Trello planning  
Github setup - TBD  
  
1st Notebook: Used to install and use SteamAPI and it's wrapper to make my queries to the SteamAPI  
This is probably the Notebook I’ll use for my DEMO  
2nd Notebook: Used to import and clean our dataset (fetched from Kaggle), do some visualization and export our tables in SQL after creating the according dataframes  
3rd Notebook: Used to address the ML part of the project

**Modifications to implement:**   
Transform categorical columns into numerical  
Verify the changes it provokes in Visualization

**VIZ**

Appid

release\_date

developer

publisher

required\_age- ok

positive\_ratings - ok

negative\_ratings- ok

average\_playtime - ok

price - ok

**RECOMMENDING**  
  
PLATFORMS

GOOD SCORE  
GOOD AVERAGE PLAYTIME

**DEVELOPPER**

PRICE

RELEASE DATE

**SQL**

**Relational Databases (RDBMS):**

**Relational databases organize data into tables with predefined schemas and use structured query language (SQL) for data manipulation. They provide strong data consistency, integrity, and support for complex queries and transactions. RDBMS are suitable for applications that require complex relationships between entities and need ACID (Atomicity, Consistency, Isolation, Durability) properties, such as financial systems, inventory management, and customer relationship management (CRM).**

**Yes, if you have chosen an SQL database, it falls under the category of relational databases, which is the first type I mentioned. Relational databases use structured query language (SQL) for data manipulation and organization. They are characterized by tables with predefined schemas and support for complex relationships, transactions, and queries. If you have specific requirements that align with the features and strengths of relational databases, then choosing an SQL database is a suitable decision.**

Table: whole store - Columns: «appid», «name», «release\_date», «english», «developer», «publisher», «platforms», «required\_age», «categories», «genres», «achievements», «positive\_ratings», «negative\_ratings», «average\_playtime», «median\_playtime», «owners», «price»

Table: Game Info - Columns: «appid», «name», «release\_date», «developer», «publisher»

Description: This table contains basic information about each game, including its unique identifier («appid»), name, release date, developer, and publisher.

Table: Game Genres - Columns: «appid», «genres»

Description: This table represents the genres associated with each game. It includes the «appid» column as a foreign key referencing the Game Info table and a column for the genres the game belongs to. Multiple genres can be stored as a comma-separated list or in a separate table if there are many-to-many relationships[.](https://file+.vscode-resource.vscode-cdn.net/c%3A/Users/theya/Mon%C2%A0Drive/iRonhack%20on%20G-Drive/n/n4.)

Table: Game Categories - Columns: «appid», «categories»

Description: This table represents the categories associated with each game. It includes the «appid» column as a foreign key referencing the Game Info table and a column for the genres the game belongs to. Multiple genres can be stored as a comma-separated list or in a separate table if there are many-to-many relationships.

Table: Game Ratings - Columns: «appid», «positive\_ratings», «negative\_ratings

Description: This table stores the ratings for each game, including the number of positive ratings and negative ratings it has received. The «appid» column serves as a foreign key referencing the game»s unique identifier in the Game Info table.

Table: Game Pricing - Columns: «appid», «price»\n - Description: This table contains the pricing information for each game, including the «appid» column as a foreign key referencing the Game Info table and the price of the game.

WOULD HAVE LIKED TO DO BETTER WITH MORE TIME

HAVE A MORE RECENT DATABASE – but spending an extra 10-15h to get it renewed was unrealistic so despite being really unhappy about I just went along with it.

GET MORE PRECISE RECOMMANDATIONS BASED ON:

WHAT FRIENDS PLAY

USER BUDGET

USER EXPERIENCE

RAPPORT :

Intro parlant du projet de Brocommender : un mot sur le GTO Solver

Expliquer les différents fichiers .ipynb que l’on va utiliser dans le projet et leur rôle

**TIPS :**

**ON DATA** **NORMALIZING**:

Don’t forget to normalize your data, especially if you are making a recommender system!

* For numerical data (e.g. price, counts, etc.), you should normalize by **column** (i.e. **feature**), for instance with StandardScaler (<https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html>)
* For one-hot encoded or count-encoded data (e.g. country, region, or text), you should normalize by **row** (i.e. **individual**), before concatenating with the other columns. You can use for that <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.Normalizer.html>