TEKUMA, INC

The Tekuma Art Curation System

{stwhite,anyati}@mit.edu

Stephen L. White and Afika Nyati

April 26, 2017

1 OVERVIEW

There has been approximately one year to date of active, ongoing technologies development at Tekuma, Inc. When development began in June 2016, very little web infrastructure existed besides the tekuma homepage[1]. The first goal realized was to build a system for onboarding, cataloging, and accessing artworks in an effort to streamline the art curation process happening at Tekuma, and provide a more scalable business model. One year later, much of the infrastructure for this goal has been implemented, and is ready for innovative and more marketable applications to be built on top of this curation infrastructure. (give a total count of lines of code, components, etc)

2 COMPONENTS

The infrastructure which would most help Tekumas curation process was not clear at the time development began. Design decisions were made as they arose, as Tekuma's needs and use cases were evolving with the development. Different prototypes were being tested to analyze different possible web technologies that could be used. After two weeks of prototyping, it became apparent that some preliminary specification needed to be defined in order to structure development of the core infrastructure and to drive production forward. In June 2016, Stephen decided that a four-sided approach would best meet the needs of Tekumas complex business model, and multiple

[1] http://tekuma.io. A WordPress static website

	Name	Targeted Users	URL	Repository
A	The Artist Portal	Artists	https://artist.tekuma.io	artist-portal
B	The Business Portal	Property Owners	https://business.tekuma.io *	n/a
C	The Curator Portal	Curators	https://curator.tekuma.io	curator-portal
D	The Discover Portal	Consumers	https://discover.tekuma.io	n/a

Figure 2.1: the ABCD User Abstraction

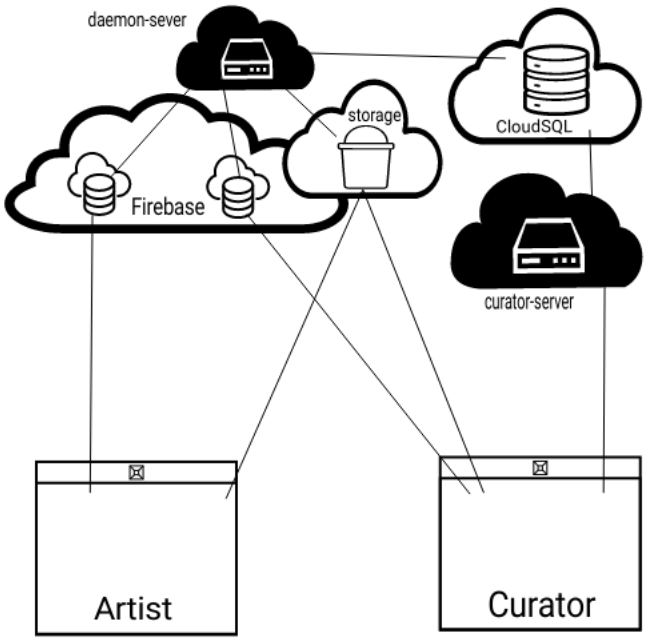

Figure 2.2: Arist-Curator System Diagram: (1) The daemon-server listens to the Firebase DBs and can insert into the SQL DB. (2) Firebase Databases hold all interface-state related data for both portals. (3) The storage bucket holds all of Tekuma's images, and acts as a CDN to the portals. (4) The curator-server mediates queries from the curator to the SQL DB.

targeted users. Specifically, an underlying cloud-based system serves 4 web apps to act as portals into different parts of Tekuma's curation system. These web apps are referred to as the ABCD user abstraction. This model is described below and summarized in figure (2.1), where each entry represents an online hosted application. All code connected to the curation system is stored in Tekuma's GitHub. [2]

[2]https://github.com/tekuma

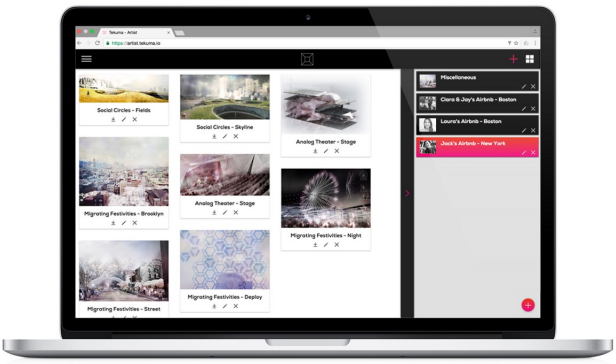

Figure 2.3: The Artist Portal

2.1 BOILERPLATE

Before creating the web applications, a general boilerplate was designed for online user interfaces. The boilerplate makes up Tekumas online graphic and visual identity. The UI primarily defaults to customs and principles dictated by Googles Material Design[3] in line with the current standards in User Interface design. Also included with the boilerplate is Tekumas general CSS stylesheet, and a set of interface icons created by Afika Nyati. The style sheet builds upon the material design guidelines by customizing it to Tekumas aesthetics. More technically, the interface is written in JavaScript ES6+ using ReactJS, transpiled using webpack and BabelJS. State in the interfaces, user account creation, and file uploads are all handled by Google Firebase and managed in the Firebase Conosole[4]. For simplicity, all code in the system is written in JavaScript and is either executed in-browser, or with NodeJS (v6.4.0LTS). Together, the stylesheet, ReactJS interface components, logo, color palette and gradient, type choices (Nexa font, utf-8) makeup Tekumas online visual identity, an extrapolation of Tekumas brand identity developed by Kwaku Opoku in collaboration with Tekumas founders. (do we hold trademarks on the font usage?)

2.2 ARTIST

The first component designed and implemented was the beta version of the Artist portal. The Artist portal is powered primarily by Google Firebase, and was originally designed to function without server-side code as a staticly served interface. All communications from the portal to the rest of the curation system are handled through the Firebase Realtime Database. The Artist Portal provides two main functionalities to its users, as well as a standard interface for editing and supplying personal information about themselves and their artistic style.

[3] https://material.io/guidelines/
[4]

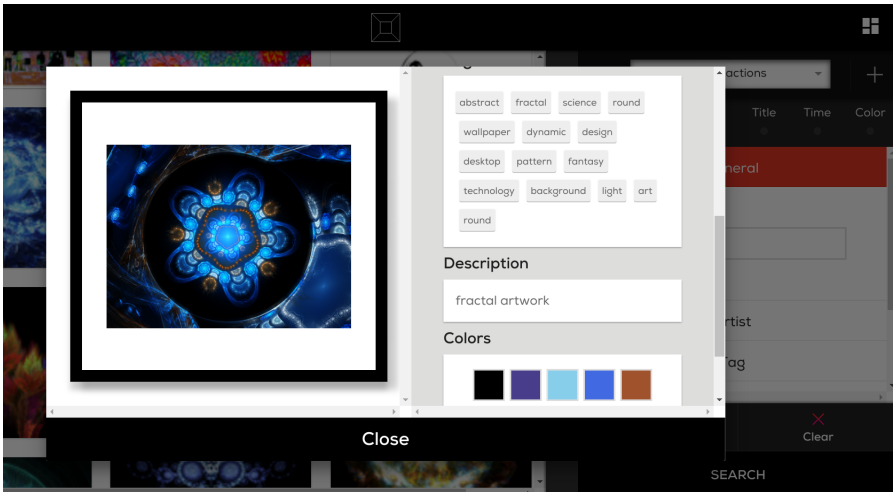

Figure 2.4: Screenshot from the Curator Portal

2.2.1 Artist Studio

The primary goal of the portal is to handle uploaded and cataloging of artworks. All uploaded files are saved into Google Cloud Storage buckets, located at in the Cloud Console[5] and all data cataloged in hierarchical (non-structural) format in the Firebase Database. The organization of these files within the storage buckets is detailed further in url-schema.md in the Documentation repositiory[6]. Later, functionality was added to do extra processing on uploaded artworks to provide more labeled data. This functionality exceeded the scope of client-side execution, and integrating HTTP communication to a server would incure a large overhead. Instead, a daemon running on a Google Compute Engine (GCE) cloud-hosted server handles this extra processing. Communication to the daemon is facilitated by Google Firebases real-time database where a first-in-first-out (FIFO) stack is used to keep track of jobs for the daemon to execute. The extra labeling includes gathering text tags and color palletes from the uploaded images using machine-vision technology, and is further explained in (X.X).

2.2.2 Artist Gallery

2.3 Business

2.4 Curator

server and daemon (modularity)

2.4.1 Curator Search

2.4.2 Curator Manage

2.4.3 Curator Review

(discuss daemon)

2.5 Discover

3 Unfinished Components

3.1 Business

3.2 Discovery

4 Databases

4.1 Arist Database

Branches of the firebase database. Used to handle state of artist.tekuma.io interface

4.2 Curator Database

branches of the firebase database Used to handle state of the curator.tekuma.io interface

[5]
[6]

4.3 TEKUMA ARTWORK DB

4.4 ART.COM DATABASE

The Art.com Database, which is roughly 10MB, contains **2,311,390** rows of data, in structured SQL format. Examples of these rows are below.

```
columns = [ 'SKU', 'ITEM_TITLE', 'ITEM_LONG_DESC', 'ARTISTNAME',
'ITEM_TYPE', 'IMAGE_URL', 'PRODUCT_URL', 'PRODUCT_TAXONOMY' ]
actual_row = [ '30742196502A',
  'Spurzheim Bust',
  'JOHANN KASPAR SPURZHEIM German phrenologist with his autograph',
  'H Corbould',
  'Photographic Print',
  'http://cache2.allpostersimages.com/MED/\\84\\8488\\5WQK300Z.jpg',
  'http://www.art.com/products/p30742196502-sa-i9032642/.htm?RFID=197560',
  'World Culture>European  Cultures>German Culture>>>>>>>' ]
```

5 RELATED WORK

1. **ArtFlip** (`https://www.artflip.com/product`) A website very similar in UI and UX to the curator portal.

2. **Artsy** (`https://www.artsy.net/collect`)

3. **Canvis** (`http://canviz.co/contact/`) A mass producible digital picture frame, including 3D printer files.

4. **StitchFix** (`https://www.stitchfix.com/`) a clothing company which uses a mixture of expert human stylists and artificial intelligence to pick out outfits for customers on a subscription plan.

5. **Turning Art** (`https://www.turningart.com/gallery`) A curator company with a similar buisness model of directed towards office, commericial, and multi-family real estate.

6. **Large-scale Classification of Fine-Art Paintings** (`https://arxiv.org/pdf/1505.00855v1.pdf`) Details large scale classification algorithms for fine art paintings.

7. **A Neural Algorithm of Artistic Style** (https://arxiv.org/pdf/1508.06576.pdf) Describes methods of transfering artist styles between artworks use Convolutional Neural Nets.

8. **Inceptionism: Going Deeper into Neural Networks** (https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html) Discussion of Google Deepdream

6 Potential Innovative Development Projects

6.1 Collaborative Art Recommender System

This section details methods which are *agnostic* to the content. In other words, the AI is unaware that it is filtering artworks specifically. Collaborative filtering is currently a ubiquitous feature of internet based creative content platforms such as Spotify and Netflix. Collaborative filtering is commonly implemented with either data clustering, or low-rank matrix multiplication (better for sparse data), and relies on having a active user base. The underlying assumption of collaborative filtering is that if user (A) likes artworks a,b,c,d and user (B) likes artworks a,b,c, then it is likely he will also like artwork d as well. The strength of this method is that it is agnostic to the content, which is especially powerful for art and media, as it is inherently abstract and hard to break it down into concrete features.

6.2 TekumAI, Curated Recommender System

This section details methods which are *content-based*, meaning that the features of the artwork are directly considered by the system. This relies on the assumption that all artworks in the system have a concrete list of their features, known as a feature vector. Features in this vector have a numerical representation, and are chosen such that they partition the artworks into separate sets which people might conceivably assign different preferences to. Then, each use is assigned a vector, θ, which is continually updated to reflect the user's preferences. When this vector is initialized (usually with random values), it is not very powerful. This is known as the cold-start problem. Methods for circumventing cold start can "borrow" data from other places. Specifically, data could be borrowed from similar users, such as descbried in the previous section with collaborative filtering. The user could also be presented with a survey of his art tastes, which could be used to initialize the vector to something more accurate than if it was just set to random values.

Given a feature vector $x^{(i)}$, we can predict a rating $\hat{y}$ by:

$$\hat{y}^{(i)} = \theta \cdot x^{(i)} = \sum_{j=1}^{d} \theta_j x_j^{(i)}$$

- "Good features are those that partition the artworks into sets that people might conceivably assign different preferences to"