



Brian - The AI Brain of Web3

Project Documentation

Introduction

Brian aims to be the AI-powered brain of Web3. With extensive training on web3 documentation and resources, Brian is equipped to provide comprehensive answers to your web3 inquiries and simplify complex concepts. Additionally, Brian features a transaction builder that allows you to communicate in plain English and effortlessly create and execute transactions with ease, making your web3 interactions more accessible and user-friendly. Whether you are a beginner or an experienced developer, Brian is here to assist you in navigating the decentralized world of blockchain technology.

Key Features

1. Fast and Accessible Web3 Information

The decentralized nature of Web3 often leads to fragmented and hard-to-comprehend information spread across various sources such as websites, documentation, blogs, and social media platforms. Brian addresses this challenge by utilizing its vast knowledge base and leveraging AI techniques to provide precise answers to questions about web3. It also simplifies complex concepts with EILI5 button, ensuring that also a newbie can understand. Brian also indicates the sources from which the information originates, giving users transparency and the ability to explore further.

Whether you need an introduction to web3 concepts or in-depth explanations of specific topics, Brian has you covered.

2. Transaction Builder

Brian goes beyond providing information and empowers users to interact with decentralized applications (dApps) within the ecosystem. One common barrier to entry for many users is the complexity of interacting with smart contracts and executing transactions like, swap, approve, supply, borrow, transfer. Brian simplifies this process by offering a transaction builder feature. Users can communicate their transaction intentions in plain English, and Brian will generate the corresponding smart contract interactions, making the process seamless and user-friendly.

One unique advantage of Brian's transaction builder is its ability to bypass the need for a functioning dApp frontend. In cases where the dApp's frontend is inaccessible or unresponsive, Brian allows users to interact directly with the underlying smart contract. By



routing transactions through Brian, users can still send transactions and interact with the protocol even if the dApp's interface is down.

Brian provides a robust and uninterrupted Web3 experience.

3. User-Friendly Interface

Brian offers a user-friendly interface that makes interacting with Web3 technology as easy as using a chatbot. Users can interact with Brian using plain English prompts eliminating the need to navigate complicated UIs and empowers them to effortlessly interact with decentralized protocols. With Brian, the Web3 experience becomes as seamless and familiar as having a conversation, enabling everyone to explore the decentralized world with ease.

Technologies Used

- **Gnosis Chain and Gateway RPC**
 - Gnosis Chain was the first network to be implemented in Brian, and we exploited its low transaction costs and fast speed to test the functioning of our powerful Transaction Builder. Thanks to Gateway, all the transactions sent to Gnosis Chain are fast and reliable, thus allowing all Brian's users to have a fast and efficient user-experience.
- **TheGraph**
 - TheGraph was used under the hood, being at the core of all our data extraction, AI-model training and user prompting analysis thanks to the different subgraphs present in the network. In the future, we aim at using TheGraph to provide real-time data about the DeFi world, like the status of different Liquidity Pools on Uniswap or the latest price of a given ERC-20.
- **AAVE and GHO**
 - AAVE and GHO was by far the biggest challenge regarding the transaction builder, but we are proud of the result: users can borrow tokens (and GHO) from AAVE by just prompting a single message!
- **NucliaDB**
 - NucliaDB is a robust database that allows storing and searching on unstructured data. It is an out of the box hybrid search database, utilizing vector, full text and graph indexes. NucliaDB is written in Rust and Python.
- **LangChain**
 - LangChain is a library that enables more precise interaction with Large Language Models. When it comes to training and utilizing these AI models, exploring prompt engineering techniques is of utmost importance. In our project, we extensively experimented with different strategies, including the implementation of CoT (chains of thoughts, zero shot, few shot, etc.).



Additionally, we capitalized on the emerging capabilities of LLMs, particularly through few-shot learning. By generating high-quality examples for the desired inputs and expected outputs, we achieved robust responses from the utilized artificial intelligence models

- **Kor**
 - LangChain is a cutting-edge library developed to enhance the interaction with LLMs (Large Language Models). Its primary focus lies in enabling the extraction of structured fields from user text. During our prompt engineering experiments, we extensively employed this library to leverage its capabilities.
- **Hugging Face**
 - While private models like OpenAI's GPT ensure optimal performance, our objective is to utilize open-source models to ensure users' privacy. That's why we have already integrated our solution with models available on the Hugging Face hub, such as Flan-t5 and the recently released Falcon.
- **Pandas AI**
 - Among the solutions we investigated, one approach involved querying a structured file that contained manually crafted query-answer pairs for different transaction building scenarios. To determine the most suitable responses, we harnessed the capabilities of Pandas AI, which enabled us to assess the semantic similarity between the user's query and the use cases stored in our database. By leveraging this method, we successfully retrieved the most relevant answers for each query, ensuring both appropriateness and accuracy.
- **OpenAI**
 - We utilized OpenAI's state-of-the-art NLP models such as ChatGPT or text-davinci for our purposes.

Transaction Builder Smart Contract

Transaction builder contract code:

- <https://github.com/brian-knows/brian-contracts/blob/main/contracts/txBuilder.sol>

Brian's transaction builder contract is deployed on the following chains:

- Gnosis Chain: 0x4e1050F00150912B0D90FFd16dC58c8ea753c76C
- Sepolia: 0x3d22f2b0f8024e29cac15a52039aca3302842a24
- Taiko testnet: 0x3D22F2B0F8024e29Cac15A52039aca3302842A24

Brian's transaction builder contract ABI can be found here:

- <https://github.com/brian-knows/brian-contracts/blob/Gnosis%2BGateway/utis/transactionBuilder.json>

The Transaction Builder contract provides a main method called multiCall, which allows the frontend to encode and execute multiple transactions using the call function:

```
function multiCall(  
    address[] calldata targets,  
    bytes[] calldata data,  
)
```



```
uint256[] calldata values,  
address[] calldata tokenAddress  
) payable external returns (bytes[] memory) {  
    bytes[] memory results = _multiCall(targets, data, values, tokenAddress);  
    return results;  
}
```

This feature enables Brian to execute one or more transactions, batching them together facilitating user interactions. For example, it allows executing a supply and borrow operation to mint GHO tokens in a single transaction or swapping DAI for USDC and then adding DAI+USDC to a liquidity pool using addLiquidity.

The contract is capable of handling ETH or other native tokens of the blockchain. Additionally, it can return any remaining ERC20 token amounts back to the user after the transaction is completed. This functionality enhances the usability and convenience of the contract, allowing for efficient and flexible transaction execution.

Future Enhancements

The development team behind Brian is committed to continuously improving and expanding its functionalities. Some potential future enhancements include:

Enhanced transaction builder capabilities

Giving users the possibility to perform advanced operations such as voting in decentralized autonomous organizations (DAOs), minting non-fungible tokens (NFTs), and engaging in various complex DeFi activities. These features will empower users to actively participate in the decentralized landscape, making Brian the perfect tool for Web3 interactions.

Community-Driven Knowledge Expansion

In line with the ethos of decentralization, Brian aims to implement a decentralized autonomous organization (DAO) that allows users to actively contribute to the expansion of the knowledge base. Through this DAO, users will have the opportunity to propose new web3 resources, tutorials, documentation, and other valuable content. Additionally, the community will have the power to vote on which resources should be included in Brian's knowledge base, ensuring that it remains up-to-date, diverse, and reflective of the ever-evolving Web3 ecosystem. This approach encourages collaboration, inclusivity, and collective decision-making, fostering a sense of ownership and empowerment among Brian's users.



Natural language understanding improvements

Brian's NLP capabilities can be refined and expanded to better understand user queries and provide more accurate and context-aware responses. Ongoing research and advancements in NLP can be leveraged to enhance Brian's conversational abilities. One of our main objectives is the exploitation of open source LLMs (eg, flan-t5, falcon,...), allowing the users to leverage the capabilities of AI without compromising their privacy. Moreover we aim to create a text-2-code generative model. In this way the model will be able to construct transactions on his own without the “support” of developers as is currently happening. This will create an immense multitude of new use cases, users are going to be able to prompt trading strategies, recurrent transactions etc...

Smart Contract Writing Assistance

In the future, Brian aims to introduce a powerful feature that will assist developers in writing smart contracts. Leveraging its advanced natural language processing capabilities and deep understanding of the Web3 ecosystem, Brian will provide intelligent suggestions, code snippets, and best practices to help developers streamline the process of writing smart contracts. By providing valuable insights and guidance, Brian becomes an invaluable companion for developers, enabling them to build robust and efficient decentralized applications with ease.

Use cases explosion with code generation

We want to create a text-2-code generative model. In this way the model will be able to construct transactions on his own without the “support” of developers as is currently happening. This will create an immense multitude of new use cases, users are going to be able to prompt trading strategies, recurrent transactions etc... Brian will become a truly versatile and adaptable AI companion for Web3 interactions.

Conclusion

Brian is a groundbreaking project that aims to simplify the world of Web3 by providing fast and accessible information and empowering users to interact with decentralized applications effortlessly. With its AI-powered brain, comprehensive web3 knowledge, and user-friendly interface, Brian is poised to become the go-to resource for individuals seeking to navigate and understand the decentralized technology landscape. Embrace the future of decentralized technology with Brian, your trusted companion for accessible Web3 knowledge. Whether you are a beginner exploring the world of blockchain or an experienced developer looking for quick and accurate information, Brian is here to support you every step of the way. By bridging the gap between users and the complexities of Web3, Brian aims to foster inclusivity, enhance user experiences, and accelerate the adoption of decentralized technologies. Join us on this exciting journey and unlock the full potential of Web3 with Brian - The AI Brain of Web3.