**2nd Partial Exam**

**Problem:**

Provide a web application that manages the CRUD operations, a search engine to view the specified videogame by its name or its developer and provide a Front-End to the user to interact and view the information.

**Priorities:**

* API to perform the CRUD operations and perform the search operations with an engine to retrieve the videogames that match either the developer company or videogame title specified in the input string.
* Database to store the videogames data.
* Implement a Front-End web page to interact with the API and display the information of each videogame or the selected ones.

**Restrictions**

* The time to develop and deploy the web application is of only 2 days so the final product will be completely functional but will need more time to be completed to have all the necessary functionalities to work without any errors on the runtime and security characteristics.

**Design:**

* System requirements:
  + Videogame CRUD
  + Database
  + Front-End web page
  + API (Back-End)
    1. List of tuples to store the SQL query results from the database.
    2. List of dictionaries to store the serialized data.
  + Render and Vercel to deploy the web application.
* Development:
  + Web development will be used since all the services are designed to be accessible from the web browsers.
* Data modeling:
  + Input:

1. String containing the name of a videogame or a specified developer company.
2. GET request.
3. POST request
   1. Title, description, developer, release\_year, classification, image and banner
4. DELETE request
   1. videogame\_id
5. PUT request
   * Output:
     1. List of dictionaries containing the information of the videogames where either their names or developers matched with the input search string.
     2. List of dictionaries containing all the information of all the videogames stored in the database.
     3. List of dictionaries containing all the information of all the videogames stored in the database including the appended videogame by the post method.
     4. List of dictionaries containing all the information of all the videogames stored in the database except the one deleted.
     5. List of dictionaries containing all the updated information of all the videogames stored in the database.

**Architecture Pattern:**

* Client-Server pattern

**Data Base Metadata:**

videogames:

* videogame\_id
* title
* description
* developer
* release\_year
* classification
* image
* banner

**API Endpoints:**

* /api/videogames
* /api/videogames/<videogame/developer name> (Just GET)

**Methods:**

* GET: to get the information of all the videogames from the Data Base or from only the one specified
* POST: to append a new videogame to de Data Base
* DELETE: to remove the specified videogame from the Data Base
* POST: update the information of the specified videogame

**Architecture Diagram:**

Diagrama

Descripción generada automáticamente

**Git Repositories:**

* API: https://github.com/EmilioRivera0/videogame\_api
* Web Client: https://github.com/C4ncino/videogame\_api\_client

**DEPLOY:**

* API: https://videogame-api-kouw.onrender.com
* Web Client: https://videogame-api-client.vercel.app/