# Storyline: Collaborative NFT Writing

Final Paper

Nathan Ang, Shane Aung, Jonathan Cheng, Ricky Lee

DApp: https://storyline-seven.vercel.app/
Source Code: https://github.com/storyline-team/storyline

## Abstract

Storyline is a blockchain based writing platform, where users come together to collectively write a story for the world to see. A storyline is composed of paragraphs (minted as NFT's), each from different writers. Each writer may contribute their unique ideas to lead the storyline in any direction they wish, collectively creating a creative and entertaining storyline. As a decentralized application, Storyline solves many of the limitations of current collective writing platforms, such as mutability, centralization, lack of anonymity, and limited incentive for writing. The platform is a space not only for readers to be entertained, but also for authors to express their creativity and be largely rewarded for it.

## 1. Problem

Collaborative writing platforms that are available today provide convenient ways to create written content with others, either through simultaneous editing via services such as Google Drive's suite of tools (Docs, Sheets, Slides, etc.), sequential contributions on personal blogs, or public media platforms such as Twitter. Tools like these are often structured and deployed as traditional web services or applications that require accounts on the respective platform to contribute or create content.

These categories of platforms solve many issues individually but still have their own problems. For example, Google Drive's suite of tools allows individuals to work on documents, spreadsheets, and slide decks together, where the content is generated and mediated by those who contribute to it. The content of these materials is easily mutable and deletable; however, the integrity of the content created is dependent on trusting those who are able to contribute to the content in question. Social media platforms such as Twitter allow for users to submit content in the form of posts as well as edits, but the content of these posts is regulated by a centralized, governing entity that can censor or take down content as they deem appropriate. Lastly, personal blogs may not be subjected to external censorship or regulation and can share content publicly if desired, but the ability to contribute and edit is restricted to a single party.

A single tool that solves all the problems mentioned above while still preserving the core ability to collaborate on content does not yet exist in the market at a large scale. This provides us with an opportunity to generate a platform that not only addresses these issues but also encourages individuals to submit content that is of high quality through automatic ownership and monetization. There is a legitimate desire in the market and business use case for a platform that removes the issues of centralized governance, trust while still allowing collaboration for written content.

## 2. Existing Work

Popular existing platforms for collaborative writing are currently centralized. These include Twitter threads, where Twitter users build off of each other's tweets in the form of jokes or a narrative, and fanfiction sites, where stories can be created collaboratively. Though these sites boast millions of users heavily invested in the storylines created on their platforms each day, they have their limitations. As mentioned before, they are centralized, which means the governing platform, and sometimes even the author of the parent post, has full control of the thread's lifespan and mutability. Next, users are limited to showing support for high-quality or entertaining portions through likes and shares which only exist within the platform without any monetary value or benefit.

Another related, existing work is mirror.xyz. They are creating a DApp for writing, but rather than collaborative writing, they are in the field of publishing. Each single-authored piece, such as an article, is published and distributed as an NFT. While some of their underlying technology likely has similarities to Storyline's, we plan to solve a problem that mirror.xyz does not, which is the limitations of collaborative writing on a centralized platform.

## 3 Solution

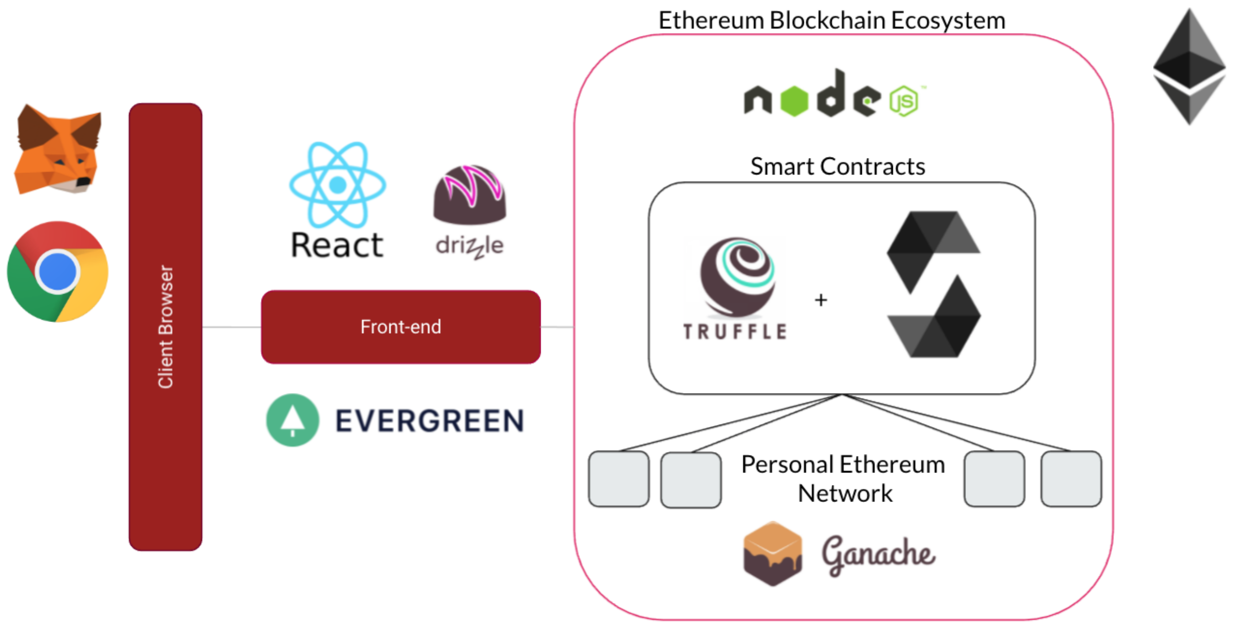


Figure 1: Architectural block diagram depicting the various technologies used in Storyline.

## 3a. Storyline Overview

Storyline is a blockchain based writing platform, where users come together to collectively write a story for the world to see. When a user visits our site, they can read the full Storyline so far, from start to finish, on the homepage. In addition, users will have the option to add new content, in the form of story elements, to the Storyline, at a small Ether writing fee. If their writing gets published to the blockchain, then the platform will display their addition at the very end of the story, appended to the rest of it. Finally, each story element is minted as its own NFT, and each user can buy, own, and sell them as they increase (or decrease) in value.

How is this different from a plain old website, or a google doc?

1. Google docs, websites, comment threads, etc, are all hosted by some governing service, which has full control over all content. Since this is blockchain-powered, **no governing service could add, delete, or modify content** previously written.
2. Users do not own the content they publish on normal websites or blogs, but they would own them here. There is **provable ownership** through the automatically minted NFT.
3. The distributed nature of blockchain networks ensure that it is extremely unlikely the Storyline gets destroyed in some freak accident. Decentralization of the data drastically **decreases the risk of data loss**.

This decentralizes the field of online writing, improving the experience for both authors and viewers. Since each paragraph in the story is its own minted NFT, our project allows each authored portion to be traded and tied to a monetary value. As demand to own an excerpt of the storyline increases, so will its market price. Additionally, each new paragraph needs to be minted and will be owned by the author, providing motivation to produce high quality and entertaining pieces that add value to the overall story. The platform will be a space not only for readers to be entertained, but also for authors to express their creativity and be rewarded for it.

## 3b. Architecture

Our application is built using the following languages.

- We used Solidity for our backend framework in order to develop our Ethereum smart contracts to represent components of the overall story, as well as NFT objects. The overall story is structured as an ethereum blockchain of story components.
- We used React and Javascript to develop our frontend application, which is connected to our smart contracts in the backend. Our website is styled using the evergreen-ui component library.

In our development process, we used several frameworks and tools to integrate, manage our version control process, and deploy our application.

- Ganache is an application that allows a user to run a local Ethereum network on their machine. We used this during development to test our application by connecting to a local Ganache network, which had several test accounts loaded with Ether. We could then test out our transactions (e.g. submitting story components, listing components for sale, purchasing components, etc) on this network.

- Metamask is a cryptocurrency wallet that is used to interact with different Ethereum blockchain networks. Users can open a browser extension available on Chromium-based browsers to access their Ethereum wallet and interact with decentralized applications. We used the Metamask Chrome extension to interact with our application locally when testing our transactions, as well as the deployed application.
- Truffle is a development environment and framework that allows us to work with our smart contracts in Solidity using the Ethereum Virtual Machine. We used this to compile our smart contracts and deploy them onto both our local Ethereum networks spun up from Ganache as well as the Rinkeby test network.
- Drizzle is a library that assists in connecting the frontend portion of decentralized applications by abstracting the process of instantiating web3 components, contracts, and accounts. We used this to link our frontend with the necessary ethereum smart contracts and wallets needed when interacting with our application.
- As a team, we used Github to maintain our code and manage version control.
- Our team used npm as our package manager given that our application was heavily based on Javascript.
- Vercel is a deployment platform for frontend frameworks that allows developers to quickly deploy static sites. We deployed our frontend application on Vercel, where it connects to our contracts deployed on the Rinkeby test network.

Our frontend application written in React is deployed on Vercel. Within the frontend application, the Drizzle package connects to the Rinkeby test network where our Solidity smart contracts were deployed using Truffle. Users can interact with our application by viewing the full story as the current state of the blockchain. They can also submit story components, list their components for sale, and purchase components by using the Metamask Chrome extension to send and receive ether.

## 3c. Smart Contracts

Here is an overview of the main smart contracts implemented:

**story.sol:** This is the primary smart contract for running storyline implementations. It supports getting the current storyline, getting specific story elements, creating a new story element (which mints its attached NFT), and buying/listing story elements which utilizes the underlying functions within the NFT smart contract.The story element list contains story elements, which is a custom struct with the form:

```
struct StoryElement {
  uint256 id;
  uint256 dateTime;
  string content;
  address author;
  NFT nft;
}
```

Figure 2. StoryElement struct

We decided to assign the id in a monotonically increasing manner, based on how many elements are currently in the storyline. For example, the first story element has an id of 1. This way, we can 'get' specific story elements in O(1) time complexity (i.e. story[_id-1]). Note the _id-1 due to the fact that the array indices are 0-indexed. This is an improvement over the naive array search which has O(n) time complexity, where n is the number of elements in the array. The front-end uses all the functions in this smart contract to interact with the storyline and individual story elements.

**nft.sol:** This smart contract is responsible for NFT functionality. It extends the ERC721[1] token contract, which is an accepted NFT standard token for digital ownership. We created functions for minting, buying, and listing tokens within our DApp, which uses standard functions/events of the ERC721 token such as _transfer() and Purchase().

**helper.sol**: This smart contract contains helper utility functions, such as uint256 to str conversions.

**migration.sol**: This smart contract is used to deploy our smart contracts onto the Ethereum network.

## 3d. Web Application

The web application is a critical part of the Storyline project. Without it, users wouldn't be able to use the smart contracts that support Storyline. While the smart contracts act as the point of contact between the user and the ethereum blockchain, the web app is how the user will interact with the contracts. Thus, it is essential to us that we make the web application as usable as possible, focusing on the two core Storyline features that users look for: blockchain-backed contributions, and NFT-ownership of writing.

There are four core technologies that power the web application: React, Truffle, and Drizzle, and Metamask. React JS[2], or React for short, is a library of modular components and

[1] ERC721 Token Standard, https://github.com/OpenZeppelin/openzeppelin-contracts/tree/master/contracts/token/ERC721

[2] React, 2021 Facebook, https://reactjs.org

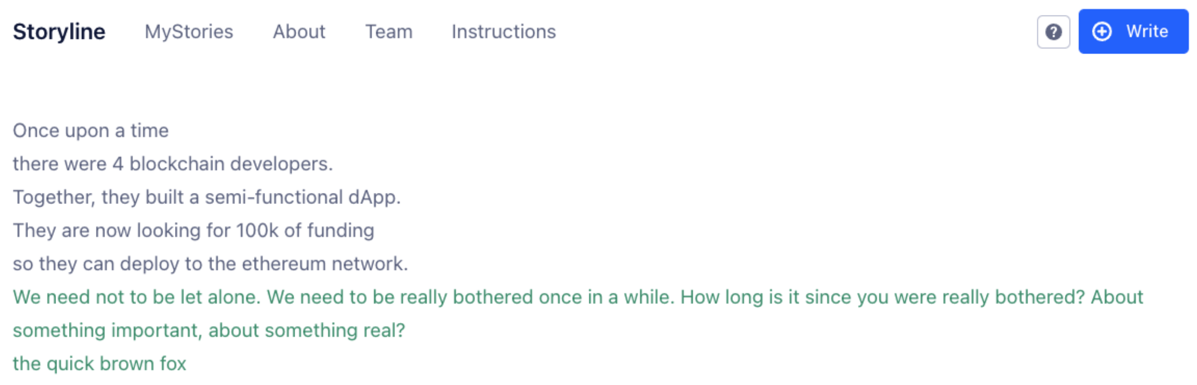
functions, written in Javascript, that can be used to construct dynamic and scalable frontend applications. It is an industry standard of web development today, and is what we use to render everything the user sees on Storyline. Truffle[3] is a development framework for Solidity (see section 3c), which doubles as the liaison between external code and on-chain smart contracts. Drizzle[4] is a small javascript library whose sole purpose is to connect the React frontend with the Truffle framework, thus connecting the web app to the smart contracts. Metamask[5] is a separate third-party browser add-on that enables Chromium-based browsers (Google Chrome, Brave, etc) to communicate with ethereum blockchains such as the one that Storyline is deployed on.

When a user visits Storyline, they are greeted by the Home page (Figure 2), which displays the full story consisting of all contributions from all authors in chronological order. Here, they can leisurely peruse the storyline and browse through the story elements to see if any are for sale (indicated in green). If the user finds an interesting story element for sale, they can purchase that NFT directly from the Home page by clicking on the element and completing the transaction. Importantly, users can contribute to the storyline through the "Write" button in the top right corner. This will open up a modal where the user will write their story element (up to 280 characters) and submit it to the application. When the submit transaction is mined by the chain, the user will be notified, and their brand new story element will be displayed on the Home page.

Moving on to the MyStories page (Figure 3), here the user can view all the story element NFTs that they currently own, as well as their market status (Hold = not listed for sale, List = listed for sale). Additionally, the user can list their stories for sale, so that other users can buy them. At the top of the MyStories page, the owner can simply choose the element they want to list for sale, set the selling price, and create the listing. Not only will the element's status change on the MyStories page accordingly, but the Home page will also reflect this status change by highlighting that story element in green.

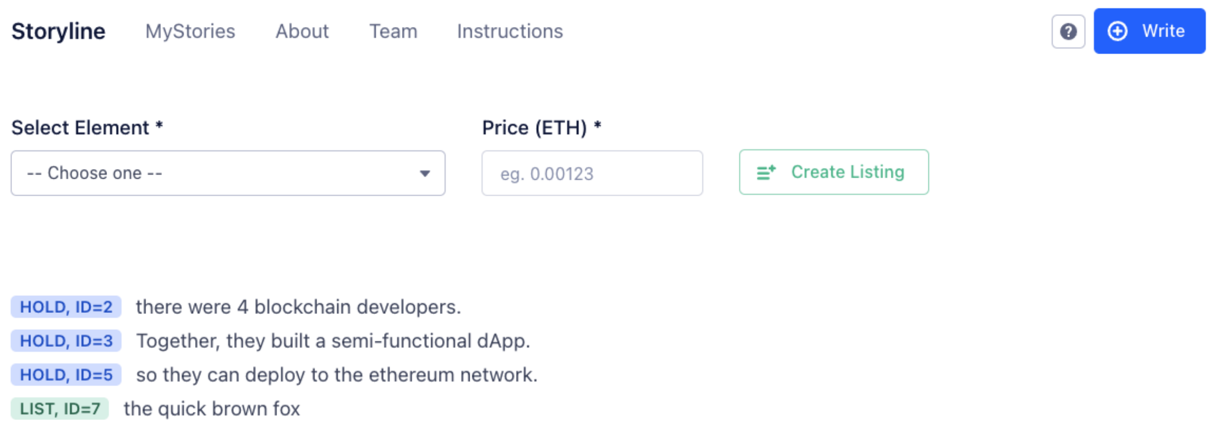
Additional pages include the About page, the Team page, and the Instructions page. These are static pages that provide information about what the Storyline project is, how it works, and the team behind it. Notably, the Instructions page includes all details and documentation about how to navigate and use Storyline for the beginner user.

[3] Truffle Suite, 2021 ConsenSys Software Inc, http://trufflesuite.com/index.html
[4] Drizzle, 2021 ConsenSys Software Inc, http://trufflesuite.com/docs/drizzle/
[5] Metamask, 2021 Metamask, https://metamask.io



Once upon a time
there were 4 blockchain developers.
Together, they built a semi-functional dApp.
They are now looking for 100k of funding
so they can deploy to the ethereum network.
We need not to be let alone. We need to be really bothered once in a while. How long is it since you were really bothered? About something important, about something real?
the quick brown fox

Figure 3. The Home page of Storyline



HOLD, ID=2 there were 4 blockchain developers.
HOLD, ID=3 Together, they built a semi-functional dApp.
HOLD, ID=5 so they can deploy to the ethereum network.
LIST, ID=7 the quick brown fox

Figure 4. The MyStories page of Storyline

To provide more transparency into the connections between different technologies in our stack, here is how a user interaction will propagate through to the ethereum blockchain. Lets say a user writes a new story element through the “Write” modal, and submits it. Then, the React JS frontend will deliver the plain text to Drizzle, which packages it up with the necessary transaction metadata and sends it to Truffle. Truffle then forms the transaction, and initiates it in tandem with Metamask, which will provide the ether gas funds necessary to complete the transaction. Our on-chain contracts then receive the gas, along with the transaction metadata and story element plaintext, and creates a brand new Story Element object, as well as mints it as an NFT, transferring ownership directly to the owner’s Metamask wallet.

## 3e Deployment

Storyline is categorized as a decentralized application (or DApp), as it is not hosted on a central server, but rather published to a blockchain network, thus living on many different nodes at once. In order to deploy DApps, we leverage the Truffle development framework to compile and migrate our contracts on chain. Instead of uploading code to a host server, as traditional applications do, Truffle can migrate our compiled Solidity code onto a chain node, which then (generally) propagates those changes to other nodes in the network through Infura[6], an endpoint provider for easy and secure access to public ethereum chains.

For the Storyline project, we played with the idea of deploying our contracts to the main Ethereum network, but gas fees and NFT minting prices were far too expensive (in the range of tens of thousands of USD). However, delivering a publicly available end product was important to us, so we opted to go for a public testnet[7] so nobody has to spend real money to test or use the app. Currently, our contracts are deployed on the Rinkeby testnet, as it's Proof-of-Authority mining scheme allows for faster average block times (~15s). In addition, we initialized our own set of Infura endpoints so that our truffle migrator can access the Rinkeby chain through HTTPS.

As for the React frontend application, which can be thought of as a separate entity that interacts with our contracts, we opted for a continuous deployment option, Vercel[8]. While Vercel is a full-featured development platform, we chose to solely use its deployment functionality. Vercel pairs well with React and Github (where our codebase lives); it will automatically detect code changes and redeploy the frontend application as necessary, with no required actions on the developer's end. Vercel and Infura, paired with React and Truffle, make deploying a seamless end-to-end DApp much easier.

Our functioning, deployed DApp can be publicly accessed here: https://storyline-seven.vercel.app/

## 4. Future Work

There is still future work to be done, driven by our current limitations. First, there is no content vetting process or capabilities. For this, we must implement a pre-minting check or filter to mitigate things such as hate speech. Another big potential risk is links being able to live on the storyline, which we can disallow through this filter. Next, there is no search, filtering, or querying functionality within the storyline. This can be solved with a search component on the front end component of our project. Another limitation is that there is only one storyline at the moment, but our project is extendable to have multiple storylines, for different topics and genres of written content. Finally, we plan to migrate this project from the Rinkerby Test Net to the Ethereum Mainnet and potentially list our NFT's on marketplaces such as OpenSea. We believe our documentation is clear and our software design architecture allows for this project to scale well.

## 5. Conclusion

Storyline is a functioning, publicly-deployed decentralized application that solves many of the limitations that come with collaborative writing on a centralized platform. Namely, these include mutability, centralization, lack of anonymity, and limited incentive for writing. Storyline allows for authors to build off each other's unique writing and ideas to form a single, entertaining

[6] Infura, 2021 Infura Inc, https://infura.io/product/ethereum
[7] Networks, Ethereum, https://ethereum.org/en/developers/docs/networks/#testnets
[8] Vercel, https://vercel.com/home

storyline for the public to view and interact with in the form of NFT's. The platform is a space not only for readers to be entertained, but also for authors to express their creativity and be largely rewarded for it.

The team has spent countless hours creating this DApp and have learned a lot along the way. We are proud of the work we've done and are glad to have created a platform that aims to revolutionize the field of collaborative writing as we know it.