**Chaktim Wong - 50280143**

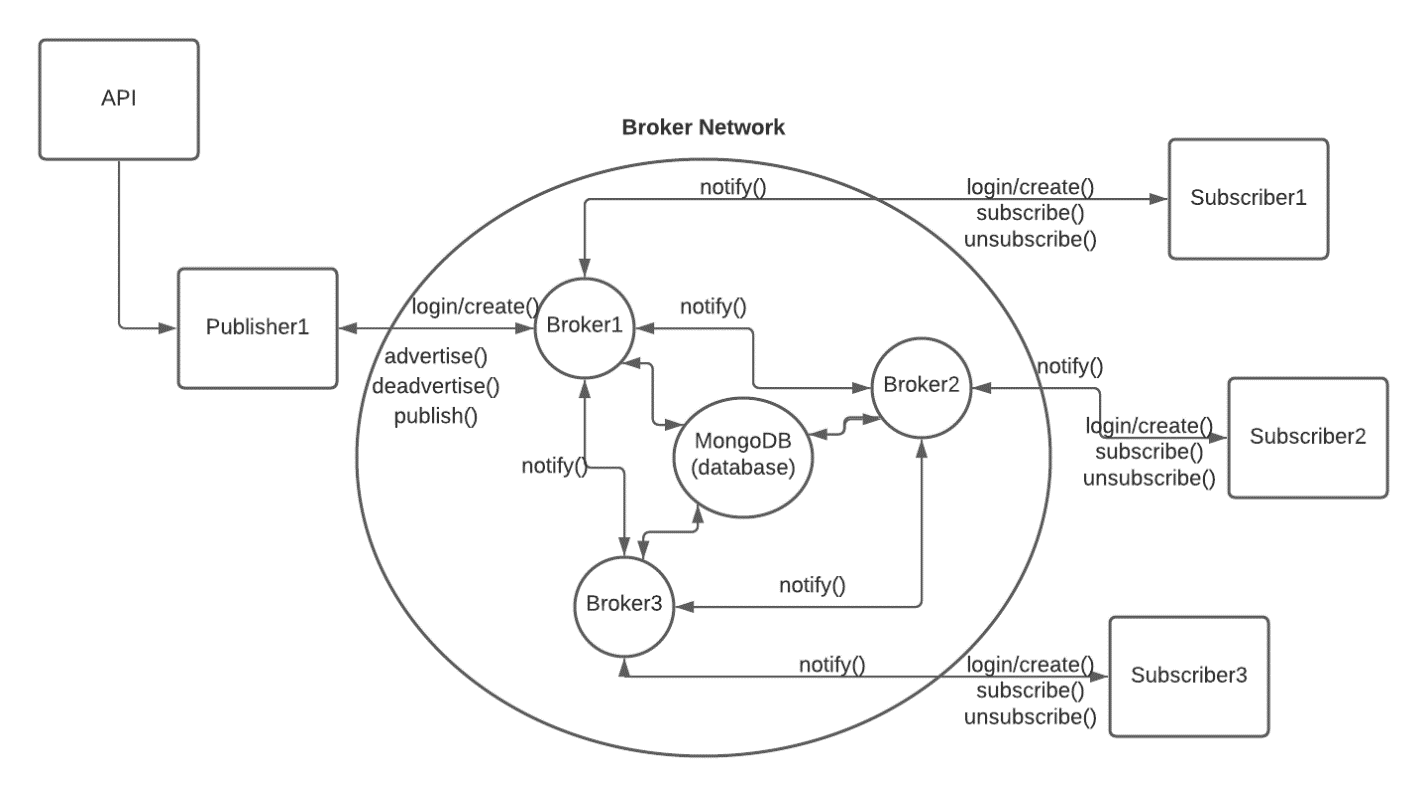
**chaktimw@buffalo.edu**

**Group Number 75**

**API USED: “https://apidocs.cheapshark.com/#intro”**

* **Displays games in the Steam Library with their normal prices, current discounts, and more**

**Architectural Diagram:**

****

**Architecture: Sub/Pub System with a simple broker network (topic-based)**

* **Brokers**
  + **Three broker containers serve as the central points of access for subscribers and publishers. Only the brokers have access to the mongoDB database container.**
* **Publisher**
  + **One publisher container contains a server that responds to the url:** [**http://localhost:8101**](http://localhost:8101)
  + **Publisher is separated from subscribers and brokers**
  + **User can create and login to their ID (no password account system)**
  + **Once login is complete, user can advertise any price topics (whole numbers only), and publish events containing game title and price from the steam library API (shown at the top of this page)**
  + **Keeps track of number of published events, and uses that as reference for the next publish (server indexes API using the tracked number)**
* **Subscribers**
  + **Three subscriber containers each have their own server for their respective URL (shown in the next section)**
  + **On the subscriber page, users can create an ID and login. From there, they can subscribe to any of the topics advertised by the publisher. Only advertised topics can be accepted. Once subscribed to a topic, if the publisher publishes a game with a price lower than the subscribed price topic, this game will display in the newsfeed section.**
  + **Any games that have been published after the creation of an ID can be viewed by that ID. In other words, published events are filtered and stored under the advertised filters in the database.**
* **docker-compose**
  + **Creates a container for each broker, publisher, subscriber, and the database**
  + **Assigns the appropriate dockerfile for each container**
  + **Connects all the container onto the same docker network**
* **Dockerfiles can be found under dockerfiles in the project directory**
* **Html files are under templates and JS code is under static**
* **Database management is handled by each broker node using the methods in database\_tools.py under the database folder**

**Startup Instructions:**

1. **Run Docker**
2. **Go to project file directory in cmd**
3. **Run command “docker-compose up --build”**
4. **Wait until docker finishes starting up**

**Publisher url:** [**http://localhost:8101**](http://localhost:8101)

**Subscriber 1 url:** [**http://localhost:8201**](http://localhost:8201)

**Subscriber 2 url:** [**http://localhost:8202**](http://localhost:8202)

**Subscriber 3 url:** [**http://localhost:8203**](http://localhost:8203)

1. **Go to publisher URL and type a whole number in the create an id section to create an id (publisher functions cannot be used without logging in)**
2. **Login with the created id**
3. **Advertise any price numbers (ex: 5, 10, 30)**
4. **Open a subscriber URL (each URL is connected to a different broker)**
5. **Create an id and login with it (separate from publisher id)**
6. **Subscribe to any of the advertised price topics**
7. **On the publisher’s page, hit publish to publish one game from the API (you can repeat press it to publish more games)**
8. **These published games will display on the subscriber’s screen if they fall under their subscription filter (Under $5, Under $10, Under $30, etc.). No repeats.**
9. **Each subscriber container will get notified of every published event through their respective broker.**