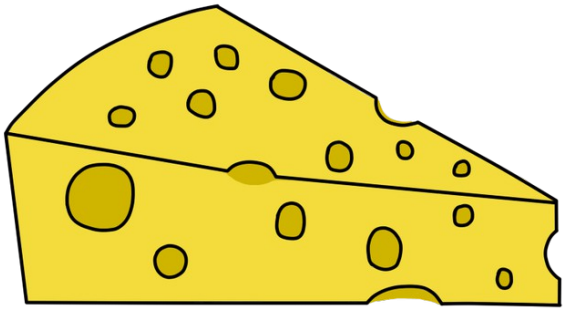
Cheeze of Insight



"I'll so offend to make offense a skill,
Redeeming the time when Cheeze Wizards think
least I will." — Shakespearean Wizard

TECHNICAL PROPOSAL: 1.0
Published: 8/14/2019
Last Updated:8/15/2019

1 ABSTRACT

As a team built from players of Satoshi's Treasure we're uniquely aware of the role statistics can have on gameplay and player psychology. Enter *Cheese of Insight*. The *Cheeze of Insight* DApp is a web application and browser extension designed to enrich player psychology in *Cheeze Wizards* gameplay by providing methods for realistic and test battle predictions, and a module for doing deep analytics of battle history. More specifically, *The Cheese of Insight* DApp approaches player psychology using a four-pronged method that includes: an **Analytics** component, a **Smart Contracts** component, a **Battle Predictions** component, and a **Prediction Market** component.

- **Analytics Component:**

  1) The Analytics Component and its UI, provides players of Cheeze Wizards with a frontend for analyzing individual Wizards and tournament battle history.

- **Smart Contracts Component:**

  1) The **Smart Contracts** component adds value to the *Cheeze Wizards* platform by providing **Battle Predictions** and a **Prediction Market** with a model for making hypothetical guesses about outcomes of realistic or hypothetical Duels.
  2) This is possible because our modified Wizards contract can "re-mint" specific Mainnet Wizards by importing them into our own contract which then allows players to battle replicated versions of their Mainnet Wizards in a testnet environement against replicated versions of other Mainnet Wizards of their choosing.

- **Battle Predictions Component:**

  1) The **Battle Predictions** component is included in both **Offline** and **Online** component layers.

  2) In the **Offline Layer**, players can play their Wizards in test matches against replicated versions of Mainnet Wizards.

  3) When playing a test match, the player invoking the Duel chooses whether they wish to dictate their opponent's turn commitments, or instead by a super smart (and super dumb) AI player. The AI player we've created, analyzes a Wizard's match history, and attempts to find patterns. Like a fine cheese, our AI matures with age, so our AI's predictions become more trustworthy as the tournament progresses.

  4) In the **Online Layer**, analytics collected from tournament battle history is used by our AI to create predictions about outcomes of Duels which are pending outcomes but not yet resolved.

5) When the tournament enters Phase III, the live Duel predictions created by our AI will be used in to create LONG and SHORT weightings for the **Prediction Market** component.

6) Since the reliability of our AI's match predictions is in many ways proportional to the data available from a tournament's match history, we can say our AI begins a tournament "dumb" and ends the tournament "smart".

- **Prediction Market Component:**

1) Decentralized prediction markets can be used to generate sets of tokens that represent a financial stake in the outcomes of any event.

2) By combining the *0x Protocol* with an *Augur* prediction market and open source code from *Veil*, once phase Phase III of the tournament has been reached our platform will automatically launch a prediction market on *Augur* that emits LONG and SHORT predictions for the outcome of every pending Duel.

3) Using *0x*, the order book for the prediction market is kept offline to limit gas transactions and both LONG and SHORT tokens are merged into a single order book.

4) When Blue Mold enters the tournament and a Wizards ability to compete is at stake, the prediction market launched on *Augur* allows anyone following the tournament to place either LONG or SHORT bets on the outcome of a pending Duel.

5) If a Wizard has challenged another Wizard but the Duel is not resolved during the dueling window for that day, any bets related to the invalid Duel are automatically returned to the orignal better.

6) If a Duel takes in bets, and does resolve during the dueling window, betters who guessed correctly are rewarded in ETH. Betters who guessed incorrectly will lose some, or all, of the ETH placed as their bet. In the unlikely event that the Duel ends in a tie (e.g. 2 neutral Wizards committing an equivalent set of 5 Turns) tied match outcomes will be processed the same as unresolved Duels, wherein all betters receive a full refund of their original bet.

7) As the Blue Mold begins to show its effects, this **Prediction Market** will become a place where players who no longer have Wizards in the tournament ("rind-in-the-game") can continue to follow and participate in the game.

"Like a fine cheese, our AI matures with age"

2 TERMINOLOGY

Layers:

The terms "**Online**" and "**Offline**" layer" refer to Duel availablity. Components which rely on the Dueling window are part of the "Online" or "Real-Time" tournament integration. However, data and analtyics tools which are usefuly outside of the Dueling window (example: test duels using replicated Wizards), are considered to be part of the "offline" layer.

Why Have an Offline Layer?

Creating an environment for participation outside of the dueling window promotes a player culture where the tournmament remains interactive outside of duels. It also gives summoners the ability to prepare and train their Wizards for future battles.

Betting:

1. For the purposes of the **Prediction Market** component, “bets” can thought of as either **LONG** or **SHORT**, and regarding a prediction that some Wizard will prevail in

a pending Duel against another Wizard who loses. *Augur* prediction markets have two outcome shares and these are unique ERC-20 tokens representing a distinct answer to a market's question which we can call "LONG" and "SHORT" tokens. LONG and SHORT tokens must be transacted as complete sets that equate to 1 ETH. Example set: 0.2 ETH LONG, 0.8 SHORT).

Given the above, we can take the terms LONG and SHORT to mean as follows.

LONG BETS:

"Prevail" or expecting an increase in the underlying asset (e.g. you're betting a particular Wizard will prevail in their pending Duel)

SHORT BETS:

"Lose" or expecting a decrease in the underlying asset (e.g. you're betting a particular Wizard will lose in their pending Duel)

COMPLETE SETS:

A perfect factor of 1 ETH transacted as a pair. Another way to think about this transaction is it is like a sliding scale between SHORT and LONG that always balances to 1 ETH.

ORDER BOOK:

The "order book" can be thought of as the total transactions of a prediction market of a given Duel. Since Augur enables you buy and sell complete sets of shares for ETH. By buying and selling complete sets on behalf of the betting user, the *Cheeze of Insight* smart contract (taken from Veil https://github.com/veilco/veil-market-creation) will merge orders for LONG and SHORT tokens into a single order book. Using the *0x Protocol*, our contract executes trades on behalf of our user base off-chain, which dramatically reduces the overall gas required to power the prediction market. The on-chain transactions that will need to occur are:

1. Initial placement of any LONG or SHORT bet

2. Final settlement of the order book after a particular Duel has resolved and the market is expired

3 DEVELOPMENT WORKLIST

Methodolgy:

The suggested workflow (seen below) is designed to optimize modularity and create a development environment where each component can be worked on individually. There should be a limited amount of interdependence between the various components during the POC stages of development. Developers can work on a component autonomously or in combination with the team.

1. **Analytics Component:**
    - POC Application - ALMOST DONE – Missing Requirements:
        - Connect current POC to Dapper to get “My Wizard” from user's actual wallet. This feature will also ultimately act as a login for the DApp.
    - Stage 2: convert to browser extension
    - Stage 3: implement design / branding

2. **Smart Contracts Component**
    - POC – NOT STARTED. Requirements:
        - Re-launch existing tournament and minting contracts with shortened Phases and successfully complete a tournament with 2 Wizards
        - Modify existing contract code to allow for importing of any Mainnet Wizard so that test Duels can be executed by the Offline Layer of the DApp.
    - Stage 2: Launch 3 simultaneous versions of the contract (either multiple contract addresses, or multiple instances within the same contract address) with a contract instance for each Phase of the tournament. This will be used in our submission to demo how the DApp behaves during each of the 3 Phases of a tournament.

3. **Battle Predictions Component**
    - AI POC – NOT STARTED. Requirements:
        - In normal Duels (e.g. conducted under conditions where neither Wizard is Ascending) having a higher power level than the opponent does provide any inherent advantage regarding win / loss of the match. So we can say that Duels are like a glorified game of Rock, Paper, Scissors with the addition of Affinity / Vulnerability stats, and 4 turn play possibilities instead of 3. The first step in creating our AI will be bootstrap the source code and strategies of this

open AI project (https://svilentodorov.xyz/blog/rps) and get it running.

- Once the base AI source code is confirmed to be running correctly, we'll need to modify the source code to account for Affinity / Vulnerability damage multipliers, and adapt it to the Cheeze Wizards model of: Water > Fire > Wind > Water. At this stage testing can still be done using a fork of the Rock, Paper, Scissors demo page (https://svilentodorov.xyz/rps).
- Stage 2: Connect the created AI to the smart contract and complete a test Duel. Presumably, at this point, the contract already has the ability to import Wizards from Mainnet. Note: Interfacing the AI with the smart contract is done at the DApp level (web development) so it doesn't knowledge of Solidity is not required to complete this task.

4. **Prediction Market Component**
   - POC – NOT STARTED. Requirements:
     - Bootstrap this source code from Veil (https://github.com/veilco/veil-market-creation) and test creating a prediction market, placing bets and settling the order book after the market deadline expires
     - Test the above workflow again based on an actual pending Duel from our smart contract
     - Ensure that each prediction market that the Veil DApp is launching takes a Duel ID as an argument so that we can avoid creating multiple prediction markets for the same pending Duel.
     - Stage 2: Connect the working (manually released) Duel prediction market model with an Oracle that will automatically creates a prediction market (can be as simple as a cron job that runs every minute to update the list of pending duels, or can be done as a server webhook to the smart contract emitted event). Note: the emitted event that signifies a pending Duel can be found at line 1601 of BasicTournament.sol:

```
emit DuelStart(duelId, wizardId1, wizardId2, duelTimeout,
isAscensionBattle);
```

Note: Since creating a **Prediction Market** based on the emitted contract event potentially creates the market before the pending Duel transaction is confirmed, it's important that the **Prediction Market** doesn't allow for taking in bets until the `_beginDuel()` tx is confirmed. Handling this scenario should be equivalent to handling an invalid or unresolved challenge.

- Alternatively, if the Oracle method is proving difficult or time consuming, an easier strategy is bootstrap the form elements from the Veil frontend in made in the POC. This allows users to create their own prediction market based on drop down list of pending Duels. Once a prediction market has been created for a particular Duel ID, we remove it from the drop down list so not to be confusing as the Duel will now appear in the “Live Duel Markets” area of the DApp.

4 DELIVERY SCHEDULE

- SUNDAY, AUGUST 18:
  - Project management documentation (high level overview, delivery schedule, workflow, and tasks distributed to devs)

- SUNDAY, AUGUST 25:
  - POC's completed for **Battle Predictions**, **Smart Contracts**, and **Prediction Market** components.
  - Basic browser extension created and Analytics POC integrated

- SUNDAY, SEPTEMBER 1 (CONTEST SUBMISSION DEADLINE):
  - Branding and design finalized and integrated
  - POC's integrated and implemented into final DApp
  - Final DApp tested
  - Project submission package created (e.g. all product documentation text finalized)

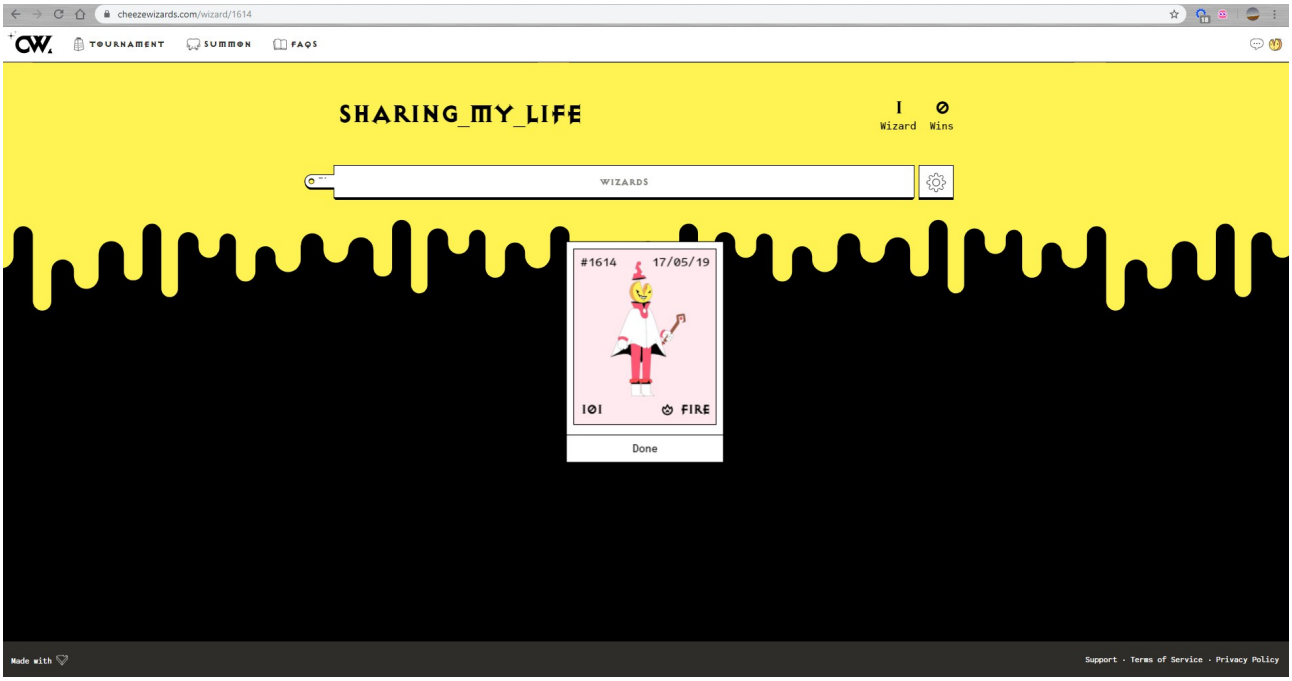
5 BRANDING GUIDELINES

Official *Cheeze Wizards* branding and lore guidelines for 3rd party developers:

https://docs.google.com/presentation/d/1bK8yWTjkhNlM1w0-hfRhaStCeBxY8yEHI3fZGOP0EXo/edit#slide=id.g5e2d0fd546_1_0

Official *Cheeze Wizards* developer design assets:

https://drive.google.com/drive/folders/1yZTs7epElKYVaDLMrpC6wRE5LdVZXbXT


cheezewizards.com/wizard/1614
TOURNAMENT
SUMMON
FAQS
SHARING_MY_LIFE
1
Wizard
0
Wins
WIZARDS
#1614
17/05/19
101
FIRE
Done
Made with
Support · Terms of Service · Privacy Policy