Project Title: Steam Games Recommender
Project Summary: Our project will utilize the steam games database. Using this database, we will construct a web application that will recommend games based on filters inputted by the user. We will have a number of filters that will allow the user to construct exactly what type of game they are interested in. This will allow multiple games to be recommended with high accuracy based on the data.

1. **Describe what data is stored in the database. (Where is the data from, and what attributes and information would be stored?)**

In this dataset, we have a comprehensive list of video games that are compatible with personal computers (PCs). The data is organized in a table, sorted alphabetically and segmented, possibly by categories like genre or operating system. While the games listed are playable on PCs, it's important to note that they are not exclusive to this platform meaning some games might also be available on consoles or mobile devices.

The dataset takes inspiration from steamdb.info, a well-known source that amalgamates information from Steam's publicly accessible APIs as well as from steamspy.com. These sources are valuable because they aggregate a wide variety of data points, making our dataset robust and multi-faceted. The games included in this dataset are compatible with a range of PC operating systems such as Windows, Linux, DOS, Unix, and OS X. This diversity ensures that the dataset caters to a broad spectrum of PC users, irrespective of the operating system they use.

One notable criterion for inclusion in the dataset is that the games must be natively playable on a PC. This means we have excluded games that are only playable on a PC through the use of an emulator—a software that mimics another system to run foreign software. This ensures that the dataset is focused only on games that are directly compatible with PC hardware.

Finally, the dataset contains multiple parameters or attributes for each game to provide a detailed view. These include:
- **Game Name:** The title of the game.
- **Developer:** The company or individual responsible for creating the game.
- **Producer:** The company or individual responsible for producing or distributing the game.
- **Genre:** The category to which the game belongs (e.g., action, adventure, RPG, etc.)
- **Operating System:** The PC operating systems that the game is compatible with.
- **Date Released:** The date on which the game was officially released to the public.

This extensive information makes the dataset a valuable resource for anyone interested in PC gaming, whether for research, analytics, or personal interest.

2. **What are the basic functions of your web application? (What can users of this website do? Which simple and complex features are there?)**

Our web application will be a game recommender based on the Steam games dataset. Users will be able to set up their preferences to find their ideal game in the dataset. Users will be able to set up filters on date released, genre, producer, developer, game name, operating system and many other characteristics of the game. We hope to have a user-friendly interface so it's easier for users to find their favorite games.

Some of the simple features in our application would be to filter games in the dataset since we can just perform operations that we have learned during class in the dataset and find the games that fulfill those requirements. One of the more complex features that we are hoping to implement is a feature that would filter games based on the text box written by the user.

3. **What would be a good creative component (function) that can improve the functionality of your application? (What is something cool that you want to include? How are you planning to achieve it?)**

The user would be able to describe the type of game they would be interested in a text box. Our web application would go word by word and find synonyms for each word input to see whether one of those synonyms falls under one of our filters and select it if true. Whenever a user thinks about which filter boxes to tick, they think far too objectively on the experience they want to have rather than what they really want. The list of ten of the top-most recommended games to the user, taking advantage of the recommendation count provided in the dataset.

To be able to find synonyms of the text-input provided by the users, we plan on using PyDictionary, a library with a built-in English dictionary with synonym finding capabilities. To coincide with our filters, we plan on storing all the synonyms for the filters using the same system. This is convenient for our application since we plan on running the entire back-end of the project solely on Python. This also supports our hopes of being able to work with multiple languages other than English, seeing how large and expansive the Python library is, there are plenty of libraries that support the same functionality in different languages.

4. **Description of an application of your choice. State as clearly as possible what you want to do. What problem do you want to solve, etc.?**

With thousands if not millions of games on Steam, it is difficult to find the right game that suits the user's needs. The application that we will be programming will help the user the perfect games based on the what the user is trying to look for, whether a games that is

- Game Style: Adventure, RPG, Simulation

- Languages: English, German,
- Number of Players: Co-op, Multiplayer
- PC Requirements: MacOS, Linux, Windows
- Released Date: Before 2000, 1980s arcade games

This website application solves the problem of users buying games that don't suit the user needs. On top of that, the website application also recommends the user games that other people have recommended.

5. **Usefulness. Explain as clearly as possible why your chosen application is useful. Make sure to answer the following questions: Are there any similar websites/applications out there? If so, what are they, and how is yours different?**

The website application is multifaceted in ways that different products can be used instead of games. Based on the data, we can filter and recommend products to the user depending on the user's needs like trying to find a specific item:

- Groceries: Dairy products, vegetables, fruits
- Kitchen: Utensils, Pans, Gloves
- Bathroom: Soap, Towels

There are applications like the Steam search bar that are similar to the website application that we are trying to implement. What makes our application special is we would recommend the user games based on a short description that the user would give.

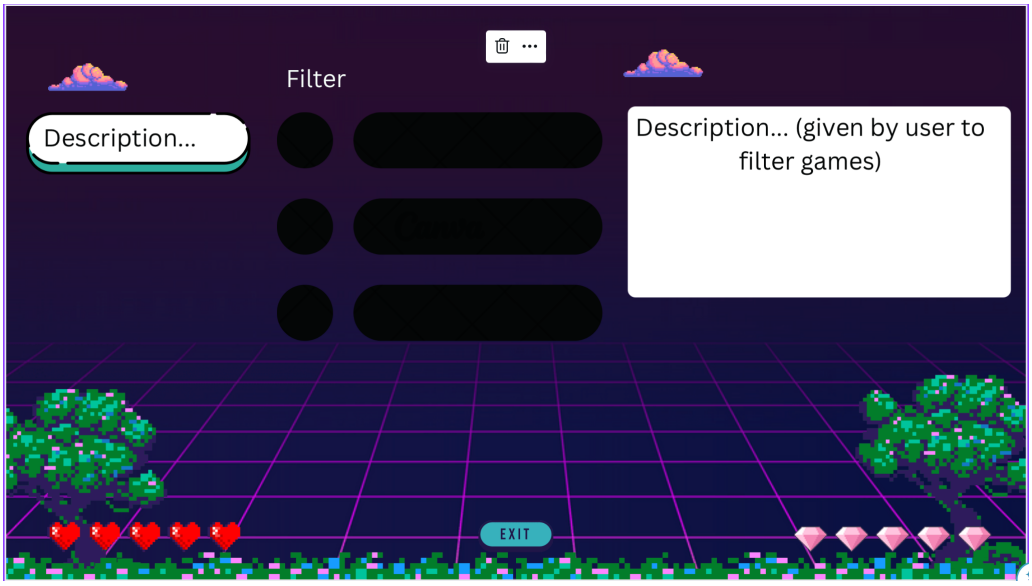
Ex.
User: "I would like to play a game with a genre FPS"
Webapp: – Filter games that has FPS based on data and description." —

6. **Realness. Describe what your data is and where you will get it.**

We rely on actual game data sourced from Steam API and steamspy.com to make our web application genuinely useful and relevant. This real-world data ensures our platform is trustworthy for gamers, developers, and researchers, offering up-to-date insights, reviews, and trends to inform their decisions and strategies.

By harnessing data from the Steam API and steamspy.com, we're able to provide users with an immersive and authentic experience within the gaming realm. Whether you're a casual gamer searching for your next adventure, a game developer seeking market insights, or an industry analyst tracking trends, our platform is your reliable compass. We believe that the power of real data makes our web application an indispensable tool for navigating the dynamic world of video games with confidence and precision.

7. **Description of the functionality that your website offers. This is where you talk about what the website delivers. Talk about how a user would interact with the application (i.e., things that one could create, delete, update, or search for). Read the requirements for stage 4 to see what other functionalities you want to provide to the users. You should include:**
    a. **A low-fidelity UI mockup**:



    b. **Project work distribution**: Who would be responsible for each of the tasks or subtasks?
    List of the person responsible for which exact functionalities in section 6. Explain how backend systems will be distributed across members. Be as specific as possible as this could be part of the final peer evaluation metrics.

For JavaScript functionalities, Julian will focus on form validation and error-handling, while Kervi will be responsible for DOM manipulations and AJAX calls to the backend. In essence, Julian will lay down the foundational elements of the frontend, and Kervi will bring them to life with interactive features.

On the backend side, Rudy and Sebastian will work in tandem to create a robust search and filter functionality. Rudy will develop the Python-based search algorithm, likely using frameworks like Flask or Django. Sebastian will complement this with SQL queries designed for efficient data retrieval. For the filter options, Rudy will handle the Python code for sorting and refining search results, whereas Sebastian will set up SQL views and stored procedures to assist in filtering. Additionally, Sebastian will be the custodian of the SQL database, establishing tables, relations, and indexes, and Rudy will fine-tune the database schema for optimal performance. Both will also collaborate to create secure and functional API endpoints.
To keep everyone in sync, the team will hold daily stand-ups to discuss progress, plans, and any potential blockers. Code reviews will be carried out within the frontend and backend teams to

ensure code quality. All members will utilize Git for version control, making use of feature branches which will be merged into the main branch post-review. Documentation of the entire project will be a collaborative effort, ensuring that every piece of code, whether frontend or backend, is well-documented.

Response to TA Feedback:

Based on our TA's feedback, we have decided to implement changes in our project proposal to incorporate CRUD operations to ensure a dynamic and interactive user experience. This will empower users to add, view, modify, or delete their game preferences and recommendations.

Our project aims to allow users to add their favorite games to a preference list. The user will be able to filter out games based on their preferences from the original Steam database and will be able to select desired games to add to the preference table. This will be the breakdown of CRUD operations:

Create: Allow users to add their favorite games to the preference list. Every time the user marks a game with the start, the game will be added to the preference list with all the information about the game. The user will also be able to create user information and log in so they can access their preference list.

Read: Display a list of game recommendations based on user preferences and show game details like genre, game style, number of players, date released, and other information about the game.

Update: The user will be able to correct information about a specific game in their preference list. An example is when the user knows the correct date released of a specific game, so the user fixes this error. The application will ensure that the correct data type is stored in each column of the preference table.

Delete: Enable users to delete games from the preference list. The user can also delete games based on one or a few filters. The user can also delete their information if they don’t want to use their account anymore.

For future stages here is our rough outline for distribution of work.

Stage 3: For the database implementation Sebastian and Kervi will work on implementing the **main tables** of the database as well as providing the DDL commands. Julian and Rudy will work on filtering data to put it into our database. Then each group of two will create at least one **advanced query** for our database. As a whole group we will then complete the indexing of each advanced query.

Stage 4: Kervi and Julian will work on front end and design of application interface. Sebastian and Rudy will construct any required **procedures** and **triggers** with Julian aiding them in connecting the trigger or procedure to the front end.