# TRUESHOE: A BLOCKCHAIN ENABLED MARKETPLACE FOR AUTHENTICATED SNEAKERS

WHITEPAPER

**Benjamin Kha** **Nikhil Sharma** **Dapree Doyle** **Ryohei Kobayashi**

## ABSTRACT

The co-founders of TrueShoe imagine a world where sneaker marketplaces riddled with counterfeits are a thing of the past. We envision a service where users can purchase trusted sneakers at a fraction of their current costs. *Business Insider* estimates that the resale sneaker market is worth over $1 billion[1]. Our analysis suggests that the sneaker reselling market is vulnerable within a sneaker industry where fraud is prevalent. After assessing the weaknesses of sneaker marketplaces, our team concluded that a solution lies using Blockchain technology. Our blockchain enabled marketplace allows users to conveniently authenticate sneakers at a much lower cost than current solutions. This is achieved by partnering with shoe manufacturers to burn in a unique identifier on their sneakers, for which they will receive a commission on all transactions involving their sneakers on TrueShoe. Our vision extends beyond just sneakers, and we would like to expand into other industries like luxury handbags and watches as well because there exists a similar need to prevent fakes. However, we decided to focus on shoes in the near term because they represent one of the most counterfeited items on the second-hand market.

## 1 Executive Summary

The sneaker resale market is currently valued at $1 billion in the U.S. alone. Limited-issue sneakers are released by sneaker manufacturers, such as Nike and Adidas, and then retailers randomly select buyers who purchase the shoes at the retail price. Soon after that, some of these buyers start trying to sell their shoes at a significant markup from the retail price. Initially, eBay was the primary reselling market for sneakers. However, as questions about the authenticity of shoes arose, consignment stores and niche apps and websites became dominant reselling markets. Even though the market is expanding, both buyers and sellers face many issues. In many cases, the sneaker reselling market is both expensive and untrustworthy. By leveraging blockchain technology, TrueShoe only allows sellers that have authentic sneakers to create listings on our platform. TrueShoe is a market in which buyers can get authenticated sneakers without paying the high costs of manual verification. TrueShoe will also help sneaker manufacturers to reduce the amount of their fraudulent sneakers, while also providing them with a commission on their shoes when they are sold in the second-hand market.

## 2 The Problem and the Marketplace

The sneaker reselling industry is a lucrative business, and there are many people who try to sell counterfeits. Currently, most sneaker reselling markets have authentication processes to ensure shoes are not counterfeits. However, many issues continue to persist for both buyers and sellers. First, these verification services often costs a lot of money, since they have to hire authenticators. For instance, an online reselling platform stockX takes a 9% selling fee for each sale. This indicates that as the resale price goes up, the seller has to pay more. Secondly, it takes more time for buyers to receive their sneakers when buying in the reselling market because of the delay of that the authentication process produces. Specifically, each pair of shoes has to be shipped twice, so it takes longer for buyers to get them. Finally, the quality of fraudulent sneakers has been improving on a daily basis, making it even harder to spot fakes. Even with an

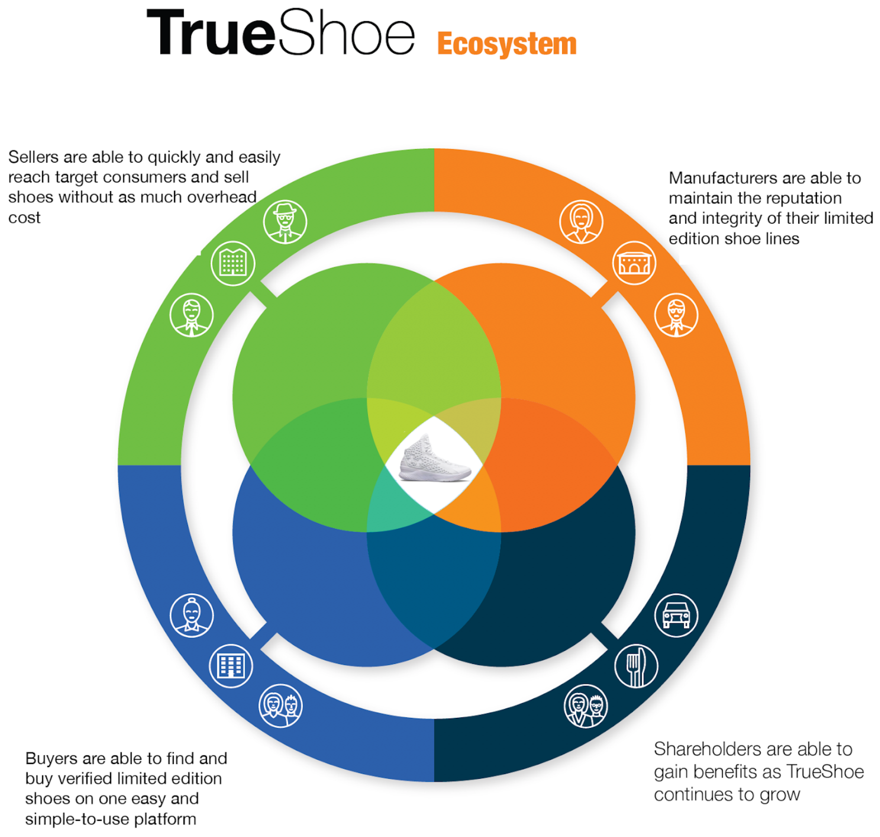
[1] https://www.businessinsider.com/inside-the-billion-dollar-sneakerhead-resale-economy-2016-5

authentication process, unfortunately, it is not 100% ensured that the sneakers are authentic. Some platforms let buyers return when they get the fake pairs, but many consumers are not capable of spotting the differences between genuine and counterfeit sneakers.

## 3 The TrueShoe Solution

Using the power of blockchain technology and its smart contract-based payment platform, TrueShoe will enable sneaker buyers and sellers to interact directly in a transparent way that all parties can trust, without the involvement of verification services. Our goal is to make quality, authentic sneakers more accessible and affordable to a broader audience. TrueShoe and it’s secure, blockchain-based platform allows buyers, sellers, and manufacturers to benefit in three main ways:

1. **Lower Service Fees**: We provide a more convenient way to authenticate via the blockchain at a much lower cost. Current competitors charge around 10% to authenticate shoes, while we plan to charge 5%.
2. **Less Fakes and Commission**: Our service reduces the number of fakes and protects the brands of shoe companies. We also pay a 1% transaction fee to shoe manufacturers for all transactions involving their sneakers.
3. **Higher Resale Value**: Shoes have a higher resale value because they have been verified.



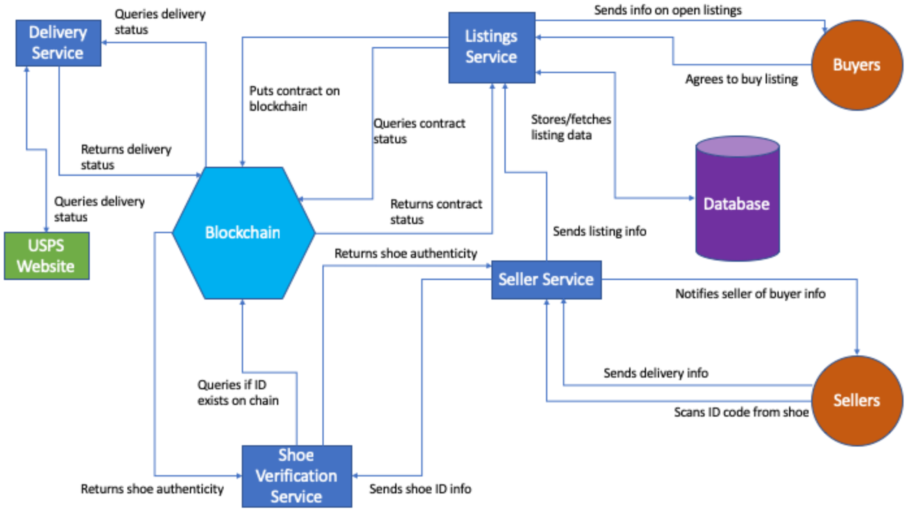
Sellers are able to quickly and easily reach target consumers and sell shoes without as much overhead cost

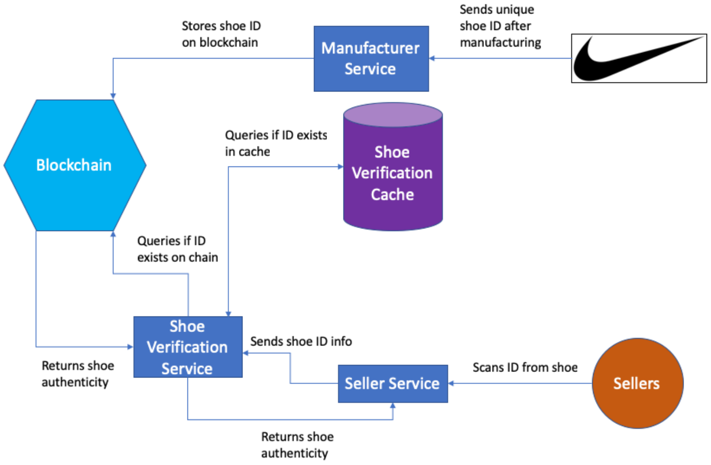
Manufacturers are able to maintain the reputation and integrity of their limited edition shoe lines

Buyers are able to find and buy verified limited edition shoes on one easy and simple-to-use platform

Shareholders are able to gain benefits as TrueShoe continues to grow

# 4 System Architecture Overview




Stores shoe ID on blockchain
Manufacturer Service
Sends unique shoe ID after manufacturing
Blockchain
Queries if ID exists in cache
Shoe Verification Cache
Queries if ID exists on chain
Shoe Verification Service
Sends shoe ID info
Seller Service
Scans ID from shoe
Sellers
Returns shoe authenticity
Returns shoe authenticity

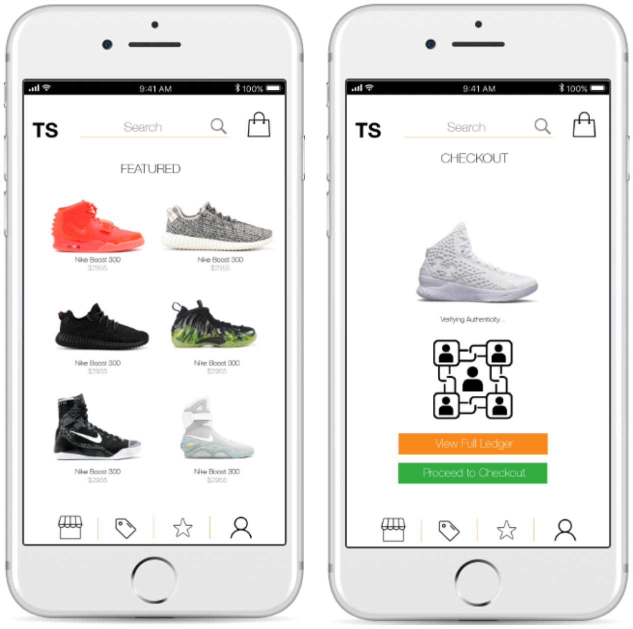
Our system architecture can be broken up into 5 main services:

1. The shoe verification service
2. The seller service
3. The listings service
4. The manufacturer service
5. The delivery service

Sellers interact through our frontend application with the seller service, while buyers are shown open listings and agree to buy listings through the listings service. When a seller posts a listing on TrueShoe, he or she scans the burned in identifier, which triggers the shoe verification service to verify that the sneaker is indeed genuine. How this works is, as shoe manufacturers create shoes, they upload the shoe IDs to our platform, which we store on the blockchain by creating transactions with batches of IDs in the transaction input data. To verify if a shoe is genuine, we check if the ID exists in one of these transactions, and we can cache these queries as an optimization. After a buyer agrees to buy a certain listing, the seller will be notified and be asked to provide the delivery information for his or her shipment. Then, the listings service will send this information to a smart contract on the blockchain, to which the buyer must send funds to. The smart contract will periodically query our delivery oracle, which will then query the USPS (or the appropriate delivery service) website for the delivery status. Once the delivery status is returned as being “delivered”, the smart contract will initiate the transfer of funds to the seller's ether address. The implementation of our platform can be found on Github[2].

## 5 Frontend and Database Store

A mockup of what the TrueShoe user interface will look like can be seen here, and there is a full interactive demo online[3].



TrueShoe will be implemented as a mobile platform with support for both Android and iOS, which we hope to extend to include a web platform as well eventually. We will construct the frontend utilizing React for development, and use Google's provided integration with Firebase to seamlessly tie our user interface together with the necessary backend support.

The Firebase integration provides a realtime database[4] which allows for efficient querying and synchronization of sneaker listings across our user base. This design choice is important due to the time-critical nature of our product; our auction-based business model mandates low-latency synchronization across consumers for optimal customer satisfaction.

Firebase also provides support for other features that are relevant to our product. Specifically, its built-in chat framework[5] support will allows us to easily integrate a feature into our application which allows vendors and consumers to get in touch and discuss any concerns relating to the details of the smart contract for a given purchase (detailed more in the Smart Contracts section below). Additionally, Firebase provides both a native integration for OAuth2[6] allowing users to authenticate with ease and an integration with Google Analytics[7] that will allow us to track app usage effectively and update our business model accordingly.

## 6 Smart Contracts and Cloud Resources

Smart contracts will be utilized to automatically transfer funds from buyers to sellers after the delivery is confirmed.

We will be using smart contracts for two distinct purposes within our system architecture:

[2] https://github.com/sharmaster96/true-shoe

[3] https://marvelapp.com/3hcg2ae/screen/50209798

[4] https://firebase.google.com/products/realtime-database/

[5] https://firebase.google.com/docs/cloud-messaging/

[6] https://firebase.google.com/docs/auth/

[7] https://firebase.google.com/docs/analytics/

1. The primary technical contribution TrueShoe makes in the space of sneaker resale is using smart contracts to transfer payment upon verified delivery of authentic sneakers from vendor to consumer. As mentioned in the previous section, the `SneakerTransferContract` will rely upon a `DeliveryService` which serves as an oracle for triggering asset transfer upon successful transaction by scraping data from the delivery carrier website (e.g. USPS).

    (a) Notably, this system allows vendors to behave maliciously; simply by never delivering any purchased sneakers to their customers, they can lock away their customers assets within the smart contract. To address this problem, we will be implementing a timeout feature, which requires vendors to provide a delivery guarantee window after which the consumer is reimbursed their payment regardless of whether or not the sneakers are delivered after the timeout.

    (b) *Timeout policy*: the specifics around the timeout policy are business- critical; without a good policy customer dissatisfaction will be high. Hence to select the best policy we've decided some analytics are required to monitor app usage and customer satisfaction. Towards this end, we will be experimenting with the following three possibilities and checking which yields the best results: Set a default global timeout window (i.e. 1 month).

        i. Set a default global timeout window (i.e. 1 month)

        ii. Require vendors to transparently provide a timeout to the users, displayed along with a base item price when adding an item to sell in the TrueShoe marketplace.

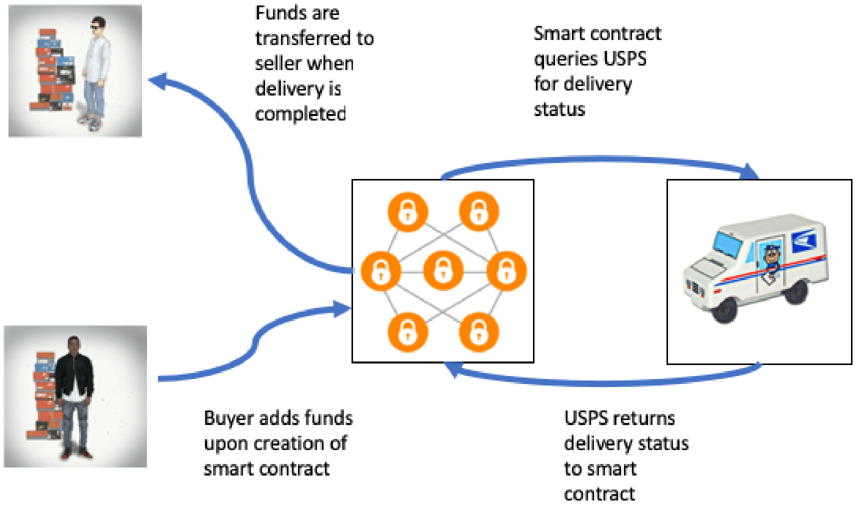
        iii. Allow customers and vendors to negotiate a timeout window pre- purchase, either as part of the auction itself or via our Firebase chat integration.

2. We will also use smart contracts to drive the sneaker resale economy via an auction-based system. Similar to eBay, consumers will be able to place bids for items directly through our interface, and these bids will be binding by depositing consumers bid amounts within the items corresponding `SneakerResaleAuctionContract`. Once the auction ends, its smart contract will reallocate funds to the vendor, the consumer, shoe manufacturers, and our company in accordance with our business model.

The blockchain platform we will utilize is the Oasis Labs Devnet[8], a new decentralized cloud platform that provides privacy-preserving cloud computing built on Ethereum. The Oasis Devnet is a very new technology which combines existing smart contract technology with secure hardware enclaves and differential privacy to provide a simple, fast, and secure interface for building dApps. The Oasis Devnet provides a `contract-kit` tool which is built on top of and integrates with MetaMask and the Truffle Suite, allowing for easy implementation and localized testing of smart contracts in Solidity. Such a platform was selected with the foresight that TrueShoe will ultimately expand to require more compute-intensive functionality (i.e. private decentralized machine learning for sneaker recommendation to our users).

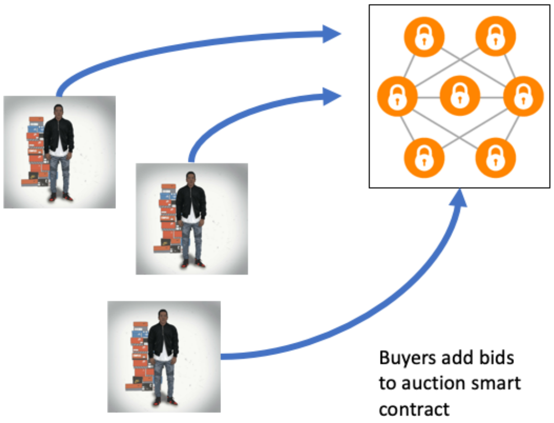
This entire process can be visualized diagrammatically below:

[8] http://docs.oasiscloud.io/en/latest/quickstart-guide/

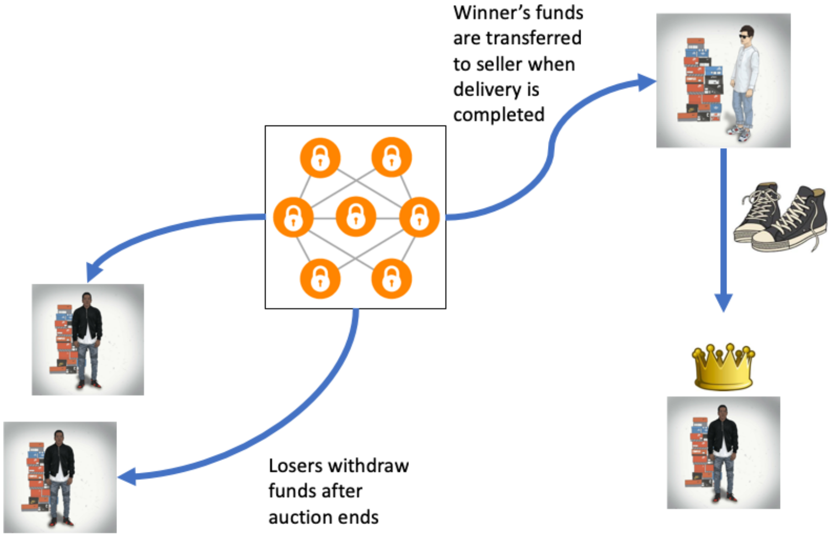

Funds are transferred to seller when delivery is completed
Smart contract queries USPS for delivery status
Buyer adds funds upon creation of smart contract
USPS returns delivery status to smart contract

## 6.1 Auctions

The TrueShoe smart contract stores state for every ongoing shoe auction. For a given auction, interested buyers add their bid to the smart contract, until the specified auction end date.



Then, everyone who didn't win the auction and was overbid can withdraw their funds. The contract continues to hold on to the winner's funds, periodically querying the delivery status of the shipment from the oracle USPS website through our `DeliveryService`. If delivery is confirmed before a prespecified timeout, the contract transfers the funds to the seller; otherwise the buyer is refunded. More details about the design of `DeliveryService` and its querying protocol are detailed in the following section describing TrueShoe's delivery oracle.


Winner's funds are transferred to seller when delivery is completed
Losers withdraw funds after auction ends

## 6.2 Delivery Oracle

TrueShoe requires an oracle for validating shoe delivery, and for this we include a `DeliveryService` in our design to query delivery tracking information on the delivery carrier website. The design choice to include `DeliveryService` as a component existing outside the products blockchain ecosystem was intentional, enabling a greater degree of adaptability to configuration/API changes made by USPS (or other carriers) without affecting the implementation of our `SneakerResaleAuctionContracts` for purchases.

We wanted to design a `DeliveryService` policy with the following two important properties:

1. The `DeliveryService` should stop querying after the vendor-specified timeout delivery period (plus an added grace period which we will select robustly using our data analytics).
2. The `DeliveryService` should attempt to pay vendors in the most timely manner possible.

As such, the local query policy for querying the delivery carrier website we have selected behaves according to the following policy:

$$\text{sample\_time}(i) = \alpha + 2^{i-1} \text{ (hours)} \qquad \text{for } i = 1, 2, \ldots$$

We stop querying when

$$\text{sample\_time}(i) > \text{timeout} + \text{grace\_period}$$

As you can observe, the smart contract starts polling for delivery at time $\alpha$ then continues to poll with exponential backoff until the timeout window (plus grace period) expires. This policy is enacted trying to achieve a balance between querying too frequently to be effective, and query too infrequently and delaying vendor payment. There exist a couple choices for $\alpha$:

1. We could set $\alpha$ to be some fixed time, such as 48 hours.
2. We could adjust $\alpha$ adaptively as we receive statistical data about delivery times. One possible option is to have $\alpha$ be a certain (tuneable) percentile of delivery times; for instance, the first quartile which we could adjust on-the-fly with streaming histogram algorithms[9].

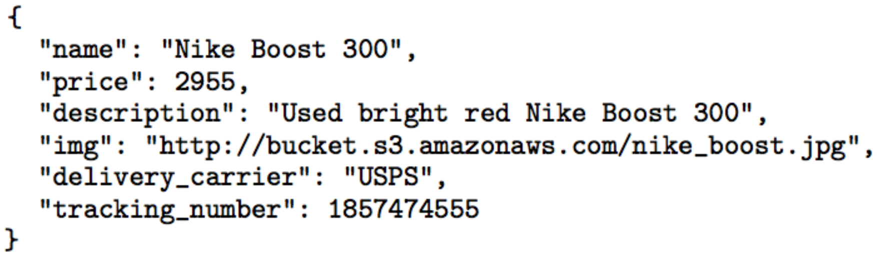
[9] https://www.cis.upenn.edu/ sudipto/mypapers/histjour.pdf

This policy is far from perfect, but we hope it is a reasonable starting point while our user base remains small. As it grows, we hope to eventually extend this local policy to a global querying policy that rate-limits queries across all active auction smart contracts; this will likely be based on query time series analysis and machine learning.

## 7 Backend/Data Format

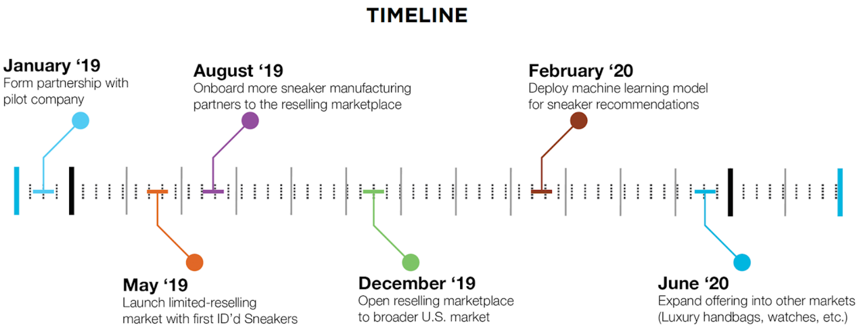
For our services, we plan to use Python, specifically leveraging frameworks such as Python Flask for the backend. To communicate between services, we plan to use to the JSON data format, specifically because it is well supported among all the different components of our product: frontend, backend, and blockchain. As an example of what this JSON data would look like, consider what happens as a seller is trying to create a listing.

After a shoe has been verified by the `ShoeVerificationService`, the seller will be asked to provide information that will be sent to `ListingsService`. The information requested will be include things like the name of the listing, the price, and also the delivery information, an example of which can be seen below:

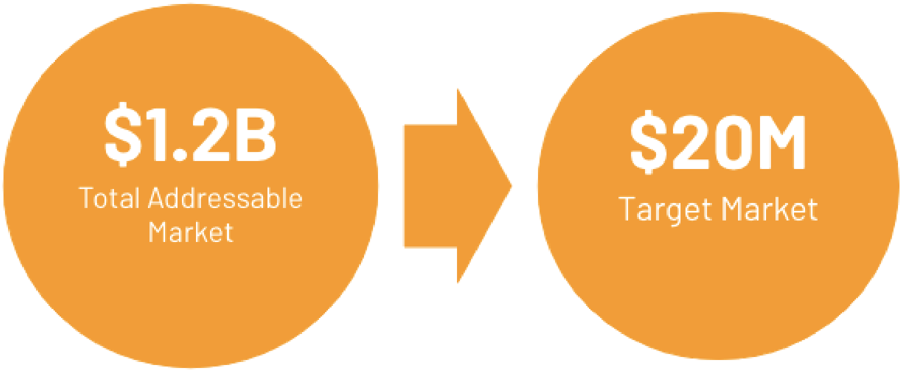


```
{
  "name": "Nike Boost 300",
  "price": 2955,
  "description": "Used bright red Nike Boost 300",
  "img": "http://bucket.s3.amazonaws.com/nike_boost.jpg",
  "delivery_carrier": "USPS",
  "tracking_number": 1857474555
}
```

# 8 Timeline


TIMELINE
January '19
Form partnership with pilot company
August '19
Onboard more sneaker manufacturing partners to the reselling marketplace
February '20
Deploy machine learning model for sneaker recommendations
May '19
Launch limited-reselling market with first ID'd Sneakers
December '19
Open reselling marketplace to broader U.S. market
June '20
Expand offering into other markets (Luxury handbags, watches, etc.)

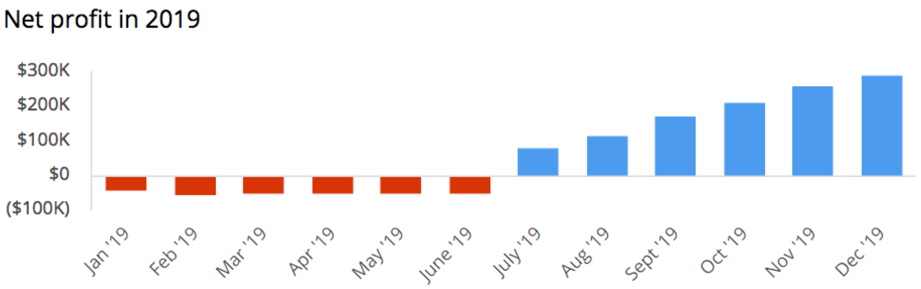
# 9 Business and Revenue Model



Our total addressable market is the $1.2 billion sneaker reselling industry, while we are currently focusing on the high end US sneaker market, the size of which is $20 million. Our revenue model is based on charging a 5% seller fee, and also by displaying sneaker ads on our platform.

# 10 Revenue Projections

Based on our revenue model and timeline described previously, here is our revenue projections for the next year. For a more detailed breakdown, please refer to section 13.



## 11 Meet the Team

**Benjamin Kha, CEO**

Benjamin is a M.S. student at UC Berkeley studying computer science. His research interests include probabilistic graphical models, deep learning, and natural language processing, and he is currently advised by Professor Stuart Russell. He has worked as a software engineer at Two Sigma and Checkbook, Inc., and is interested in new developments in blockchain and cryptocurrencies.

**Nikhil Sharma, CTO**

Nikhil is a M.S. student at UC Berkeley studying computer science. His research interests include machine learning and secure hardware, and he is currently advised by Professor Dawn Song. He has worked as a software engineer at Google and is currently working at Oasis Labs, where he is exploring how to implement a privacy-preserving blockchain network.

**Ryohei Kobayashi, CFO**

Ryohei is an exchange student at UC Berkeley from Tokyo. His major is economics and he has also learned business in his home university. His current interests include marketing, branding, and urban development. He has been collecting sneakers and he tries to improve TrueShoe experience from a customer's perspective.

**Dapre Doyle, COO**

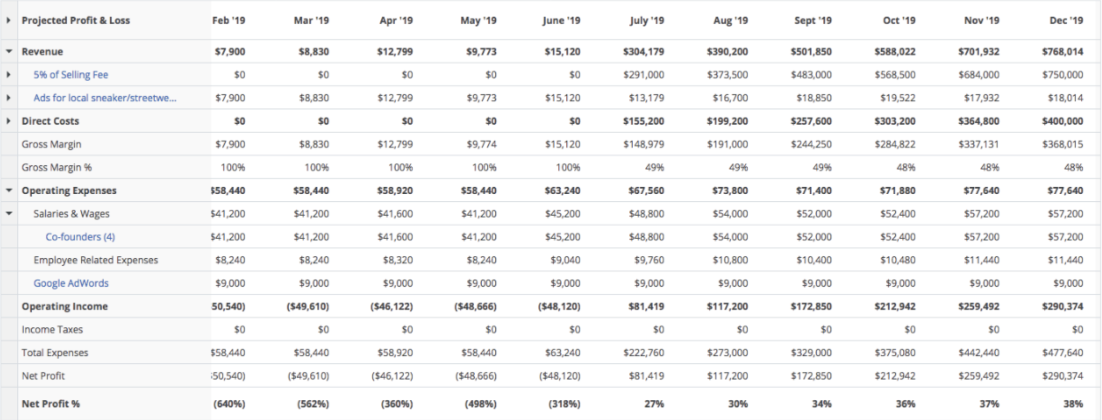
Dapree is a senior at UC Berkeley studying Business Administration. He has worked as the editor of the Daily Cal's Multimedia Department. His passions lay in the creative processes of video production and marketing.

## 12 Appendix A: Glossary of Terms

- **Blockchain** - A blockchain is a growing list of records, called blocks, which are linked using cryptography.
- **Ethereum** - A blockchain designed specifically to encourage development of smart contract Platforms, created by Vitalik Buterin.
- **stockX** - An existing service for sneakers, handbags, and other luxury goods where users have to manually send in items for authentication and pay a 9% commission fee.
- **Smart contract** - Decentralized applications that facilitate financial exchanges for services. Smart contracts are also known as self-executing contracts.
- **Oasis** - A privacy-first cloud computing platform on blockchain started by Professor Dawn Song of UC Berkeley.

## 13 Appendix B: Financials

Below is a detailed account of our projected financial data over the course of the next year as we develop TrueShoe:



| Projected Profit & Loss | Feb '19 | Mar '19 | Apr '19 | May '19 | June '19 | July '19 | Aug '19 | Sept '19 | Oct '19 | Nov '19 | Dec '19 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Revenue** | **$7,900** | **$8,830** | **$12,799** | **$9,773** | **$15,120** | **$304,179** | **$390,200** | **$501,850** | **$588,022** | **$701,932** | **$768,014** |
| 5% of Selling Fee | $0 | $0 | $0 | $0 | $0 | $291,000 | $373,500 | $483,000 | $568,500 | $684,000 | $750,000 |
| Ads for local sneaker/streetwe... | $7,900 | $8,830 | $12,799 | $9,773 | $15,120 | $13,179 | $16,700 | $18,850 | $19,522 | $17,932 | $18,014 |
| **Direct Costs** | **$0** | **$0** | **$0** | **$0** | **$0** | **$155,200** | **$199,200** | **$257,600** | **$303,200** | **$364,800** | **$400,000** |
| Gross Margin | $7,900 | $8,830 | $12,799 | $9,774 | $15,120 | $148,979 | $191,000 | $244,250 | $284,822 | $337,131 | $368,015 |
| Gross Margin % | 100% | 100% | 100% | 100% | 100% | 49% | 49% | 49% | 48% | 48% | 48% |
| **Operating Expenses** | **$58,440** | **$58,440** | **$58,920** | **$58,440** | **$63,240** | **$67,560** | **$73,800** | **$71,400** | **$71,880** | **$77,640** | **$77,640** |
| Salaries & Wages | $41,200 | $41,200 | $41,600 | $41,200 | $45,200 | $48,800 | $54,000 | $52,000 | $52,400 | $57,200 | $57,200 |
| Co-founders (4) | $41,200 | $41,200 | $41,600 | $41,200 | $45,200 | $48,800 | $54,000 | $52,000 | $52,400 | $57,200 | $57,200 |
| Employee Related Expenses | $8,240 | $8,240 | $8,320 | $8,240 | $9,040 | $9,760 | $10,800 | $10,400 | $10,480 | $11,440 | $11,440 |
| Google AdWords | $9,000 | $9,000 | $9,000 | $9,000 | $9,000 | $9,000 | $9,000 | $9,000 | $9,000 | $9,000 | $9,000 |
| **Operating Income** | **50,540)** | **($49,610)** | **($46,122)** | **($48,666)** | **($48,120)** | **$81,419** | **$117,200** | **$172,850** | **$212,942** | **$259,492** | **$290,374** |
| Income Taxes | $0 | $0 | $0 | $0 | $0 | $0 | $0 | $0 | $0 | $0 | $0 |
| Total Expenses | $58,440 | $58,440 | $58,920 | $58,440 | $63,240 | $222,760 | $273,000 | $329,000 | $375,080 | $442,440 | $477,640 |
| Net Profit | 50,540) | ($49,610) | ($46,122) | ($48,666) | ($48,120) | $81,419 | $117,200 | $172,850 | $212,942 | $259,492 | $290,374 |
| **Net Profit %** | **(640%)** | **(562%)** | **(360%)** | **(498%)** | **(318%)** | **27%** | **30%** | **34%** | **36%** | **37%** | **38%** |