

Endpoints

The endpoint for the site will be in the following format (the AWS API doesn't exist yet):
[https://\[API-ID\].execute-api.us-east-2.amazonaws.com/](https://[API-ID].execute-api.us-east-2.amazonaws.com/)

The following will be endpoints that append to the end of the above API URL.

- Home Page
 - /collection
 - Request Type: GET
 - Response Type: JSON
 - Success: 200
 - Failure: 500 (Server Error)
 - Resolves to the home page of the web application. This will return the entire collection of all currently stored video game entries
- Add Game screen
 - /add
 - Request Type: POST
 - Response Type: JSON
 - Success: 200
 - Failure: 405 (Invalid Input)
 - This will accept game information from the form and submit it to the database as an insert
 - **Example: Adds a game to the database with the below information**
 - Request:

```
{
  "Title": "Arcade Madness",
  "Developer": "Super Dev",
  "Publisher": "Lucas Arts",
  "Genre": "Action",
  "Description": "Test description",
  "Rank": 3,
  "Image": {image blob},
}
```
 - Response: Code: 200

- View Game screen
 - /delete/{GameID}
 - Request Type: DELETE
 - Response Type: JSON
 - Success: 200
 - Failure: 400 (Game ID not found)
 - This endpoint accepts a GameID to delete from the database
 - **Example: Delete record with GameID 123**
 - Request:


```
{
                    "GameID": 123
                  }
```
 - Response: 200
 - /update/{GameID}
 - Request Type: PUT
 - Response Type: JSON
 - Success: 200
 - Failure: 402 (Game ID not found)
 - This endpoint accepts a GameID along with any information to update on that database record.
 - **Example: Updates game id record 222 and has its rank changed to 5**
 - Request:


```
{
                    "GameID" : 222
                    "Rank": 5,
                  }
```
 - Response: 200
 - /rank
 - Request Type: GET
 - Response Type: JSON
 - Success: 200
 - Failure: 500
 - This request pulls back all of the games in the database that the current profile owns, and then ranks them on a graph based on the the rank table property
 - **Example:**
 - Request: NONE
 - Response: 200 (all game records owned by profile id)

User login / registration

- Will be utilizing AWS Cognito to manage user registration and login
- Login success returns access token

Stretch Features

- View Game screen
 - /request/{GameID}
 - Request Type: POST
 - Response Type: JSON
 - Success: 200
 - Failure: 402
 - This request grabs the GameID and sends a request to the owner of that game.
 - **Example: Request for game id record 12**
 - Request:

```
{
  "GameID": 12
}
```

Response: 200