



# Datacoins valued in data size units

A simplified valuation model where 1 MB of data = 1 datacoin.



July 2019

# Why a simplified valuation model is required?

- Not all data are created equal
  - Storecoin Platform will host **different types** of tokenized **apps** and each of them will create **data** of **different categories** (structure or *what* the data is) and **classes** (value of the data).
  - Different **data valuation methodologies** may be employed to **value the data** created by different apps. Even within the same app, different classes of data may exist.
  - Data may be discovered and accessed in myriad of ways — same data shared among multiple buyers (market analysis, research data, etc.), exclusive access to the data, subscription to existing and future data, and so on. This complicates pricing.
- But, this complexity need not be exposed to miners, app developers, and data buyers
  - Different data access patterns described above make **valuing the data hard**.
  - Data buyers, miners, and app developers will all deal with **datacoins**. If the pricing model differs from app to app, it makes it harder to crawl, query, and access data.
  - So, an **app agnostic valuation model** is desired, so a **uniform approach** can be employed to price data.

# Data categories and classes

- Data **category** facilitates data **discovery**
  - Examples: GPS data, PII, weather forecasting, crime data, drug research data, etc.
  - Multiple apps may create data for the same category. When buyers discover this category, they may purchase and access data created from multiple apps.
  - Data categories may influence data valuation methodologies used to value the data.
  - Storecoin Platform provides standard APIs to discover available categories across all apps. Data buyers use these APIs to **discover data**.
- Data **class** decides data **price**
  - Data **class** can be imagined as a **firehose**. Each app may have different set of data classes to control how the data is accessed by data buyers.
  - Each class of data is typically priced differently to simplify how data is queried and accessed.
  - Examples: An app may expose its data in 4 different classes —
    - *bronze* — 1MB of data = 2 datacoins. Premium for certain types of data.
    - *silver* — 1MB of data = 5 datacoins. More specialized data or access pattern such as subscription.
    - *gold* — 1MB of data = 10 datacoins.Note that these prices are for **exclusive access to the specific class of data**.

# 1 MB of data = 1 datacoin, irrespective of *classes* of data

- Users are familiar with this model
  - Wireless carriers, online storage providers, and others use similar model to price their services, so users are accustomed to this model.
- Hides complexities of underlying data valuation methodologies
  - Data produced by one app, App-A, may be more valuable than the data produced by another app, App-B, but they both expose same pricing interface to data buyers. They **both sell their data at 1 MB = 1 datacoin**. This means App-A's datacoin is more valuable (say, 1 datacoin = 2 STORE) than App-B's datacoin (say, 1 datacoin = 0.1 STORE).
  - Irrespective of the data valuation methodologies employed, the apps will eventually price their data for 1 MB of data requested. If the app has data with different values, this **pricing can be done for each class** of data, **if such classes are requested exclusively**. For example, an app may trade 3 classes of data — class-A data at 1MB = 2 App-datacoin, class-B data at 1MB = 5 App-datacoin, and class-C at 1MB = 20 App-datacoin. However, their **combined value is still 1 MB = 1 datacoin**.

# Datacoin value = cost to create and access data + profits

- Permits absorbing the cost of hosting the app into the pricing model
  - Storing and servicing data access requests for the app data requires compute, storage, and bandwidth resources, which are provided by miners. The cost of infrastructure varies from app to app.
  - This infrastructure cost can be **normalized** into this pricing model, so:

$$\begin{aligned} \text{price of 1MB of data} = & \text{cost of storage to store 1MB of data} + \\ & \text{cost of compute to run the app instances while storing data} + \\ & \text{cost of bandwidth used while storing data} + \\ & \text{cost of compute to serve the data to data buyers} + \\ & \text{cost of bandwidth used while serving the data to data buyers} + \\ & \text{profit for app developers and miners} \end{aligned}$$

- Any class premiums can be added to the above, if the app trades data with different classes.
- Buyers may not be willing to pay the asking price, but that problem is inherent with the value of the data itself and not due to this pricing model.

# Computing the cost of hosting a tApp

1

Length of execution/instance:	<b>500ms</b>
Memory used/instance:	<b>0.5GB</b>
Storage used/instance:	<b>16MB</b>
Bandwidth/instance:	<b>32MB</b>
Write TPS:	<b>200</b>
Read TPS:	<b>5000</b>

Developers submit app proposals describing the resource usages for their apps. The resource usages describe runtime, storage, and bandwidth requirements for creating the app data and accessing it later by data buyers.

2

Total runtime cost:	<b>\$ 52M*</b>
Total storage cost:	<b>\$350M</b>
Total bandwidth cost:	<b>\$ 5M</b>
	----
<b>Total cost:</b>	<b>\$407M</b>

miners use the app proposals to determine the annual infrastructure cost to support the apps on Storecoin Platform. This analysis doesn't account for the value of the data; it only computes the infrastructure cost, if miners were to approve the apps. This cost includes the **costs for both Validators and Messagenodes**.

3

Total cost:	<b>\$407M</b>
Compute premium:	<b>25%</b>
<b>Total compensation:</b>	<b>\$509M</b>

miners may impose a compute premium as the desired profit on the infrastructure provided to the apps. The value of the data and hence the profit sharing (default 30%) with miners should at least be high enough to meet this compensation requirement.

# Estimating the revenue from different classes of data

4

Datacoin ticker: **DXYZ**  
Supported classes:  
bronze - 50%  
silver - 40%  
gold - 10%

The app proposals also estimate how the data is accessed. An app may have one or more **classes** of data and it provides an estimate of % of data created and accessed for each data class. These estimates are used to determine the price of the data in datacoins per MB of data.

5

miner compensation: **\$509M**  
miner revenue share: **30%**  
**Revenue needed:** **\$1.7B**  
App developers get\*: **\$1.2B**

App developers and miners may employ different data valuation methodologies to value data, but a back-of-the-napkin calculation is necessary to determine if the revenue generated will be sufficient for miners to be profitable. If this estimated revenue is backed by data valuation, miners are likely approve the apps.

6

Revenue estimated: **\$1.7B**  
Revenue per class of data:  
bronze - 50%: **\$850M**  
silver - 40%: **\$680M**  
gold - 10%: **\$170M**

Once the total revenue is estimated, the revenue from individual classes of data can be estimated too. In this example, \$850M revenue is estimated to come from bronze class of data and so on.

# Datacoins issued daily = amount of data created daily in MBs

7

Write TPS:	200
Storage/record:	16KB
Data created/day (in MB): (200x60x60x24 x 16KB)	270,000
<b>Daily datacoin issuance:</b>	<b>270,000</b>

Given the write throughput and data size per record, we can compute the amount of the data created per day in number of MBs. The data created in a day will be a **mix of different classes of data**. However, **the datacoin issuance always assumes 1 MB = 1 datacoin**, so the number of datacoins issued in a given day is same as the number of MBs of data created that day. This means, **1 datacoin is normalized across the values of multiple data classes**.

7.1

Supported classes:	
bronze - 50% revenue 150,000 MB	} <b>0.9 datacoins</b>
silver - 40% revenue 70,000 MB	
gold - 10% revenue 50,000 MB	} <b>0.54 datacoins</b>

The price for 1 MB of data of individual classes can be determined as follows.

$$\text{Price} = \frac{\% \text{ revenue of the data class} \times \text{Total MB of data created}}{\text{MB of data of the class}}$$

For example, price of 1 MB of bronze class =  
 $(0.5 \times 270,000) / 150,000 = 0.9 \text{ datacoins}$



# Price per 1MB of data of specific class may vary over time

7.2

Supported classes:

bronze	- 30% revenue	}	0.77 datacoins
	700,000 MB		
silver	- 50% revenue	}	1.00 datacoins
	900,000 MB		
gold	- 20% revenue	}	1.80 datacoins
	200,000 MB		

Total data **1800,000 MB**

Over time, the % revenue for a class of data, the volume of data produced for a class of data, and any premium associated with the class may change. This example illustrates changes in % revenue and volume of data produced for a class of data.

8

Revenue estimated:	<b>\$1.7B</b>
Annual data created*: (270,000 x 365)	<b>98,550,000MB</b>
<b>Annual datacoin emission:</b>	<b>98,550,000</b>
<b>Price/datacoin:</b> ( $\$1.7B / 98,550,000$ )	<b>\$17.25</b>

The price per datacoin can be estimated based on the estimated revenue from the app data and the annual datacoins issued. This assumes that all the data created are sold. In practice, different possibilities exist. For example, only a portion of the data created may have been sold or all the data created are sold to multiple buyers.

# How app developers estimate the price for 1MB of data?

< App Proposal +


Enter the name for your app  
Name of the app

Enter the ticker for your app's datacoin  
Datacoin ticker. Ex: DXYZ

**App specification**

Expected runtime to execute one app instance  
Memory required in GB

Run time in seconds

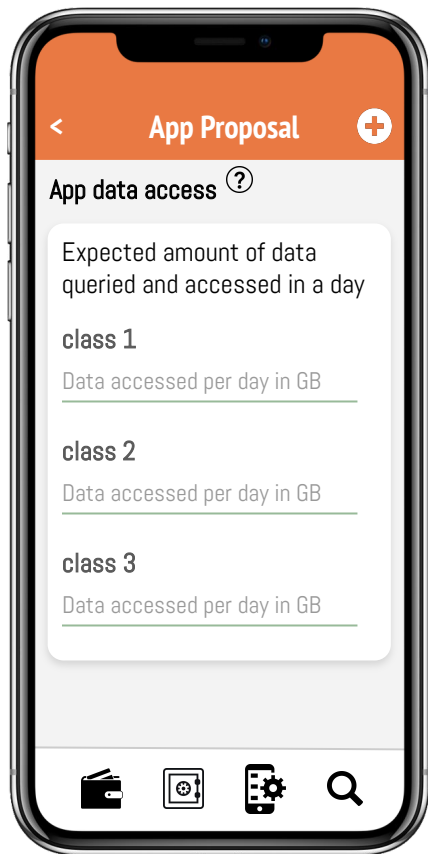
- 1 App developer selects the **App Proposal** option  to submit their app to Storecoin. Data price estimation is done as part of this proposal.
- 2 App developer creates a new app proposal for their app. Modifying an existing proposal uses the same flow.
- 3 App developer specifies the app name and ticker symbol for their datacoin. For simplicity, we ignore other details captured in this flow, such as app icon, developer contact information, GitHub links, etc.
- 4 App developer provides an estimate of the resources needed to run their app.
  - 4.1 The runtime specification is similar to AWS Lambda requirement (in GB-seconds). It specifies the memory required in GB to run one instance of the app and the duration in seconds that the app instance runs.

# App specification — data classes

The image shows a smartphone screen with an orange header bar labeled 'App Proposal' with a back arrow and a plus icon. Below the header is a section titled 'App data classes' with a question mark icon. Inside this section is a white card with the following fields: 'Expected amount of data created by this app in a day' (with a sub-label 'Data created per day in GB'), 'Value of 1 MB of data in \$', and 'Name of this class of data'. Below the card is a checkbox labeled 'Contains PII data' with a question mark icon. At the bottom of the card area is a button 'Add more data classes' with a plus icon. The bottom of the screen features a navigation bar with four icons: a wallet, a gear, a smartphone with a gear, and a magnifying glass.

- 1 App developer describes one or more data classes of data created by their app. Multiple data classes can exist, if the app produces data of different values. Storecoin Platform uses this information to predict the write throughput.
- 2 App developer estimates approximate volume of data created in a day (in GB) and the value of data in fiat for 1 MB of data. The name is optional, if the app creates only class of data. Data created is requested in GB for simplicity.
- 3 App developer flags if this class of data contains personally identifiable information (PII). The treatment for PII is beyond the scope of this spec.
- 4 App developer can add more classes with this option. The same information as above is repeated for every class of data.

# App specification — accessing data



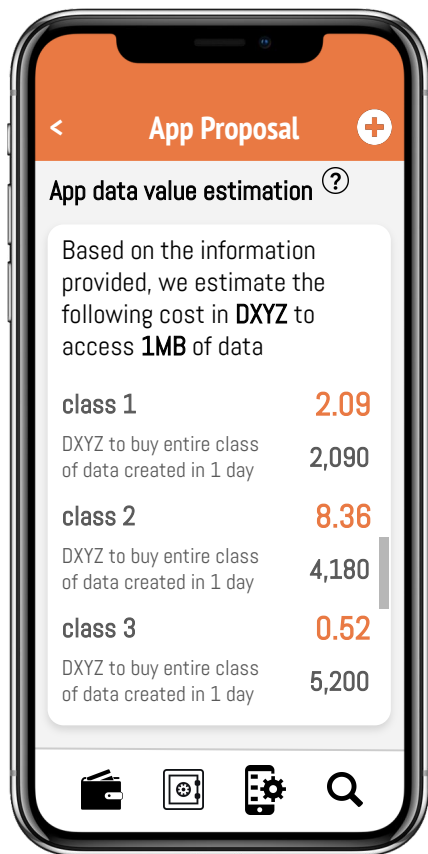
1

In this step, the app developer estimates how their app data is accessed by data buyers. Storecoin Platform uses this information to predict the read throughput.

2

This illustration assumes 3 data classes of this app. The amount of data accessed for each class is estimated. This can change over time. When the app is deployed, the Platform keeps a record of both read and write operations for accurate auditing, so this is meant to be for estimation purposes only.

# App specification — data price estimation in datacoins



## Assumptions:

This illustration assumes the following values for the data classes in slide 12.

- Data class 1: 1GB of data created. This class of data is valued at \$2 per MB. So, total value of data created in 1 day =  $\$2 \times 1000 \text{ MB} = \$2,000$ .
- Data class 2: 0.5GB of data created. This class of data is valued at \$8 per MB. So, total value of data created in 1 day =  $\$8 \times 500 \text{ MB} = \$4,000$ .
- Data class 3: 10GB of data created. This class of data is valued at \$0.5 per MB. So, total value of data created in 1 day =  $\$0.5 \times 10,000 \text{ MB} = \$5,000$ .

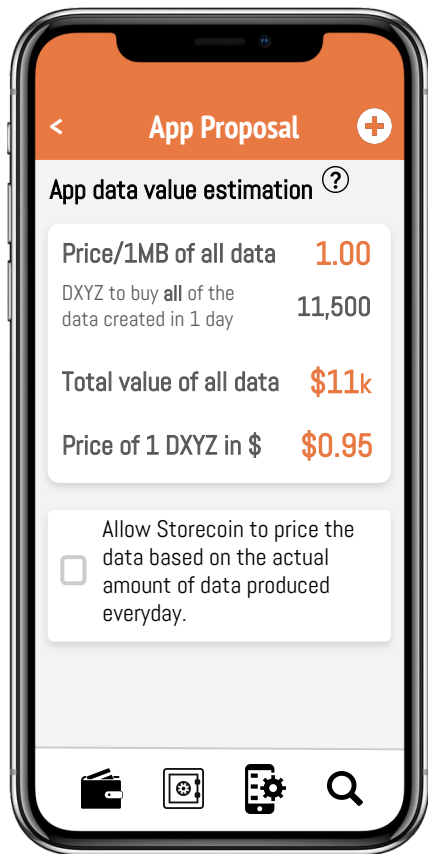
Total value of data created in 1 day = \$11,000.

Total volume of data created in 1 day = 11.5GB.

## Calculations:

- % of value for Data class 1 =  $\$2,000 / \$11,000 = 18.18\%$ .
- % of value for Data class 2 =  $\$4,000 / \$11,000 = 36.36\%$ .
- % of value for Data class 3 =  $\$5,000 / \$11,000 = 45.45\%$ .
- Price for 1 MB of Data class 1 in DXYZ =  $(18.18\% \times 11.5\text{GB}) / 1\text{GB} = 2.09$ .
- Price for 1 MB of Data class 2 in DXYZ =  $(36.36\% \times 11.5\text{GB}) / 0.5\text{GB} = 8.36$ .
- Price for 1 MB of Data class 3 in DXYZ =  $(45.45\% \times 11.5\text{GB}) / 10\text{GB} = 0.52$ .

# App specification — estimate the price of 1 datacoin in fiat



## Calculations:

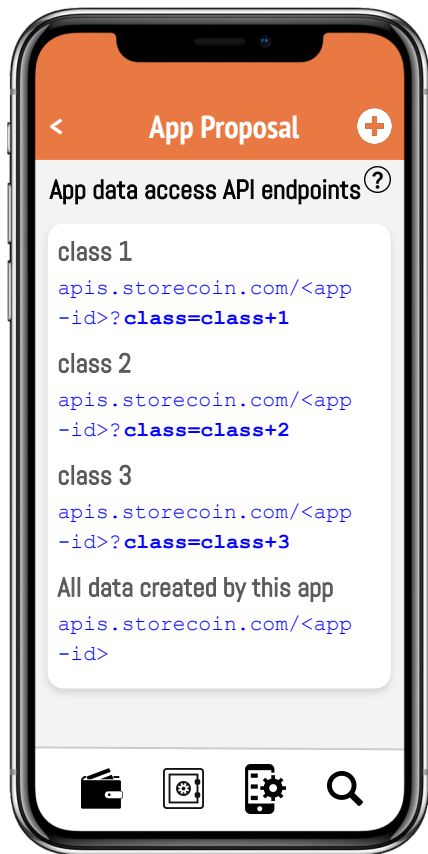
- Price for 1 MB of data in DXYZ to buy entire data created in 1 day = **1.00**.
- Total cost to buy entire data created in 1 day = **11,500**.
- Price of 1 DXYZ in fiat =  $\$11,000 / 11,500 =$  **\$0.95**.

1

The actual amount of data produced every day may vary from what is estimated here. The data volume will also vary from one day to another. This option allows Storecoin Platform to determine the price of the data dynamically based on the actual amount of data produced.

The app proposal flow continues with the estimation of hosting costs to miners, but that flow is outside the context of the developer experience. The rest of the flow will be discussed separately as part of miner experience.

# App specification — suggested data access endpoints



The API endpoints enabled for this app are displayed here. The `<app-id>` is the unique id generated for this app. Note that these API endpoints will be live *after* the app is deployed on the platform (whether zero-fee or paid).

1

Accessing each of these endpoints requires a proof-of-payment in the form of an authorization token. The payment processing flow is discussed later in this spec.

Accessing the above endpoints *without* the authorization token will return the size of the data and other metadata information. This is discussed in the next slide.

# Buying app data — different modes of buying and how they work

Data buying process	Buying options	Data price	Payment options
Buyer wants to purchase <b>all</b> the data created by the app. The data is priced at 1 datacoin for 1MB of data. Buyer cannot buy partial data.	<ul style="list-style-type: none"> <li>- Buy all existing data</li> <li>- Buy all existing and future data</li> </ul>	1MB = 1 datacoin, always.	<ul style="list-style-type: none"> <li>- App's datacoin</li> <li>- STORE (most likely)</li> <li>- Fiat (USD)</li> </ul>
A special case of above where the buyer wants to purchase all the data from multiple or all apps hosted on Storecoin Platform.	<ul style="list-style-type: none"> <li>- Buy all existing data</li> <li>- Buy all existing and future data</li> </ul>	1MB = 1 datacoin of specific app for each app.	<ul style="list-style-type: none"> <li>- Apps' datacoins. Multiple payments required, one for each app.</li> <li>- STORE (most likely)</li> <li>- Fiat (USD)</li> </ul>
Buyer wants to purchase a <b>class</b> of data. This assumes the app has multiple classes of data.	<ul style="list-style-type: none"> <li>- Buy all existing data of specific class</li> <li>- Buy all existing and future data of specific class</li> <li>- Buy part of existing and/or future data of specific class using a filter or query criteria</li> </ul>	1MB = X datacoin where X is determined by app developers based on the data class and its value.	<ul style="list-style-type: none"> <li>- App's datacoin (typically)</li> <li>- STORE</li> <li>- Fiat (USD)</li> </ul>

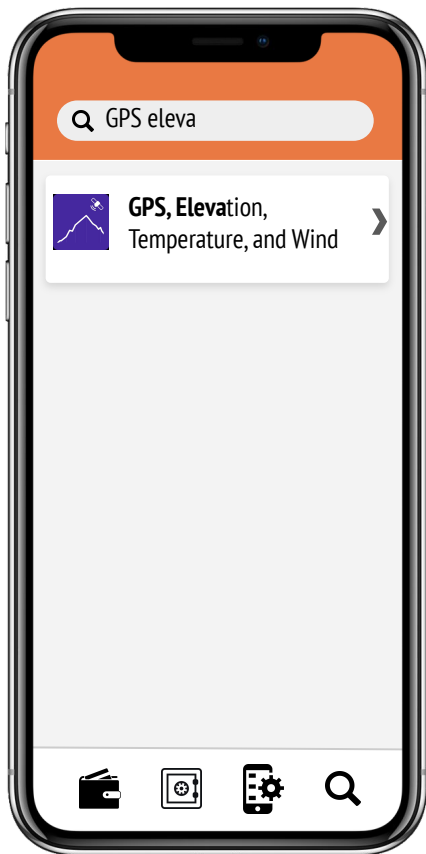


# Buying app data — different modes of buying and how they work

Data buying process	Buying options	Data price	Payment options
A special case of above where the buyer wants to purchase multiple classes of data from same or multiple apps.	<ul style="list-style-type: none"><li>- Buy all existing data of specific classes of one or more apps</li><li>- Buy all existing and future data of specific classes of one or more apps</li><li>- Buy part of existing and/or future data of specific classes of one or more apps using app-specific filter or query criteria</li></ul>	1MB = X datacoin where X is determined by the respective app developers based on the data classes and their value.	<ul style="list-style-type: none"><li>- Apps' datacoins. Multiple payments required, one for each app.</li><li>- STORE (most likely)</li><li>- Fiat (USD)</li></ul>
Buyer purchases data based on certain filter or query criteria. The resulting data may belong to multiple data classes of the same app or multiple apps. This is a special case of the above use case.	<ul style="list-style-type: none"><li>- Buy data based on the filter or query criteria. The filter or query may result in buying future data from the matching apps.</li></ul>	1MB = X datacoin where X is determined by the respective app developers based on the data classes, which matched the filter or query criteria and their value.	<ul style="list-style-type: none"><li>- Apps' datacoins. Multiple payments required, one for each app.</li><li>- STORE (most likely)</li><li>- Fiat (USD)</li></ul>

# Data buyer experience — discover data classes and data price

Purchase all data or all data of specific data class



## Assumptions:

- Data buyer is interested in discovering different classes of data available for sale from a specific app. We use the app described earlier for this illustration.
- The buyers want to know how much the data costs for each class, how much data is available for purchase, the cost of buying all the data created by the app, etc.
- The app described in the app proposal above is assumed to be a GPS, elevation, temperature, and wind velocity monitoring app and the 3 data classes offer this data with varying precision, real time capability, and so on. the specific details are not necessary for describing the buyer experience.

2

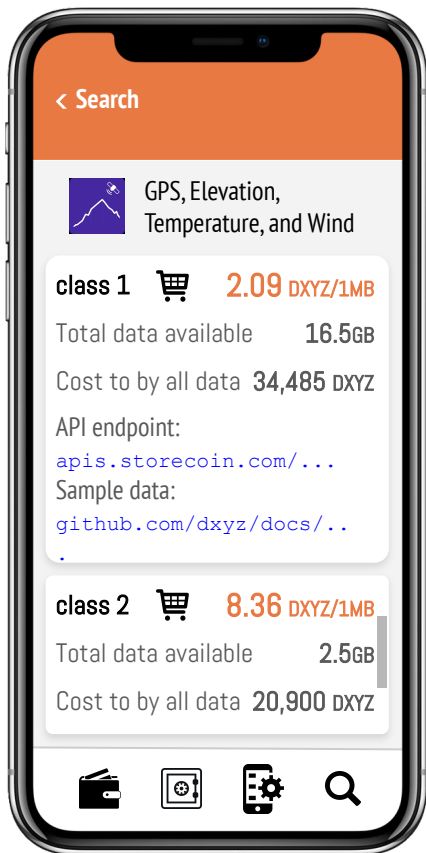
Data buyer enters the search text in the Search box. All matching apps will be displayed as the user types search text. The buyer selects the app they are interested in to discover what it offers.

1

Data buyer selects the **Search** feature in the Storecoin Connect app to search for the app.

# Explore data classes, their price, API endpoints, and sample data

Purchase all data or all data of specific data class




The data classes available for purchase are displayed. For each data class, the following information are displayed:

- the price for 1MB of data in the app's datacoin
- total data size available for purchase
- total cost to buy all the data in this specific class
- the API endpoint to access data in this class, sample data output, and other metadata about the data class.

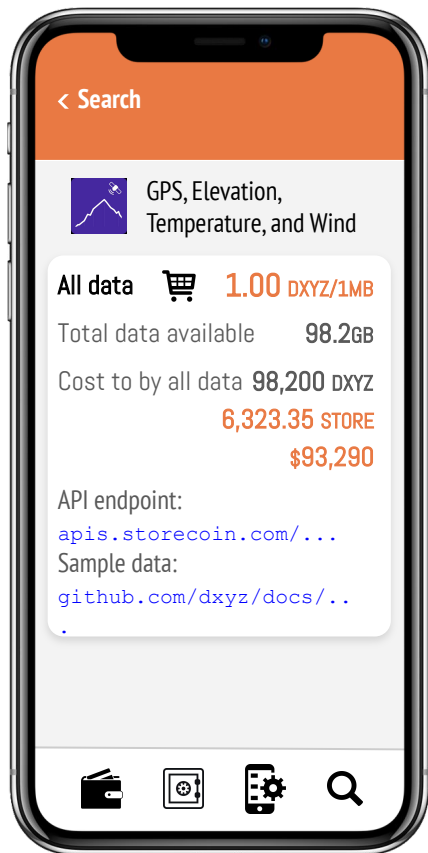
1

The buyer can pay for the data in STORE or fiat also. The price in these currencies are displayed in the checkout screen.

The buyer can buy the class of data by clicking on  and completing the checkout process. The user can filter the data before purchasing it. The filtering and purchase flows are described later in this spec.

# Cost to purchase all the data created in the app

Purchase all data or all data of specific data class



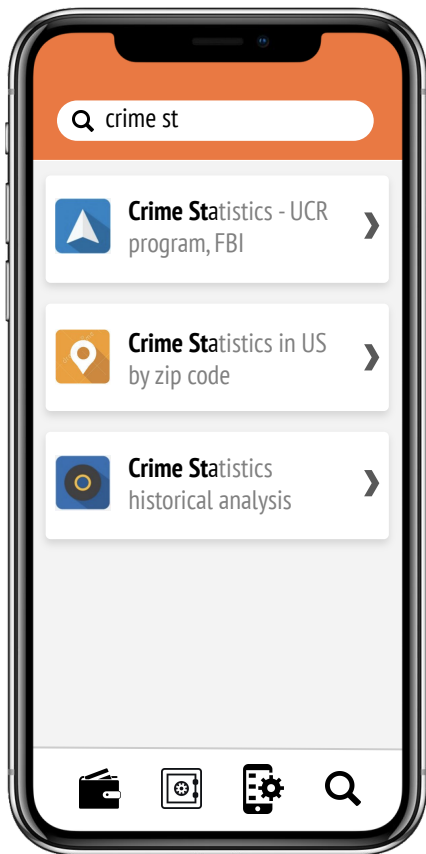
1

Data buyer can purchase all the 3 classes of data together for 1 datacoin for 1MB of data. In this scenario, the price in STORE and fiat are also displayed explicitly instead of just in the checkout screen because the purchaser is more likely to pay in those currencies. The conversion from datacoin to STORE and fiat are done based on the current exchange rates.

The payment and purchase flows are described later in this spec.

# Search by data category or free text

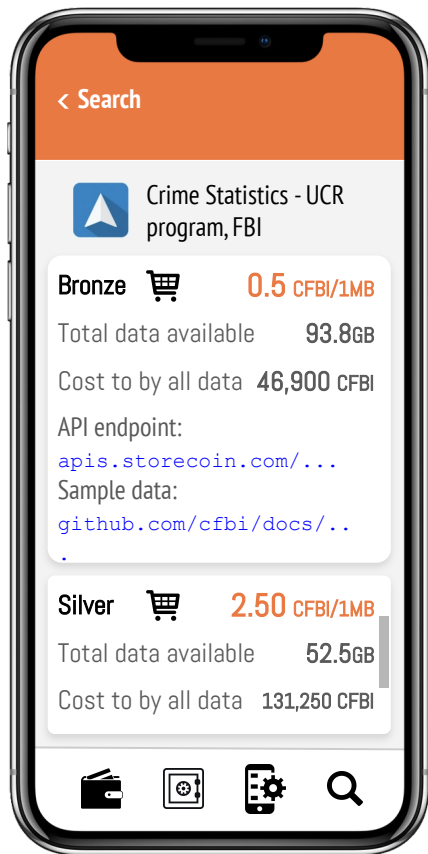
Purchase data using filter or query criteria to search data across multiple apps



- 1 Buyer selects the **Search** feature in the Storecoin Connect app to search for data categories or free text search.
- 2 Buyer enters the search string in the search box.
- 3 The apps matching the search criteria are listed as the buyer enters search text. The search text is matched for app name, description, data categories, searchable, non-PII data, and any metadata marked for aiding the search.
- 4 Buyer selects matching apps to learn more about the data they sell.

# Search may span multiple data classes within or across apps

Purchase data using filter or query criteria to search data across multiple apps



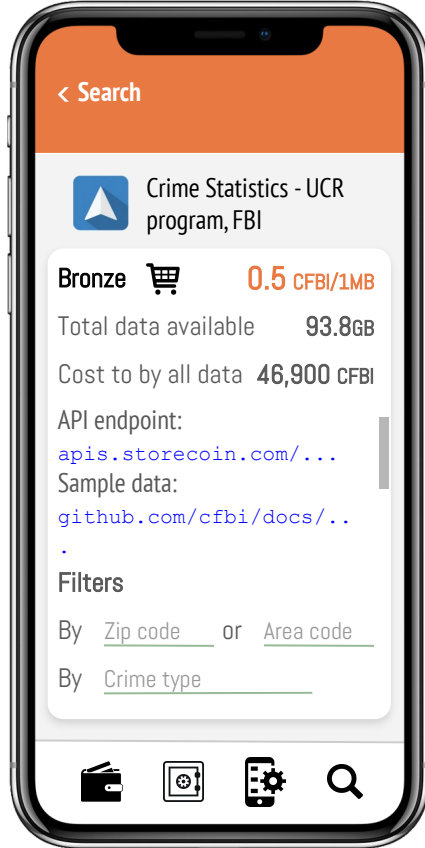
1

This interface is the same as in slide 20 where the buyer searched by app for the data classes made available by that app. The interface is repeated here to show that a given search may result in matching multiple data classes even within the same app. The buyer may look at sample data from multiple data classes to decide which particular class of data to purchase.

Notice that the same search based interface is used to discover specific apps as well as data categories and other searchable metadata. The matching data classes are displayed and the buyer can purchase one or more classes (or all) of data.

# Filtering search results to buy partial data

Purchase data using filter or query criteria to search data across multiple apps

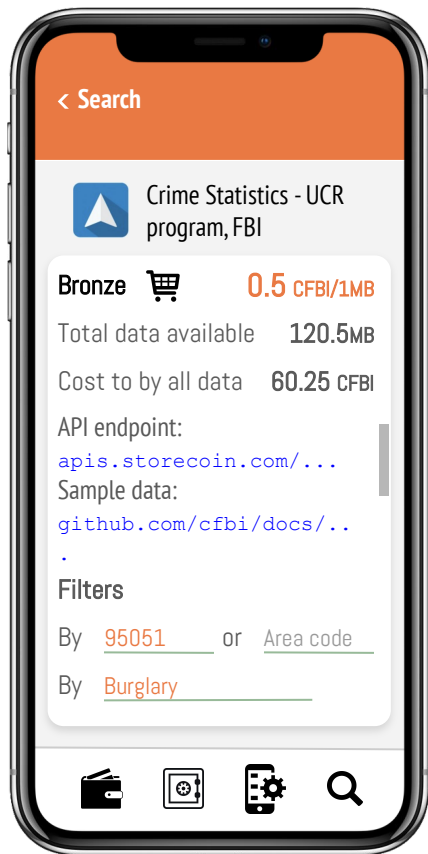


1

The buyer selects a particular data class, which displays extended options available for that class. If the app allows filtering by specific criteria, the filtering options are displayed. The buyer can select appropriate filtering options to narrow down the results and the size of the data they will purchase.

# Data size and price reflect filtered data

Purchase data using filter or query criteria to search data across multiple apps



1

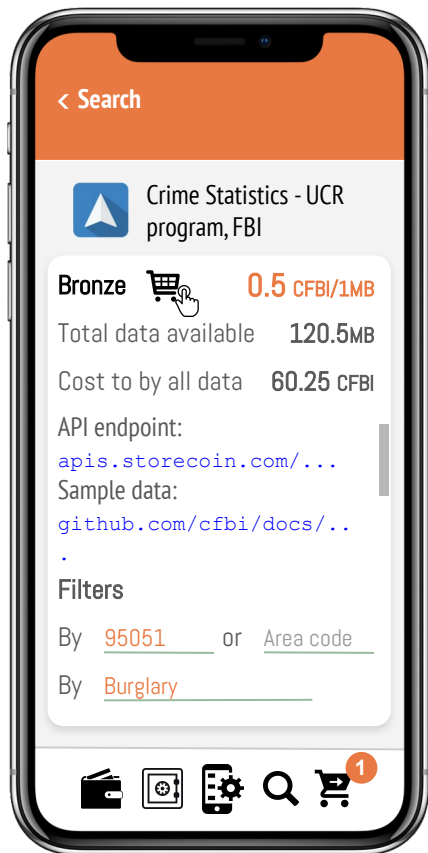
Total data available and the cost to buy the data will change based on the filters applied. The sample data URL may also change (based on the implementation of the app) to show a sample set of actual data matching the search and filtering criteria.

Note that the buyer cannot see the results or access the data until the payment is completed and an authorization token is produced. The buyer only gets to know the size of the data and the cost of accessing the data.




# Purchasing access to the data

Purchase flow for data buyers.



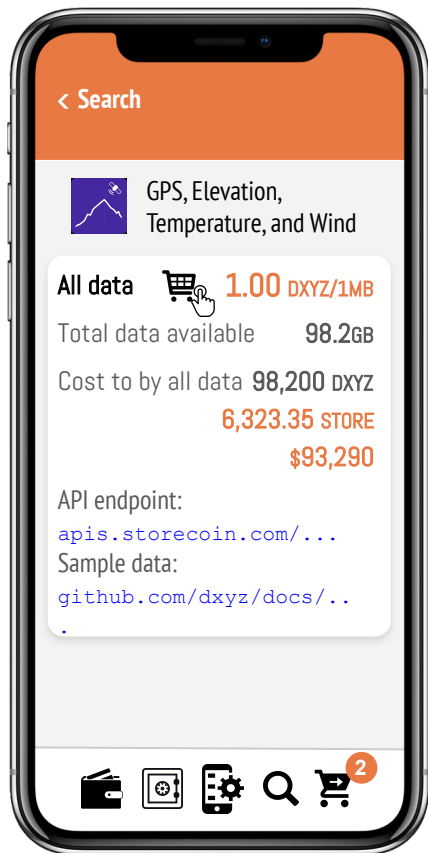
1

The buyer can purchase the data — all data created in the app, all data for a specific data class, or just filtered data — by clicking on  icon. In this illustration, the buyer purchases filtered data of size 120.5GB for 60.25 datacoins.

The shopping cart at the bottom is updated with the number of accesses purchased.

# Purchasing data from multiple apps

Purchase flow for data buyers.



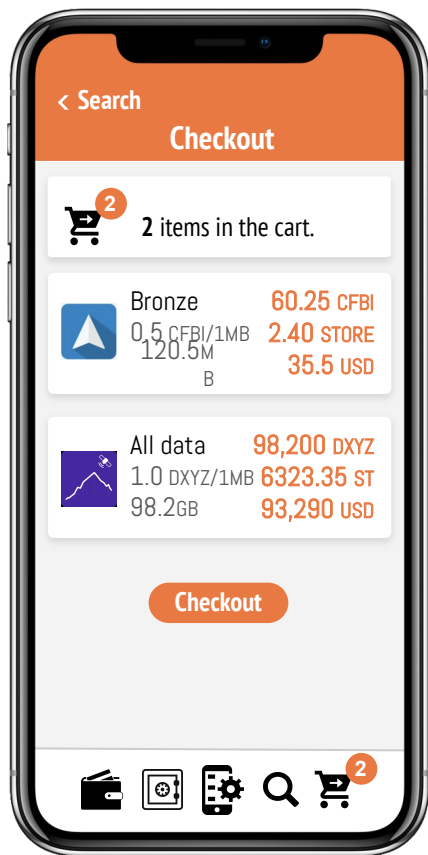
1

This illustrates buyer purchasing disparate data from multiple apps. In this case the buyer purchases all the data produced by the GPS app.

The shopping cart is updated to reflect the number of accesses purchased.

# Checkout process

Checkout process for buyers.



2 The number of checkout items is displayed.

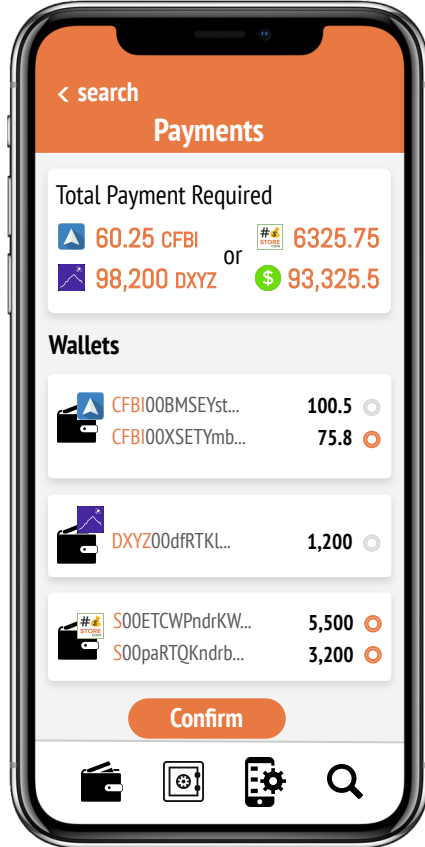
Checkout item details contain:

- 3
- the app, which created the data
  - data class of the data, if any
  - price for 1MB of data in app's datacoin
  - total data size purchased
  - the cost of access in app's datacoin, STORE, and fiat.

4 The buyer selects the **Checkout** option to complete the purchase process.

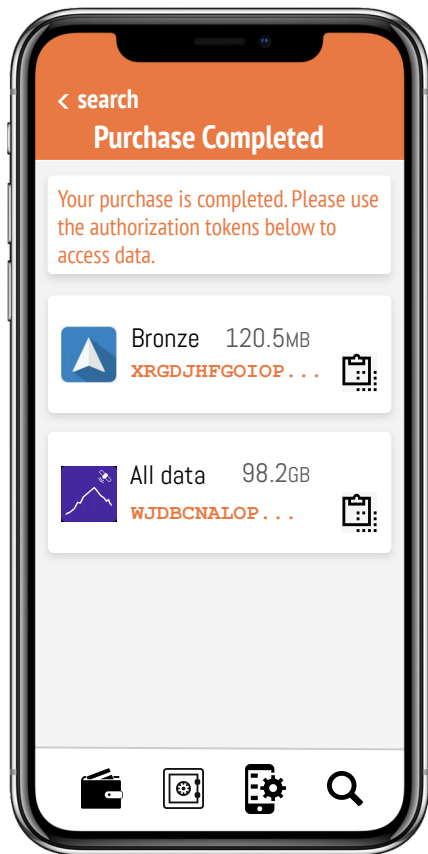
1 The buyer can select the shopping cart to begin the checkout process.

# Buyers complete the payment process



- 1 Total payment required in respective datacoins or in STORE or in the fiat currency are displayed. The buyer can choose any of the payment option.
  - 2 The wallet addresses with respective balances are displayed. The buyer can choose desired addresses to complete payment. The buyer may not have purchased one or more datacoins. They can purchase the datacoins first and then complete the transactions here or they may choose to pay in STORE (or fiat).
  - 3 In this illustration, the buyer has insufficient balance in DXYZ wallet address. So, it is paid with STORE wallet.
- After selecting desired payment addresses, the buyer confirms payment.

# Buyers get authorization tokens for each data class purchased



- 1 Upon successful completion of the transactions, the buyer gets authorization tokens for each data class purchased. The authorization token encodes any filters and other access restrictions, to ensure that the buyer gets exactly what they have paid for.
- 2 The authorization tokens must be protected from unauthorized access, since they can be used by anyone to access the data for which they are authorized. We may require the authorization tokens to be signed by the buyer's private key to prevent unauthorized access by others, so only the authorized buyer can access data.

# Data access example with authorization key

API endpoint:

```
https://apis.storecoin.com/category=crime_statistics&appId=<id of CFBI app>  
&authorization=XRGDJHFGOIOPRKAVYEUFK
```