framework ligero pero poderoso, proporciona las herramientas necesarias para construir aplicaciones web de manera eficiente.

Objetivo de la Aplicación

El objetivo primordial de la aplicación es aplicar un proceso de Blockchain a un documento de texto (.txt) que contiene datos de ejemplo. Este proceso implica la creación de bloques de datos enlazados mediante funciones criptográficas para asegurar la integridad de la información. La aplicación no solo lleva a cabo esta implementación, sino que también presenta visualmente la cadena de bloques resultante a través de una tabla y un gráfico, proporcionando una experiencia intuitiva para comprender el funcionamiento interno del Blockchain.

Arquitectura de la Aplicación

La arquitectura de la aplicación se basa en un modelo de servidor web desarrollado con Flask. El proceso de Blockchain se lleva a cabo utilizando bibliotecas específicas en Python para el manejo de criptografía y transacciones. La aplicación sigue un enfoque de arquitectura de tres capas, que incluye:

Capa de Presentación (Flask):

Flask se utiliza como el marco de desarrollo para la capa de presentación. Maneja las solicitudes HTTP y responde con las vistas correspondientes. Además, gestiona la comunicación con la capa de lógica de la aplicación.

Capa de Lógica de Aplicación:

En esta capa, se implementa la lógica principal de la aplicación, que incluye la generación y verificación de bloques de Blockchain. Aquí se utilizan las bibliotecas de criptografía de Python para garantizar la seguridad y la integridad de los datos.

Capa de Datos:

Aunque la aplicación no requiere una base de datos persistente, se utiliza la capa de datos para almacenar temporalmente la información del bloque y el documento TXT.

Dependencias y Tecnologías Utilizadas

Python 8.2.0:

- **Descripción:** La versión específica de Python utilizada en el desarrollo de la aplicación.

Flask 2.1.1:

- **Descripción:** El framework utilizado para construir la aplicación web y gestionar las solicitudes HTTP.

Bibliotecas de Criptografía:

- **Descripción:** Se emplean bibliotecas de Python, como hashlib, para implementar la funcionalidad de Blockchain y garantizar la seguridad de los datos.

Interfaz de Línea de Comandos (CLI):

- **Descripción:** Se proporciona una interfaz de línea de comandos para interactuar con la aplicación y ejecutar el proceso de Blockchain.

Otras dependencias:

- **asttokens==2.0.5:** Utilizado para analizar el código fuente de Python y tokenizarlo.
- **backcall==0.2.0:** Proporciona herramientas para el manejo de llamadas de retorno (callbacks).
- **click==8.1.2:** Utilizado para la creación de interfaces de línea de comandos.
- **colorama==0.4.4:** Proporciona métodos para imprimir texto en colores en la consola.
- **debugpy==1.6.0:** Una herramienta de depuración para Python.
- **decorator==5.1.1:** Proporciona herramientas para la creación de decoradores en Python.
- **distlib==0.3.4:** Biblioteca para distribuir y trabajar con paquetes de Python.
- **entrypoints==0.4:** Descubre y carga puntos de entrada en Python.
- **executing==0.8.3:** Proporciona una abstracción para rastrear la ejecución de código.

Proceso de Blockchain

Generación de Bloques:

La aplicación implementa un método que calcula el hash utilizando el algoritmo SHA-256. Este método se encarga de tomar la información del bloque y generar un hash único que garantiza la integridad de los datos.

Clase Blockchain y Constructor:

Se ha diseñado una clase Blockchain para gestionar el proceso de Blockchain. El constructor de la clase se encarga de inicializar la cadena de bloques, creando una lista vacía que almacenará los bloques.

Creación del Bloque Génesis:

La aplicación incluye un método dedicado para crear la primera cadena de bloques, también conocida como 'Bloque Génesis'. Este bloque inicial es esencial para establecer la base de la cadena de bloques y contiene información específica sobre el inicio del proceso.

Creación de un Nuevo Bloque:

La aplicación ha implementado un método que permite la creación de un nuevo bloque y su posterior agregación a la cadena de bloques. Este método se encarga de recopilar la información del nuevo bloque, calcular su hash, y vincularlo al bloque anterior, asegurando la continuidad y la validez de la cadena de bloques.

Estos métodos garantizan un flujo coherente y seguro en el proceso de Blockchain. A medida que se añaden nuevos bloques a la cadena, se verifica la integridad de la información mediante el cálculo de hashes y se asegura la conexión lógica entre los bloques sucesivos. Este enfoque contribuye a la robustez y la confiabilidad de la aplicación de Blockchain desarrollada con Python y Flask.

Plan de Trabajo

Semana	Actividades
1-2	• Investigar conceptos clave de Blockchain y criptografía. • Establecer requisitos y funcionalidades básicas del proyecto. • Crear un plan detallado con hitos y entregables.
3-4	• Instalar Python y Flask, asegurando la compatibilidad. • Configurar un entorno virtual para el proyecto. • Crear un repositorio en GitHub y realizar la primera commit.
5-6	• Escribir funciones para el cálculo de hash. • Implementar la clase Blockchain con su constructor.
7-8	• Desarrollar el método para crear el bloque génesis. • Definir la estructura de datos para la cadena de bloques.
9-10	• Configurar rutas y vistas en Flask. • Diseñar una interfaz amigable para usuarios.

11-12	• Conectar la lógica de aplicación con la capa de presentación. • Pruebas de integración y resolución de problemas.
13-14	• Crear documentación para usuarios y desarrolladores. • Optimizar el código y mejorar el rendimiento.
15-16	• Realizar pruebas exhaustivas de la aplicación. • Preparar materiales para la presentación final. • Realizar la redacción final del documento técnico.

Presentación

Link Presentación: https://www.canva.com/design/DAF0VkRRdW0/stVL5kh-XcN1t4I3WKMjCg/edit?utm_content=DAF0VkRRdW0&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton

Conclusiones

La aplicación de proceso Blockchain desarrollada con Python 8.2.0 y Flask 2.1.1 representa un paso significativo hacia la comprensión y aplicación de la tecnología Blockchain en entornos de desarrollo. A través de este proyecto, se logró implementar un proceso sólido y seguro que utiliza el algoritmo SHA-256 para calcular hashes, la clase Blockchain para gestionar la estructura de bloques, y métodos específicos para la creación del bloque génesis y la adición de nuevos bloques a la cadena.

La utilización de Flask como framework para el desarrollo web facilita la interacción con la aplicación, permitiendo una implementación elegante y accesible. La arquitectura de tres capas utilizada, con una capa de presentación (Flask), una capa de lógica de aplicación y una capa de datos, proporciona una estructura modular y fácilmente mantenible.

La integración con un repositorio de control de versiones, disponible en https://github.com/jarevaloing/BlockChain, mejora aún más la colaboración y la posibilidad de contribuciones externas.

En cuanto a la instalación y ejecución, el archivo Nota.txt proporciona instrucciones claras y concisas, asegurando que los usuarios puedan configurar el entorno de manera eficiente y sin complicaciones. La instalación de dependencias mediante el comando pip install -r requirements.txt simplifica la gestión de bibliotecas externas.

En conclusión, este proyecto no solo demuestra la viabilidad técnica de aplicar Blockchain en un contexto práctico, sino que también sirve como recurso educativo y punto de partida para aquellos interesados en explorar más a fondo las capacidades de esta tecnología. Con un diseño robusto y documentación detallada, la aplicación proporciona una base sólida para futuras mejoras y adaptaciones. La combinación de tecnologías utilizadas y buenas prácticas de desarrollo establece un estándar de calidad que puede ser ampliado y refinado en futuras iteraciones del proyecto.