In-Person Mobile Token Verification

Detailed Plan Addendum for Fund 10 Idea 100225

Dane McBride
linenotsecure@gmail.com

Category: Products & Integrations
ADA Requested: ₳111,216 ($27,804)
Percentage of Category: 1.22%
Estimated Completion: 5.3 months

Overview

Project details directly relating to the Ideascale submission requirements.

Problems Facing Real-World Event Teams

There is high demand for a simple in-person NFT verification process for events but there are none suitable for realistic demands of ticket verification with issues like re-entry and duplication.

Event companies utilizing top ticketing and check-in software like Ticketmaster still suffer from methods of ticket duplication, counterfeiting, and logistical and cost issues derived from managing paper tickets.

Paper tickets are one-time use and, unlike an NFT-based token gating system, cannot be dynamically programmed for access to future events nor can extraneous utility be easily stacked upon a paper ticket.

Identified Community Needs

The Cardano community in particular needs an easy-to-use side service for creating abstractions between NFT ownership and ticket identifier payloads. NFT themed conventions do not currently use NFTs as tickets! Such a system needs to be compatible with any check-in software already in use by ongoing events.

Proposed Solution

A self-service web app whereby event companies can replace the use of paper tickets with an NFT-based, in-person usage tracking system that is compatible with their existing check-in software.

The solution involves a token-gating resource server to provide wallet-based access to gate payloads, in this case tickets, and a Collections concept to group token gates, which represent tickets, into a single event object with a start and end date.

There is already a self-service token gating app online and working that was funded in Fund 8 and can already be used to add token-gated utility to project team’s own website. This proposal covers a specific package of work sprints that would create a complete, mobile-friendly, self-service event ticketing system that any Cardano project can use for NFT ticketing, as a standalone solution or in cooperation with the team’s check-in software of choice.

Primarily, the required work involves an optional mobile passcode login alternative to the existing wallet-based login scheme. It also requires a new interface for gate creators to scan wallet addresses to produce the gate payload on the creator's device, as opposed to on a third-party website on the NFT holder's device as it currently runs.

After finalizing ticket information on their software of choice as normal, event teams would export that ticket data and create a bulk upload file for this app, creating one token gate per ticket and associating them with assets in a specific NFT collection in one process. At the event, team members would use the proposed mobile passcode alternative to log into the app on a mobile device, and use a wallet scanning interface

The wallet scanning interface would replace the existing token-gating OAuth flow with a QR code camera library that scans receive addresses from consumer phones, using the Receive menu of the mobile wallet of choice. With a collection in mind and a wallet obtained, the interface not only confirms attendance but checks against passbacks by associating the first wallet used for that NFT into a token usage log for the event. The scan UI will reveal the ticket payload belonging to an NFT and the original registrar can reverify (re-entry) unlimited times, while ensuring that the same token cannot be used by a different wallet for the same event.

This proposal envisions a check-in desk with a laptop running web2 ticket scanning software as normal, attached to a handheld USB scanner. Beside that, a mobile phone belonging to the event team resting on a stand, logged into this app and on the Scan page. Almost nothing needs to change about how the check-in desk does business, except that instead of scanning a paper ticket, they are using their mobile phones to scan a wallet, to obtain the digital ticket QR, then scanning that into the system as if it were the paper version.

Main Goals

Provide a way for NFT projects to facilitate their own NFT-based event ticketing solutions on the fly, on their terms, without needing to know how to code.

Provide a product that leverages the advantages of blockchain to make web2 ticketing systems more secure than they could ever be when utilizing paper tickets, saving money on counterfeiting, duplicates, customer service issues with re-entry, and saving money on paper tickets and ticket logistics.

Become the first solution that the well-known Cardano conventions and meet-ups use instead of paper tickets.

Onboard the business world to Cardano by making an easy-to-use system that secures tickets as NFTs in ways the web2 ticketing solutions cannot, and yet is compatible with the ticketing systems they must continue to use.

Ecosystem Impact

This solution allows creators to monetize live events as NFTs instead of paper tickets, opening up the event hosting business to anyone with or can make an NFT project, eliminating the tech and financial barriers most ticketing systems present, while being even more secure against duplication.

This proposal provides an opportunity for the Cardano ecosystem to support mainstream event management companies in a seamless way, thereby bringing all their consumers into Cardano as NFT ticket holders.

Market Adoption

We will partner with existing content creators on Cardano as well as physical events in order to cross-promote artists with other events and projects, using this service layer suite as a go-between for branded third-party entertainment websites, token holders, and the web2 event management world. Our primary focus will be Cardano adoption - while we want to assist cNFT projects to start using the tool, we see a major opportunity to provide event teams with a true business case for using blockchain.

Open Sourcing

This proposal scopes work for a package of upgrades to an existing token gating as-a-service app. In return for not offering much to the developer ecosystem, the app in this proposal will be used to help event teams on Cardano run NFT-based event ticketing for the first time, at no cost to them, and will be used to outreach larger web2 event teams and encourage them to use Cardano.

Revenue Models

There are independent plans for adding features to this app that would require a monthly subscription payment in order to use higher levels of gate service and advanced gate features. However, that functionality is outside the scope of this project.

The revenue model that will be available to the project at the moment of launch is the concept of the login token. This app will sell login tokens through a popular minting service whose policy ID is synonymous with account profiles. As the user base grows, we may be able to self-fund our billing server without the proposal we wrote specifically for it.

Feasibility Audit

Functional Milestones

Our milestones are all fully-functional deliverables and are defined primarily by a successful online deployment and a phase of testing. Here are our milestones covering a 3, 6 and 12 month scope.

- **Q1**
  - **Ticket Creation Interface:** Create an account-based web app for users to create token-gated event tickets, NFT-protected gate configuration objects that associate NFTs with ticket information. Interfaces include Register, Login, Profile, Wallet, Collections, Gates, Verify and Scan. The Verify page is used by consumers to access 2FA codes after confirming NFT ownership and selecting a valid token gate. The Scan page is used by gate creators to scan 2FA codes and confirm they belong to an active gate, that the gate belongs to this Collection, and that the scanner has permission to obtain the payload as the creator of that gate. The result of a gate owner scanning a 2FA code related to one of his gates is that the gate owner receives the payload, not the consumer who owns the trigger NFT. This enables a NFT verification system that is perfectly platform-agnostic to the check-in systems real-world event teams currently rely on - the team is meant to use their legacy system to scan the gate payload from their scanning phone into the legacy system on their laptop.
    - 9 work sprints covering estimated 285 hours
    - Starting 10/8/23, target launch 11/18/23
    - Projected cost: ₳ 68,752 ($17,188)
  - **Gate Resource Server:** The API server that serves the NFT Verification Modal to verify token ownership, provide available token gates, and produce authentication codes that third-party sites use to securely obtain gate payloads which, in the context of this project, will be ticket QR codes.
    - 2 work sprints covering estimated 120 hours
    - Starting 11/21/23, target launch 12/8/23
    - Projected cost: ₳ 7,184 ($1,796)
  - **Ticket App API:** The API server that communicates between the Ticket Creation Interface app and the system's MongoDB data tables.
    - 9 work sprints covering estimated 274 hours
    - Starting 12/11/23, target launch 1/19/24
    - Projected cost: ₳ 38,008 ($9,502)
  -
  -
- **Q2**
  - **On-Page Script:** A module on the ticketing web app for token gate owners to scan consumer wallets, obtaining the ticket payloads (QR) belonging to gates triggered by tokens owned in the wallet. It includes a QR code camera scanning library that works on mobile and desktop, to scan wallets in a brand-agnostic way and verify ownership of NFTs belonging to token-gates tickets. It allows event teams to transpose NFT verification into a ticket identifier that can be scanned into any classic ticketing system that the team currently relies on. This means instant

application to currently run events, a cost savings in printing tickets, and a level of anti-fraud security for in-person events that cannot be matched with a paper ticket system.

- 2 work sprints covering 40 hours
- Starting 1/22/24, target launch 1/28/24
- Projected cost: ₳ 2,392 ($598)

- **MongoDB Database Sets:** The MongoDB Atlas account that contains NoSQL articles for Users, Wallets, Wallet Keys, Gates, Collections, Session Auths, Gate Payload Auths, and 2FA Codes. Each component in this system will use a unique database access key with the proper scope for the work.
    - 3 work sprints covering estimated 155 hours
    - Starting 1/31/24, target launch 2/22/24
    - Projected cost: ₳ 9,280 ($2,320)
- **Load Testing Period:** An in-person solution like this requires extensive load-testing and that must be done in a realistic environment. Therefore, the app must play host to several "beta testing parties", concerts and other get-togethers that steadily grow in size as performance issues are identified and worked out, while the audience is aware that the solution is in a beta phase.
    - 1 work sprint covering estimated 120 hours
    - Starting 2/25/24, target launch 3/13/24
    - Projected cost: ₳ 7,184 ($1,796)

- **Q3**
    - **Public Launch**
- **Q4**
    - **Onboarding Assistance**

Timelines for milestones are based on work-hour estimates for the work sprints involved in completing each. For details please see the proposal worksheet for this project.

Deliverables and Outputs

1. **Ticket Creation Interface**
    - a. **Login UI - Mobile Passcode Entry:** Make the login screen mobile responsive and apply an alternative login mechanism, the mobile passcode. A single input as opposed to a username and password. A Login button beside the passcode input initializes a new version of the login API, using passcode input instead of wallet address.
    - b. **My Profile: Mobile Login Opt-In:** Create a profile section to allow the generation and re-generation of a 30-character passcode that would bypass the need to connect a browser wallet to log in on mobile. This mobile login method allows gate creators (event managers) to log into their app on a mobile device in order to utilize the Scan interface and check in guests. A 30 character alphanumeric string with special characters is estimated to take centuries to crack. This requires a frontend library for generating random alphanumeric strings.

c. **My Collections UI:** A list of collection objects created by the user, click each title to drop down an informational pane. Each item includes an Edit button and a Scan button that would redirect the user to /Scan?collection=[collection_id]

d. **New Collection UI:** An input form that upon submission creates a new collection object for the user. Collection objects have attributes for Title, Description, ID (generated), and Start and End dates that affect the gate scanning process.

e. **Edit Collection UI:** An input form that allows users to edit collections they have created.

f. **Create Gate UI:** Modify the existing gate creation page to include a new domain whitelist input section. Add an input field for new attribute collection_id, not required.

g. **Edit Gate UI:** Modify the existing Edit Gate interface to include the updated gate object properties, domain whitelist and collection ID.

h. **Collection Scan UI:** Logged-in users browsing the Collections they created can click a Scan button to visit page Scan, with the collection ID added as a URL parameter. This interface has three containers - an info box for the Collection, a Scan button, and results of the prior scan. An "auto-scan" toggle button will scan a QR, render results, wait a given number of seconds, and reset the camera to scan another. Once the camera scans a QR code and fetches the getValidGates API, it receives the list of valid gates and grabs only the first valid entry, skipping the Gate Selection Modal found in the OAuth interstitial. It makes a call to get an authcode for that gate and is redirected to the callback URL (same page) with the auth code.

i. **Bulk Gate Upload UI:** Allows logged-in users to upload a CSV file representing token gates to create or edit in bulk, making it easier to create tickets for an event. Each gate represents one ticket, and is tagged to one Collection at a time. Collections represent events and have start / end dates for being active, and their own trigger domain that supercedes gate triggers. The UI provides a progress bar of server processing and a list of results for each line item. Users can export a CSV of items that succeeded, and output a CSV file of error messages relating to each line number in the sheet.

2. **Gate Resource Server**

    a. **API Route: Get Valid Gates:** The getValidGates route will be modified at the point of filtering by matching domain. If the gate does not belong to a collection, check for a match between the callback URL and the gate trigger URL. If a gate belongs to a collection, match the trigger domain of the collection to the callback URL instead and ignore the gate trigger domain. Integrate the new data subset for gates, domain_whitelist, and perform additional checks when filtering valid gates by the callback URL. When the given wallet is extrapolated to assets owned and iterated, at the time that a found asset matches an active gate, if that gate is tagged to a collection, ensure that the wallet in this API route matches any existing nft usage log for this collection. Include a new array with returned results, a rejection list of NFTs that matched gates but could not be added for reasons such as Passback Detected (same NFT, same event, different wallet). Include with the return package a new data object, collection_statistics, revealing total verifications for the event, total unique wallets for event, number of unique gates triggered in the collection, and number of gates in the collection not registered yet.

    b. **API Route: Make Gate Auth:** Refactor the makeGateAuth route with the restrictions introduced by collections. If a payload auth is requested for a gate belonging to a collection, the current date must be within the start and end attributes of the collection, and the calling URL must match the collection trigger URL. If the gate does not belong to a collection, the is_active and domain_trigger attributes of the gate apply. Update gate usage data, profile query counts, and usage counts by collection.

3. **Ticket App API**

    a. **API Route: Logins: Mobile Passcode:** A duplicate API route of the traditional wallet-based login process is refactored, replacing the wallet-based login section with a step to verify the login passcode among the user table. The API route input parameter for wallet is replaced by the login passcode.
    b. **API Route: Edit Profile:** Modify the API route for editing user profiles to include adding or editing the separate Mobile Passcodes table when a user creates or edits his optional mobile passcode.
    c. **API Route Update: Create Gate:** Modify the existing Create Gate route to store the new format of the gate object with new attributes - domain_whitelist, collection_id
    d. **API Route Update: Edit Gate:** Refactor the Create Gate route to incorporate the new gate data structure.
    e. **API Route: Edit Collection:** Edit a collection object from an authorized user.
    f. **API Route: Delete Collection:** Delete a collection object from an authorized user
    g. **API Route: Create Collection:** New API route to store collections from authorized users. The trigger URL of a collection is the collection ID appended to the /Scan page, and supercedes trigger domains for gates. Start and End dates set a window by which scanning token gates for this collection is allowed.
    h. **API Route: Bulk Gate Creation:** Receive and process CSV data from the web app when users bulk-upload gates to create or edit. Provides a results output for each line item or failure message for improper syntax. It will overwrite existing gates if the gate ID is mentioned in the sheet row for bulk editing.
4. **On-Page Script**
    a. **Modified Start Auth - QR Camera:** The modified OPS (on-page script) will live on the Scan page on the app as opposed to third-party sites. This version will replace the OAuth wallet connection process with a camera QR code scanning library that scans consumer receive addresses from any mobile wallet.
    b. **Modified Start Auth - Gate Processing:** Once the QR scanner obtains a wallet address, call the getValidGates API route but do not print a grid of result options. Instead, automatically use the first gate object to request a payload auth code from the gate API, then redirect to the current domain with the auth code parameter appended. The Scan script fetches the payload from the auth code and produces a QR code from the gate payload, which is the ticket identifier. In addition to rendering collection usage information and prior scan results, render results of an nft rejection list provided by the gate API - NFTs found in the wallet that matched gates but did not belong to the originally registered wallet - AKA, "passback attempts". Ensure the logged-in user is allowed to scan this collection as the owner. The consumer / ticket holder cannot scan his own ticket for the QR, because the owner of the related collection must be logged in during the API call. If a gate has no collection at all, consumers can use the Scan page to obtain their own payloads, which is useful for video-type gates.
5. **MongoDB Database Sets**
    a. **Database Initialization and Access:** Initialize a MongoDB Atlas account for the various API components to share stored information in NoSQL articles pertaining to the web app - Users, Gates, Collections, Wallets, Mobile Passcodes and configuration references.
    b. **Gate API Database Calls:** Ensure that the resource server (token gating OAuth) can read and write to the MongoDB Atlas account using an API key with proper scope.
    c. **Web API Database Calls:** Ensure that the web app API server can read and write from web app related tables using an API key with proper scope.
6. **Load Testing Period**

a. **Beta Test Parties:** The system must be stress-tested in a real in-person environment. Find local communities that are willing to attend NFT-ticket based events with the understanding that the technology is in beta, troubleshooting scalability and security issues as they arise.

Performance Metrics

While our functional milestones represent the definition of complete for this project, key performance indicators measure the success of each milestone against arbitrary targets.

Our core KPIs for this app after launch are listed below. Rate of growth is unknown, so success is measured as a percentage increase over the prior analytic sample.

The metrics we plan to measure post-launch are:

- Number of business users creating NFT ticket gates
- Number of active collections (events) in a 3 month rolling average
- Number of event check-ins in total
- Average RSVP rate - number of NFT tickets VS unique wallet check-ins

We hope the beneficial effects to the ecosystem would be:

- Many more NFT projects hosting parties and events without help from developers.
- Large Cardano conventions, parties and other events will use NFTs as tickets for the first time.
- A powerful onboarding tool for Cardano; attract web2 event management teams with the added security and dynamic abilities of an NFT-abstracted ticket solution they can use with their existing system. Big events using the app means more new consumers using Cardano to purchase NFT tickets.

Risk Factors

We have carefully assessed the risks to completion, from project scope creep to external unforeseen events, and we have a high level of confidence that we can complete all milestones on time. Below are a few of the primary concerns our team will be continuing to monitor.

The price of ADA could decrease significantly and if the market value of ADA is below 0.25 cents at any point in the next 12 months, the project could potentially be at risk of losing value it presumed it would have. However, we feel that 0.25 is a moderately low conservative guess for ADA moving forward this year.

The OAuth flow is a complex security mechanism for providing access to private resources, so the project has allotted a budget for an Oauth 2.0 security contractor and an API architecture consultant to help ensure that the app not only works, but can work safely as expected at a large scale.

Deliverability

The team involved in this project has already created Cardano apps in prior funds and are looking to expand on what we have learned along the same theme of token-gating as a service. The app described in this proposal is already online and working, and is missing only the components listed in this proposal to be a full-fledged application for the in-person event space.

We have already set up our UML diagramming, shared drives, and project management through ClickUp. Top-level work-sprints were made to assess the time and cost of this project, and once funded will be broken down into smaller tasks for assignment.

Sharing Results

We will share the progress and results of this work product on our website as blog updates as well as in partnership with project partners on Twitter and Youtube. We want to assist Cardano organizations to utilize this service layer to token gate all manner of existing events in Cardano and beyond, ideally through the use of existing popular and commonly used NFT series.

Budget Analysis

Here is a breakdown of our projected hard costs and labor, organized by expense category. For a full list of all planned purchases and expected expenses please review the proposal worksheet, tab name Project Costs.

- Project Team
  - Dane, Full Stack Developer (proposal author)
    - $14,882 USD for 994 work hours
    - Amounts to approximately $15 per hour
- Hardware
  - We do not need to purchase any physical equipment.
- Marketing
  - This proposal does not take into account paid methods of advertising this app. A portion of post-launch revenue will be diverted to a marketing budget and the app team will continue to engage with the community online and in-person for onboarding.
- Technical Contractors
  - Mobile Web App Consultant
    - $5,400 USD for 90 work hours
- Service Providers
  - Netlify, 12 Months of Pro Tier, Web App, $228
    - Used throughout development and testing.

    - Netlify, 12 Months of Enterprise, Oauth App, $1,800
        - For app performance post-launch, with 99% uptime, faster machines and a high-performance edge network.
    - Render, 12 Months of Team Plan, $288
        - 500GB of bandwidth for hosting web app and oauth app.
    - Render, 12 Months Standard Tier Node Service, Web API, $300
        - One year of standard hosting for the self-service web app
    - Render, 12 Months Standard Tier Node Service, OAuth API, $300
        - One year of standard hosting for the ticket resource (OAuth) server used in this project to demonstrate the usage of the 2FA server to be built.
    - MongoDB Atlas Dedicated Cluster, GCP Tier M30, 12 Months, $4,665
        - Data storage at a service tier fit for development and first-year public use.

For a matrix of which cost will be needed at which phase in development, please refer to the worksheet: https://docs.google.com/spreadsheets/d/1lKg-6eqtI8RCOBlrCeyKp61PoC3EhwoyhGo3Xf5SsBI/edit?usp=drive_link

Value for Money

The cost of this proposal is a one-time price that will produce a self-service Cardano app that can be used by the entire community to create their own events for their CNFT communities, without needing to code, and without paying fees to a centralized entity - a simple, self-service solution for in-person NFT verification for events.

Once online, Cardano community can begin reaching out to web2 event companies with a new reason to use Cardano for more secure and more engaging events.

All team members have committed to a baseline cost of living for the duration of the project. All hard costs have been plotted out, and the hourly rate for labor amounts to approximately $15 per hour. We do not plan to outsource development work, and this proposal will allow the team to focus on the project around the clock until it is completed.

Project Team

For the duration of the project, all project members will commit to living at a bare-minimum cost plus 20%, which is stored in the Proposal Workbook per team member. This amount factors into the proposal cost, as milestone timelines are based on work sprints which have been assessed for work-hours. This buffer amount is for safety to guard the continuity of work, not for extracurricular expenses.

Dane McBride

Fullstack developer and project manager

Dane creates operational business strategies from competitive and internal analyses of e-commerce brands. As a Plutus Pioneer student he devotes his time to learning smart contracts for use in B2B services to grow utility and tooling on Cardano.

Relevant Experience: In over 14 years of experience in competitive consumer industries, Dane was in charge of strategy development and execution, managing large-scale technical projects with teams of up to thirty-five people. Dane is bringing a mix of Cardano blockchain experience, self-taught coding skills and digital marketing background to the project. Dane was in the Plutus Pioneer Program to lead Haskell development for Cardano projects.

Benefits to Cardano

For the first time, Cardano conventions will be able to use NFTs as tickets without changing much about their existing operations, while saving money on printing and shipping paper tickets.

This is a highly dynamic system that would allow NFTs representing tickets to enjoy multiple utility perks at the same time. It would allow future events (collection objects) to re-use the same tickets for access, making ticket NFTs much more engaging long-term.

This system is not a proprietary ticketing system, it is a service layer that helps abstract NFTs into tickets and verify them. Once built, Cardano projects will be able to create their own NFT based event tickets and keep all the profit. Even teams that have an NFT collection but have never used ticketing software, don't need to use any - by using this scanning app by itself, you can confirm admission by scanning the consumer's phone and receiving the ticket payload - no need to do anything further if all you need to do is confirm a person owns the right NFT, which would be ideal for afterparty use.

There is no new mobile wallet to develop, no mobile app to install, consumers who are familiar with practically any mobile wallet can use it, and event teams familiar with practically any check-in software can use the app’s back-end to design this NFT utility without needing any coding skills or advanced knowledge.

NFT tickets can be used for multiple events throughout the year, with unlimited re-entry for the holder per event, while at the same time eliminating the possibility of “passbacks” or ticket duplication.