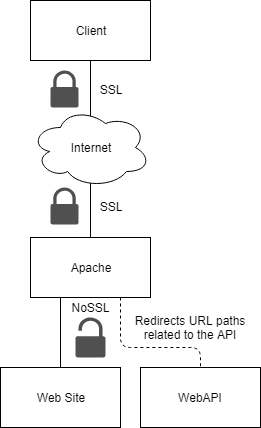
Background

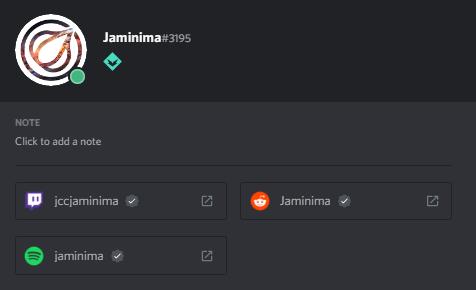
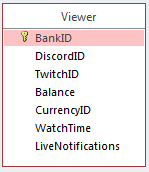
I intend to create a Viewer Reward Currency, which will reward members of a streamers community, for their time spent watching streams and chatting in discord and twitch. Further rewarding them for donating and subscribing. The main improvement over similar systems is the ability to have discord and twitch accounts linked. Along with the introduction of a web api where bots will be able to modify user balances, once authorised by the currencies owner.

Similar to existing systems like StreamElements, currency owners will be able to configure the currencies configuration files. This will include the setting of responses, ie what I said to the user when something happens, initial balances etc. As well as integration with Nightbot(an automatic song playing service), streamlabs (for their donation handling and sound alert system) and youtube (in order to get the latest video on their channel).

The integration with Discord and Twitch will require the use of their respective API’s, which will each return a similar set of objects which can’t be immediately interchanged and used in place of one another. Hence to remove the need for duplicate code. I instead will create a standardised set of objects, which will convert the Discord and Twitch message objects into cross compatible formats. That I can then use to create a single command handler instead of 2 separate.

Our web api along with the other api integrations will require the use and implementation of OAuth, in order to ensure no one can impersonate us.

Our web api will not have any local encryption, but the apache server the traffic is redirected through, will be SSL enabled, hence data will be secure on the www but insecure on the local machine.

Due to the requirement for users to access the same viewer account using both discord and twitch, the Viewer object will have to track both their Discord and Twitch Id’s. Hence being 2 identifiers, that when combined with a CurrencyID will be the main way of uniquely identifying the Viewer.

To create the link between Discord and Twitch, we will read the connections present on a users Discord account, if a twitch connection is present, we will use the connections twitchid as the viewers twitchid from then on. ONLY if it is not being used by another viewer of the same currency.

People who wish to create a currency using the system, will be able to create an account using our website. Where they will then be able to create and customise the currency using a simple and intutive configuration page. For users who wish to integrate with the api, they will be able to create a bot. Which can then be given permission by a currenies owner to modify their currency. Authorised bots, like currency owners, will have full power to modify the currency, but wont be able to delete the currency or authorise other bots.

Simple integrations are possible, that wont require authorisation. But these will only be able to read public information, like the currency configs, viewers, etc. all private data like auth tokens, emails etc will not be revealed to anyone except those authorised to see said information.

Exisiting solutions

**StreamElements**

As previously mentioned StreamElements provides a viewer reward currency that will be similar to function as mine. To create the currency using StreamElements you are required to sign into their site using your twitch, where you are then able to configure the currency. Once configured the system will immediately begin running.

StreamElements provides a website where command details can be viewed by viewers along with other related data. Which I also intend to create in order to make the viewers experience easier.

However it lacks integrtion with Nightbot, StreamLabs and youtube. Which we will use to expand functionality.

**NightBot**

NightBot uses their own bot which sits in the stream in order for them to respond to song requests. It lacks in all other functionality, so it is of little use.  
My bot will be able to handle these commands in its place, hence removing the need for NightBots bot.

Objectives

1. Web API
   1. Interacts with a normalised database
      1. Parameterised SQL
   2. Reading and writing of JSON configuration files
      1. Login Configuration
         1. Stores required 0Auth Details
      2. Command Configuration
         1. Configure what strings trigger a command
         2. Configure what string is sent to a user, using a simple parameterised message structure, that data can easily be inserted too.
         3. Configure rewards, probabilities and costs
   3. Ability to respond to GET requests
      1. Logins
      2. Currencies
      3. Viewer
      4. Bots
      5. Modify a currencies login configuration on receipt of an OAuth authorisation redirect
   4. Ability to respond to POST requests
      1. Serve protected information
         1. To Authorised bots
         2. To logged in users
      2. Create and login users
      3. Create currencies
      4. Modify currency configuration files
      5. Modify currencies viewer accounts
      6. Authorise bots for a currency
2. Apache server to redirect traffic from SSL enabled external connection
3. Objects to represent database tables
   1. Viewer
   2. Currency
   3. Login
   4. Bot
4. Discord-Twitch bot
   1. Run multiple instances of bot for every currency on the WebAPI
   2. Refresh bot instances configuration files, to ensure they are working with up to date configuration files.
   3. Use of Discord.Net and TwitchLib C# Libraries
   4. Interaction with Youtubes, Nightbots, StreamElements and our own Web API
      1. Perform 0Auth correctly in order to acquire Authorization tokens.
   5. Have easy to edit configuration files, to allow for easy customisability
   6. Reward Users for
      1. Watching Stream
      2. Messaging in Discord or Twitch
   7. Commands to
      1. Ability to Pay other users
      2. Gamble / Slot, Slot has lower odds but higher multiplier
      3. Duel another user, to win or lose the duel amount
      4. Easily add simple “echo” commands, like social platforms
      5. View the latest video on the youtube channel
      6. Fish, cast a “line” and gain a “item” which has an equivalent value
      7. View information relating to the music player
      8. Request a song
      9. Cancel last requested song
      10. Alert, play a sound effect
   8. Moderator Commands to
      1. Give a user some currency
      2. Refresh the configuration files
      3. Set game or title of stream
      4. Remove a song from queue
      5. Play/pause or skip the song

Interview w/Harbonator – The Main Client

Q: What do you look for in a Viewer Reward Currency?

A: Mainly the simplicity for me to setup and maintain it. Our current system requires basically no work on my part, as all of the complexity is hidden away. Also the ability to fairly reward viewers which feels progressive and satisfying.

Q: What is good about the current system?

A: It works consistently, also it was incredibly easy to setup and required no maintenance/upkeep on my end.

Q: What problems do you have with the current system?

A: The inability to create new features/commands greatly restricts what can be achieved and the rewards where quite lacklustre.

Q: What new features would you want in the new system?

A: A vast variety of rewards, easily customizable and easy for the user to interact with.

Q: How would you like people to be able to use the new system?

A: In twitch chat and discord.

Q: How would you like to modify the currencies configuration?

A: Preferably using some sort of web site, in order to make it easy and to reduce hassle.