Note* I wasn't able to implement authentication tokens because I got confused on how to hide the userId inside the token and then check for it every time in the methods. I did implement other forms of validation and sanitizing input. Other than that and a slight redesign of each endpoint, I delivered all of the ideas from my pitch.

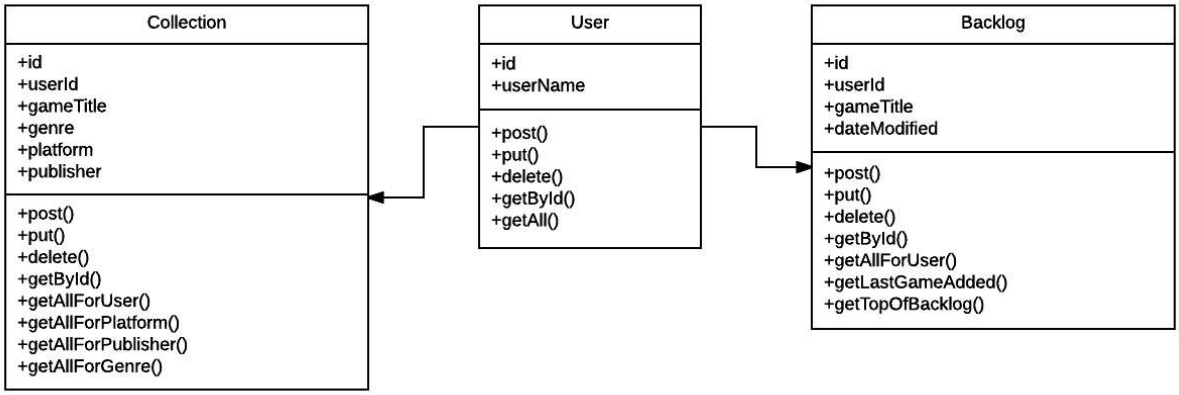
Here's some basic documentation to describe and help test my web api:

Project Summary

I wanted to make a web api for a video game backlog. A backlog is essentially a living list of video games any person is currently playing through at a given time. It's a tool to help solve a problem I personally have with playing video games: I start playing them, but I rarely see most of them through and finish any of them. It's a very nerdy way to organize a nerdy hobby. It's separate from the entire collection because there may be video games that a person owns but has either completed or not started yet. The backlog is loosely ordered, meaning it doesn't necessarily depend on a strict ordering system (like a queue), but knowing order is nice to theoretically prevent having games that never leave the list.

Endpoint Descriptions

Here is a crude class diagram. This roughly describes the relationship between the three endpoints:



User:
The user endpoint is fairly basic and straightforward. You are able to add a user, edit a user, delete a user, get a user by ID, and get all users. See below for a list of links to test the api, followed by expected JSON data for each method.

Backlog:
I decided to leave extra details of the game off of the backlog list, since the user will be able to have a more detailed list of their collection. Besides, my original design of the Collection and Backlog were a bit too similar, so I decided to change them up a little. The vital information on a backlog is the game's title, and date it was added/modified. Each user is able to add to and remove from, and update their list. A user is able to get the entire list, the last game added/modified to their list, as well as the top of the list (first game added). Additionally, there is a getByID, which returns any game by any id. This is for any list. Again, see below for test links and expected JSON for necessary links

Collection:
The collection endpoint is perhaps the most interesting. I decided to add this to increase the scope of the project. A user is able to add, update, and delete games to their collection. In this list, the games contain extra information such as title, genre, publisher, and platform. Additionally, the user is able to get their list by all of these fields. See below for a list of links to test the api, followed by expected JSON data for each method. Again, see below for test links and expected JSON for necessary links.

Endpoint urls

User
POST newUser: http://icarus.cs.weber.edu/~rs82945/vgbacklog/v1/users
```
{
  "userName":"Batman"
}
```

PUT updateUser: http://icarus.cs.weber.edu/~rs82945/vgbacklog/v1/users
```
{
  "userId":"2",
  "userName":"Bob Ross"
}
```

GET allUsers: http://icarus.cs.weber.edu/~rs82945/vgbacklog/v1/users/all
GET userById: http://icarus.cs.weber.edu/~rs82945/vgbacklog/v1/users/{userId}
DELETE userById: http://icarus.cs.weber.edu/~rs82945/vgbacklog/v1/users/{userId}

Backlog

POST newGame: http://icarus.cs.weber.edu/~rs82945/vgbacklog/v1/backlog

```
{
  "userId":"1",
  "gameTitle":"Batman: Arkham Asylum"
}
```

PUT updateGame: http://icarus.cs.weber.edu/~rs82945/vgbacklog/v1/backlog

```
{
  “backlogId”:”1”,
  "userId":"1",
  "gameTitle":"Batman: Arkham Knight"
}
```

DELETE ById: http://icarus.cs.weber.edu/~rs82945/vgbacklog/v1/backlog/game/{userId}
GET entireBacklog: http://icarus.cs.weber.edu/~rs82945/vgbacklog/v1/backlog/{userId}/all
GET lastGameModified: http://icarus.cs.weber.edu/~rs82945/vgbacklog/v1/backlog/{userId}/last
GET topOfBacklog: http://icarus.cs.weber.edu/~rs82945/vgbacklog/v1/backlog/{userId}/top
GET backlogByID: http://icarus.cs.weber.edu/~rs82945/vgbacklog/v1/backlog/game/{backlogId}

Collection

POST http://icarus.cs.weber.edu/~rs82945/vgbacklog/v1/collection

```
{
  "userId":"1",
  "gameTitle":"Halo 2",
  "genre":"Shooter",
  "platform":"Xbox",
  "publisher":"Bungie"
}
```

PUT http://icarus.cs.weber.edu/~rs82945/vgbacklog/v1/collection

```
{
  "collectionId":"5",
  "userId":"1",
  "gameTitle":"Super Mario 64",
  "genre":"Platformer",
  "platform":"Nintendo 64",
  "publisher":"Nintendo"
}
```

DELETE byId: http://icarus.cs.weber.edu/~rs82945/vgbacklog/v1/collection/game/{collectionId}
GET byId: http://icarus.cs.weber.edu/~rs82945/vgbacklog/v1/collection/game/{collectionId}
GET collectionByUser: http://icarus.cs.weber.edu/~rs82945/vgbacklog/v1/collection/{userId}/all
GET byPublisher:
http://icarus.cs.weber.edu/~rs82945/vgbacklog/v1/collection/{userId}/publisher/{publisher}

GET byPlatform:
http://icarus.cs.weber.edu/~rs82945/vgbacklog/v1/collection/{userId}/platform/{platform}
GET byGenre: http://icarus.cs.weber.edu/~rs82945/vgbacklog/v1/collection/{userId}/genre/{genre}