Roblox Games App

C++ Final Project

Julius L. Luttmann

A C++ Console Application with Long-Term Data Storage

PROGRAM DETAILS

Program Overview

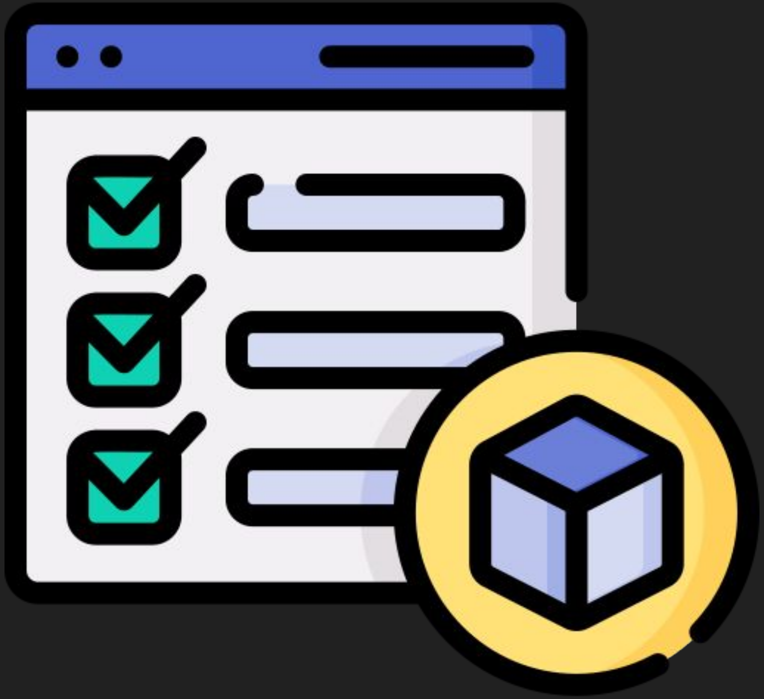
The Roblox Games App allows users to:

- Create accounts and log in securely
- Browse a dataset of Roblox games
- Search and filter games by name or rating
- Save favorite games for long-term use

The program is entirely console-based but structured like a full application

Key Features

- User authentication (login & sign-up)
- CSV dataset loading and parsing
- Search and filtering functionality
- Per-user favorites system
- Persistent data stored across program runs
- Modular, readable program structure

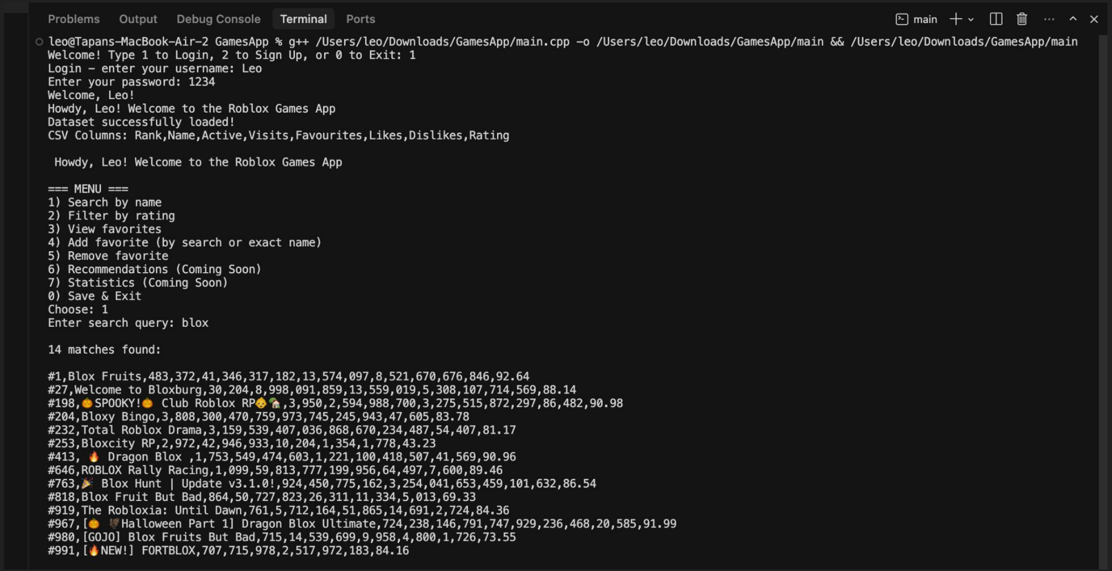


User Interface (GUI) Description

- The program uses a text-based user interface (TUI)
- Features a repeated menu system that guides users step-by-step
- User input is validated to prevent crashes
- Clear prompts and labeled menu options improve usability

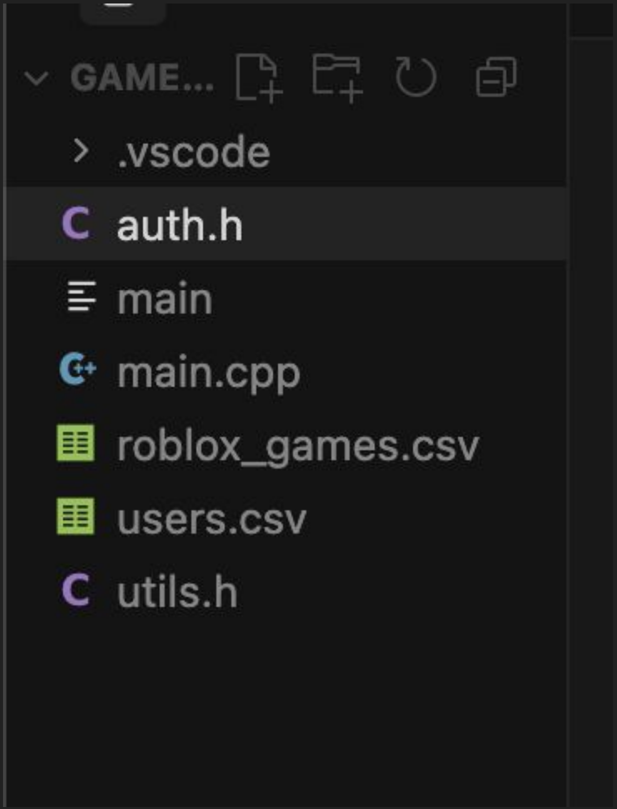
Example:

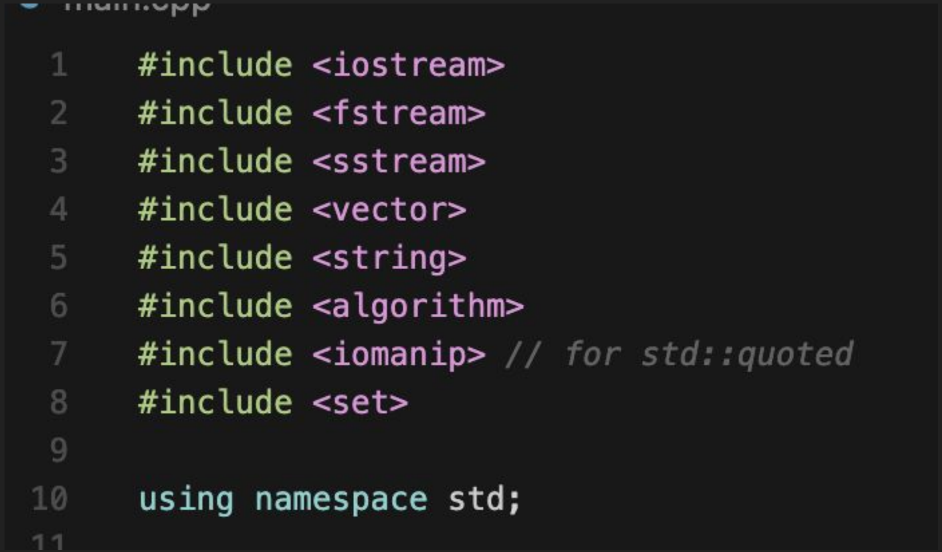
- Numeric menu choices (1–7)
- Text prompts for search queries and ratings



```
leo@Tapans-MacBook-Air-2 GamesApp % g++ /Users/leo/Downloads/GamesApp/main.cpp -o /Users/leo/Downloads/GamesApp/main && /Users/leo/Downloads/GamesApp/main
Welcome! Type 1 to Login, 2 to Sign Up, or 0 to Exit: 1
Login - enter your username: Leo
Enter your password: 1234
Welcome, Leo!
Howdy, Leo! Welcome to the Roblox Games App
Dataset successfully loaded!
CSV Columns: Rank,Name,Active,Visits,Favourites,Likes,Dislikes,Rating

 Howdy, Leo! Welcome to the Roblox Games App

=== MENU ===
1) Search by name
2) Filter by rating
3) View favorites
4) Add favorite (by search or exact name)
5) Remove favorite
6) Recommendations (Coming Soon)
7) Statistics (Coming Soon)
0) Save & Exit
Choose: 1
Enter search query: blox

14 matches found:

#1,Blox Fruits,483,372,41,346,317,182,13,574,097,8,521,670,676,846,92.64
#27,Welcome to Bloxburg,30,204,8,998,091,859,13,559,019,5,308,107,714,569,88.14
#198,🎃SPOOKY!🎃 Club Roblox RP👶🏡,3,950,2,594,988,700,3,275,515,872,297,86,482,90.98
#204,Bloxy Bingo,3,808,300,470,759,973,745,245,943,47,605,83.78
#232,Total Roblox Drama,3,159,539,407,036,868,670,234,487,54,407,81.17
#253,Bloxcity RP,2,972,42,946,933,10,204,1,354,1,778,43.23
#413, 🔥 Dragon Blox ,1,753,549,474,603,1,221,100,418,507,41,569,90.96
#646,ROBLOX Rally Racing,1,099,59,813,777,199,956,64,497,7,600,89.46
#763,🎉 Blox Hunt | Update v3.1.0!,924,450,775,162,3,254,041,653,459,101,632,86.54
#818,Blox Fruit But Bad,864,50,727,823,26,311,11,334,5,013,69.33
#919,The Robloxia: Until Dawn,761,5,712,164,51,865,14,691,2,724,84.36
#967,[🎃 🦇Halloween Part 1] Dragon Blox Ultimate,724,238,146,791,747,929,236,468,20,585,91.99
#980,[GOJO] Blox Fruits But Bad,715,14,539,699,9,958,4,800,1,726,73.55
#991,[🔥NEW!] FORTBLOX,707,715,978,2,517,972,183,84.16
```

Files & Libraries





```cpp
#include <iostream>
#include <fstream>
#include <sstream>
#include <vector>
#include <string>
#include <algorithm>
#include <iomanip> // for std::quoted
#include <set>

using namespace std;
```

Program Startup Flow

1. Program launches
2. User selects:

   > Login

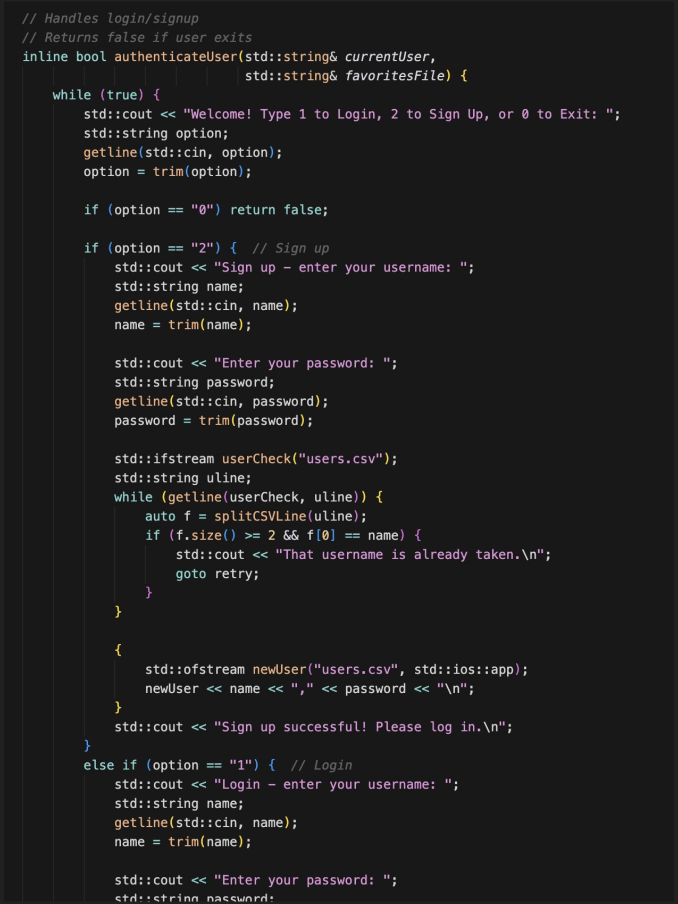
   > Sign up

   >Exit

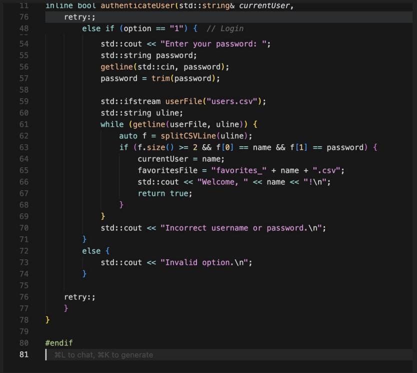
3. Credentials are validated using stored data
4. On successful login:

   > Personalized greeting is shown

   > User-specific favorites file is assigned



```cpp
// Handles login/signup
// Returns false if user exits
inline bool authenticateUser(std::string& currentUser,
                             std::string& favoritesFile) {
    while (true) {
        std::cout << "Welcome! Type 1 to Login, 2 to Sign Up, or 0 to Exit: ";
        std::string option;
        getline(std::cin, option);
        option = trim(option);

        if (option == "0") return false;

        if (option == "2") {  // Sign up
            std::cout << "Sign up - enter your username: ";
            std::string name;
            getline(std::cin, name);
            name = trim(name);

            std::cout << "Enter your password: ";
            std::string password;
            getline(std::cin, password);
            password = trim(password);

            std::ifstream userCheck("users.csv");
            std::string uline;
            while (getline(userCheck, uline)) {
                auto f = splitCSVLine(uline);
                if (f.size() >= 2 && f[0] == name) {
                    std::cout << "That username is already taken.\n";
                    goto retry;
                }
            }

            {
                std::ofstream newUser("users.csv", std::ios::app);
                newUser << name << "," << password << "\n";
            }
            std::cout << "Sign up successful! Please log in.\n";
        }
        else if (option == "1") {  // Login
            std::cout << "Login - enter your username: ";
            std::string name;
            getline(std::cin, name);
            name = trim(name);

            std::cout << "Enter your password: ";
            std::string password:
```



```cpp
inline bool authenticateUser(std::string& currentUser,
    retry:;
        else if (option == "1") {  // Login
            std::cout << "Enter your password: ";
            std::string password;
            getline(std::cin, password);
            password = trim(password);

            std::ifstream userFile("users.csv");
            std::string uline;
            while (getline(userFile, uline)) {
                auto f = splitCSVLine(uline);
                if (f.size() >= 2 && f[0] == name && f[1] == password) {
                    currentUser = name;
                    favoritesFile = "favorites_" + name + ".csv";
                    std::cout << "Welcome, " << name << "!\n";
                    return true;
                }
            }
            std::cout << "Incorrect username or password.\n";
        }
        else {
            std::cout << "Invalid option.\n";
        }

    retry:;
    }
}

#endif
```

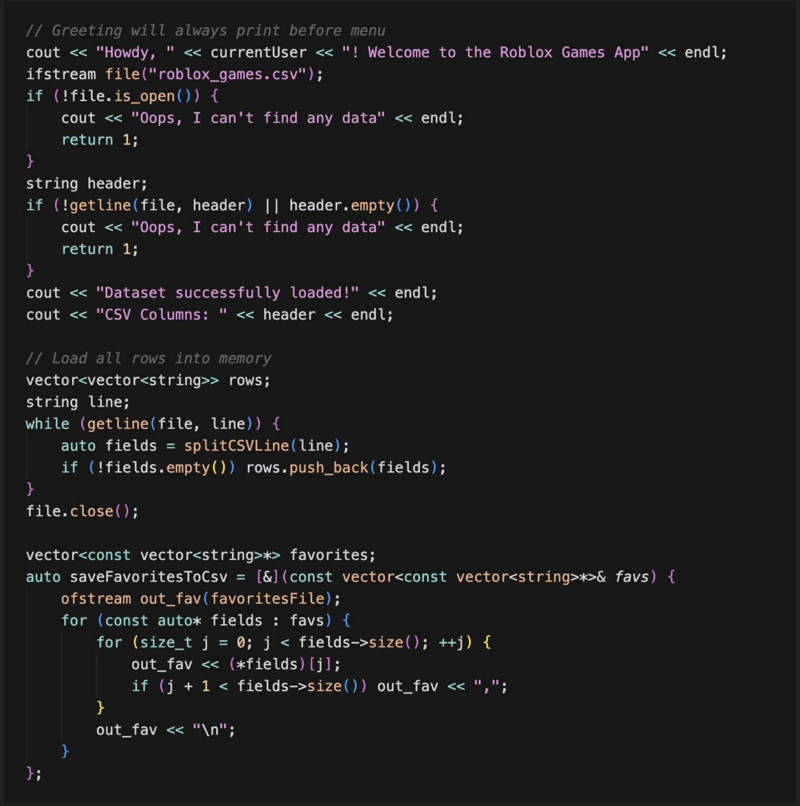
Dataset Loading

The program loads **`roblox_games.csv`** at startup

Steps:

- Opens the file using file streams
- Reads the header row
- Loads all remaining rows into memory

Each row represents a Roblox game
Data is stored in a vector for fast searching and filtering



```cpp
// Greeting will always print before menu
cout << "Howdy, " << currentUser << "! Welcome to the Roblox Games App" << endl;
ifstream file("roblox_games.csv");
if (!file.is_open()) {
    cout << "Oops, I can't find any data" << endl;
    return 1;
}
string header;
if (!getline(file, header) || header.empty()) {
    cout << "Oops, I can't find any data" << endl;
    return 1;
}
cout << "Dataset successfully loaded!" << endl;
cout << "CSV Columns: " << header << endl;

// Load all rows into memory
vector<vector<string>> rows;
string line;
while (getline(file, line)) {
    auto fields = splitCSVLine(line);
    if (!fields.empty()) rows.push_back(fields);
}
file.close();

vector<const vector<string>*> favorites;
auto saveFavoritesToCsv = [&](const vector<const vector<string>*>& favs) {
    ofstream out_fav(favoritesFile);
    for (const auto* fields : favs) {
        for (size_t j = 0; j < fields->size(); ++j) {
            out_fav << (*fields)[j];
            if (j + 1 < fields->size()) out_fav << ",";
        }
        out_fav << "\n";
    }
};
```

Data Structures Used

vector

- Represents a single row from the CSV file

vector<vector>

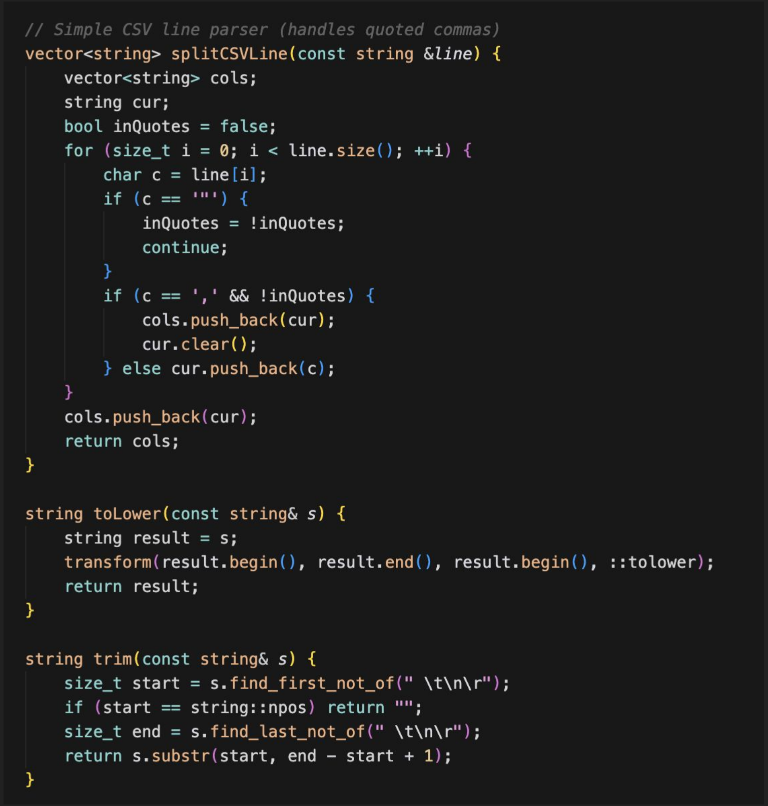
- Stores the complete dataset in memory

vector<const vector*>

- Efficiently tracks favorites without duplicating data

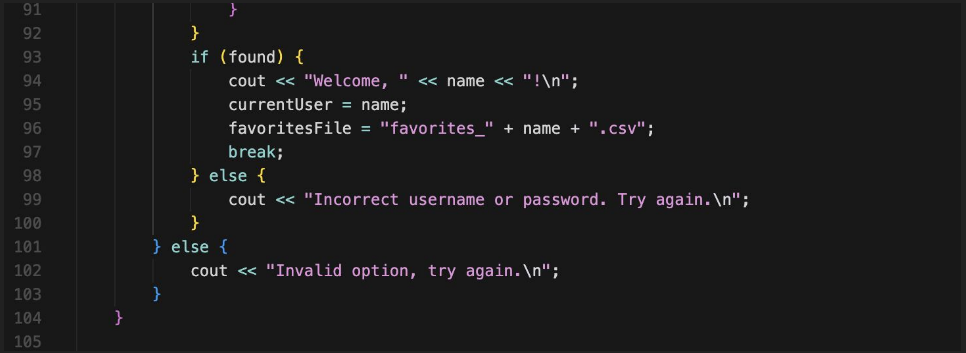
Strings and numeric conversions are used for flexible parsing

CSV Parsing & Text Processing



```cpp
// Simple CSV line parser (handles quoted commas)
vector<string> splitCSVLine(const string &line) {
    vector<string> cols;
    string cur;
    bool inQuotes = false;
    for (size_t i = 0; i < line.size(); ++i) {
        char c = line[i];
        if (c == '"') {
            inQuotes = !inQuotes;
            continue;
        }
        if (c == ',' && !inQuotes) {
            cols.push_back(cur);
            cur.clear();
        } else cur.push_back(c);
    }
    cols.push_back(cur);
    return cols;
}

string toLower(const string& s) {
    string result = s;
    transform(result.begin(), result.end(), result.begin(), ::tolower);
    return result;
}

string trim(const string& s) {
    size_t start = s.find_first_not_of(" \t\n\r");
    if (start == string::npos) return "";
    size_t end = s.find_last_not_of(" \t\n\r");
    return s.substr(start, end - start + 1);
}
```

- Custom CSV parser handles:
    - Quoted commas
    - Variable-length rows
- Helper functions:
    - `splitCSVLine()` – Parses CSV safely
    - `toLower()` – Case-insensitive searching
    - `trim()` – Cleans user input



```cpp
                }
            }
            if (found) {
                cout << "Welcome, " << name << "!\n";
                currentUser = name;
                favoritesFile = "favorites_" + name + ".csv";
                break;
            } else {
                cout << "Incorrect username or password. Try again.\n";
            }
        } else {
            cout << "Invalid option, try again.\n";
        }
    }

```

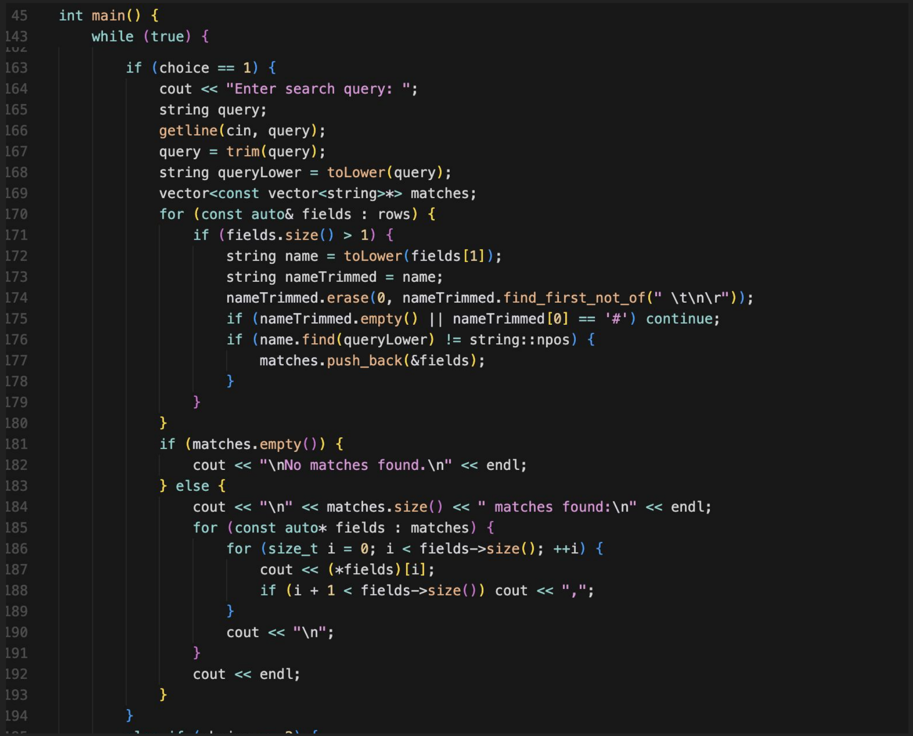
Search Functionality

Users can search games by name

The program:

- Converts both input and dataset to lowercase
- Supports partial matches
- Skips commented or invalid rows

Matching results are printed in full detail



```
int main() {
    while (true) {
        if (choice == 1) {
            cout << "Enter search query: ";
            string query;
            getline(cin, query);
            query = trim(query);
            string queryLower = toLower(query);
            vector<const vector<string>*> matches;
            for (const auto& fields : rows) {
                if (fields.size() > 1) {
                    string name = toLower(fields[1]);
                    string nameTrimmed = name;
                    nameTrimmed.erase(0, nameTrimmed.find_first_not_of(" \t\n\r"));
                    if (nameTrimmed.empty() || nameTrimmed[0] == '#') continue;
                    if (name.find(queryLower) != string::npos) {
                        matches.push_back(&fields);
                    }
                }
            }
            if (matches.empty()) {
                cout << "\nNo matches found.\n" << endl;
            } else {
                cout << "\n" << matches.size() << " matches found:\n" << endl;
                for (const auto* fields : matches) {
                    for (size_t i = 0; i < fields->size(); ++i) {
                        cout << (*fields)[i];
                        if (i + 1 < fields->size()) cout << ",";
                    }
                    cout << "\n";
                }
                cout << endl;
            }
        }
```

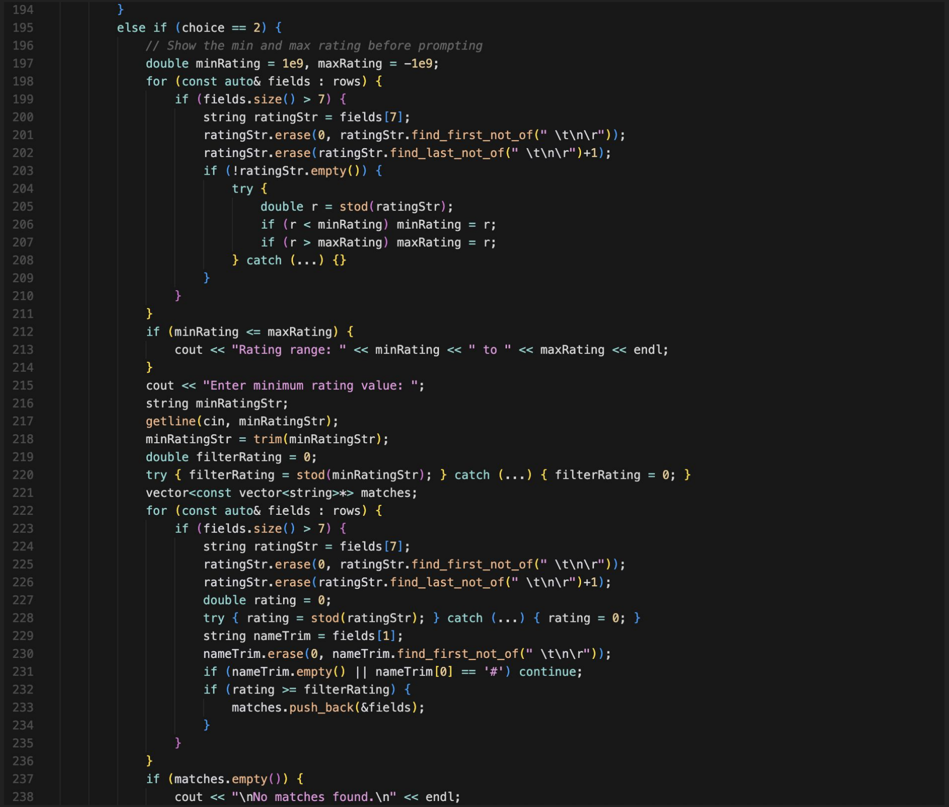
Filter By Rating

Users can filter games by minimum rating

The program:

- Scans the dataset to find valid rating ranges
- Converts rating strings to numbers
- Displays only games that meet the criteria

Improves user decision-making with numeric filtering



```cpp
        }
        else if (choice == 2) {
            // Show the min and max rating before prompting
            double minRating = 1e9, maxRating = -1e9;
            for (const auto& fields : rows) {
                if (fields.size() > 7) {
                    string ratingStr = fields[7];
                    ratingStr.erase(0, ratingStr.find_first_not_of(" \t\n\r"));
                    ratingStr.erase(ratingStr.find_last_not_of(" \t\n\r")+1);
                    if (!ratingStr.empty()) {
                        try {
                            double r = stod(ratingStr);
                            if (r < minRating) minRating = r;
                            if (r > maxRating) maxRating = r;
                        } catch (...) {}
                    }
                }
            }
            if (minRating <= maxRating) {
                cout << "Rating range: " << minRating << " to " << maxRating << endl;
            }
            cout << "Enter minimum rating value: ";
            string minRatingStr;
            getline(cin, minRatingStr);
            minRatingStr = trim(minRatingStr);
            double filterRating = 0;
            try { filterRating = stod(minRatingStr); } catch (...) { filterRating = 0; }
            vector<const vector<string>*> matches;
            for (const auto& fields : rows) {
                if (fields.size() > 7) {
                    string ratingStr = fields[7];
                    ratingStr.erase(0, ratingStr.find_first_not_of(" \t\n\r"));
                    ratingStr.erase(ratingStr.find_last_not_of(" \t\n\r")+1);
                    double rating = 0;
                    try { rating = stod(ratingStr); } catch (...) { rating = 0; }
                    string nameTrim = fields[1];
                    nameTrim.erase(0, nameTrim.find_first_not_of(" \t\n\r"));
                    if (nameTrim.empty() || nameTrim[0] == '#') continue;
                    if (rating >= filterRating) {
                        matches.push_back(&fields);
                    }
                }
            }
            if (matches.empty()) {
                cout << "\nNo matches found.\n" << endl;
```

Favorites System

Each user has a personalized favorites list

Favorites are:

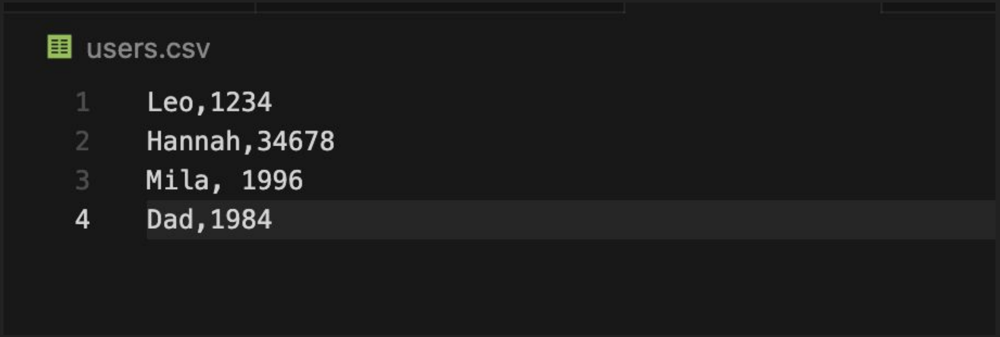
- Selected from search results
- Prevented from duplicating
- Stored both in memory and on disk

Long-Term Data Storage

- The program uses **persistent file storage**
- Stored files include:
  - `users.csv` – usernames and passwords
  - `favorites_<username>.csv` – per-user favorites

Benefits:

- Data remains saved even after the program exits
- Multiple users can have independent data

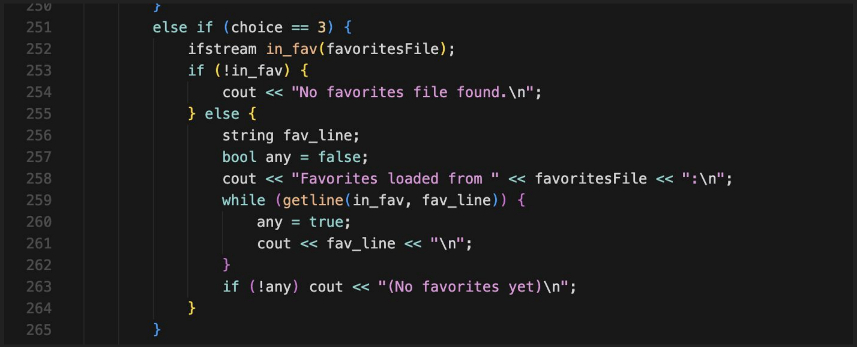

users.csv

```
Leo,1234
Hannah,34678
Mila, 1996
Dad,1984
```

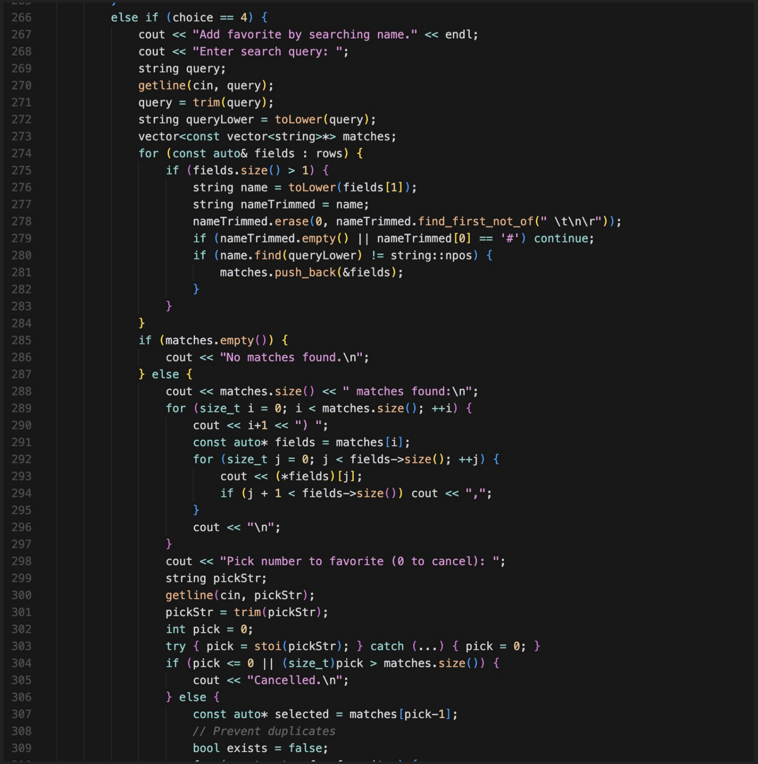
How Favorites Persistence Works

1. User adds or removes a favorite
2. Favorites are updated in memory
3. Program writes favorites to a CSV file immediately
4. On next program run:
    - The favorites file is reloaded
    - User data is restored

This ensures **long-term storage reliability**



```
        }
        else if (choice == 3) {
            ifstream in_fav(favoritesFile);
            if (!in_fav) {
                cout << "No favorites file found.\n";
            } else {
                string fav_line;
                bool any = false;
                cout << "Favorites loaded from " << favoritesFile << ":\n";
                while (getline(in_fav, fav_line)) {
                    any = true;
                    cout << fav_line << "\n";
                }
                if (!any) cout << "(No favorites yet)\n";
            }
        }
```



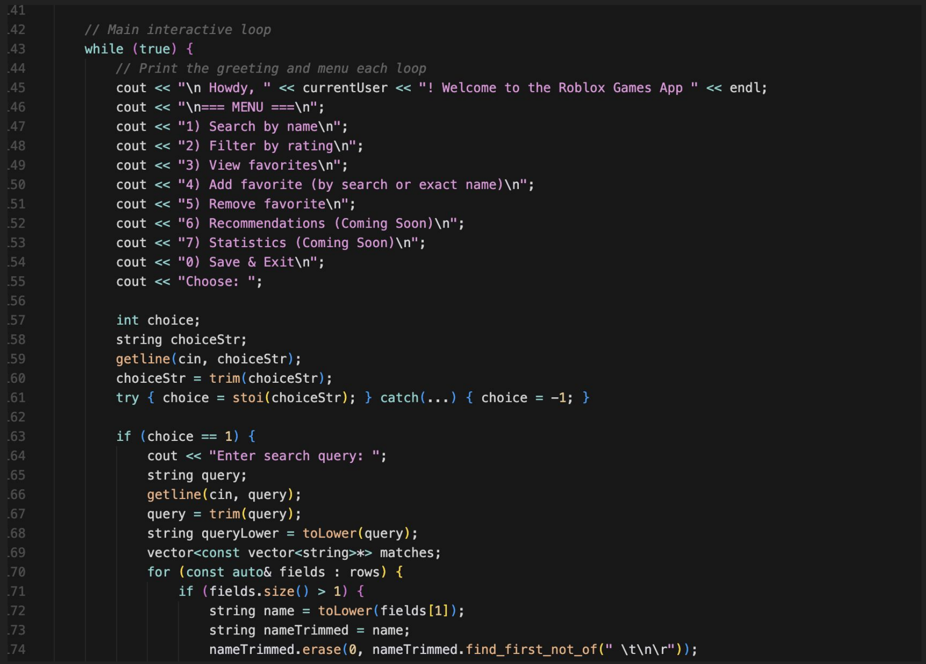
```
        else if (choice == 4) {
            cout << "Add favorite by searching name." << endl;
            cout << "Enter search query: ";
            string query;
            getline(cin, query);
            query = trim(query);
            string queryLower = toLower(query);
            vector<const vector<string>*> matches;
            for (const auto& fields : rows) {
                if (fields.size() > 1) {
                    string name = toLower(fields[1]);
                    string nameTrimmed = name;
                    nameTrimmed.erase(0, nameTrimmed.find_first_not_of(" \t\n\r"));
                    if (nameTrimmed.empty() || nameTrimmed[0] == '#') continue;
                    if (name.find(queryLower) != string::npos) {
                        matches.push_back(&fields);
                    }
                }
            }
            if (matches.empty()) {
                cout << "No matches found.\n";
            } else {
                cout << matches.size() << " matches found:\n";
                for (size_t i = 0; i < matches.size(); ++i) {
                    cout << i+1 << ") ";
                    const auto* fields = matches[i];
                    for (size_t j = 0; j < fields->size(); ++j) {
                        cout << (*fields)[j];
                        if (j + 1 < fields->size()) cout << ",";
                    }
                    cout << "\n";
                }
                cout << "Pick number to favorite (0 to cancel): ";
                string pickStr;
                getline(cin, pickStr);
                pickStr = trim(pickStr);
                int pick = 0;
                try { pick = stoi(pickStr); } catch (...) { pick = 0; }
                if (pick <= 0 || (size_t)pick > matches.size()) {
                    cout << "Cancelled.\n";
                } else {
                    const auto* selected = matches[pick-1];
                    // Prevent duplicates
                    bool exists = false;
```

File I/O Implementation

- Uses `ifstream` for reading
- Uses `ofstream` for writing
- Appends new users safely
- Overwrites favorites files to keep data consistent
- Prevents data corruption and duplication

Program Loop & Flow Control

- The app runs inside a main `while(true)` loop
- Menu is re-displayed after every action
- The program only exits when the user saves and chooses Exit
- Ensures continuous, responsive interaction

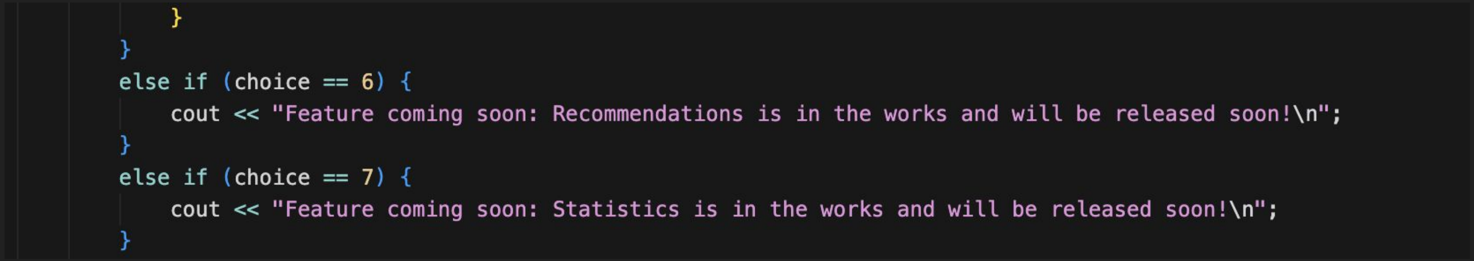


```cpp
// Main interactive loop
while (true) {
    // Print the greeting and menu each loop
    cout << "\n Howdy, " << currentUser << "! Welcome to the Roblox Games App " << endl;
    cout << "\n=== MENU ===\n";
    cout << "1) Search by name\n";
    cout << "2) Filter by rating\n";
    cout << "3) View favorites\n";
    cout << "4) Add favorite (by search or exact name)\n";
    cout << "5) Remove favorite\n";
    cout << "6) Recommendations (Coming Soon)\n";
    cout << "7) Statistics (Coming Soon)\n";
    cout << "0) Save & Exit\n";
    cout << "Choose: ";

    int choice;
    string choiceStr;
    getline(cin, choiceStr);
    choiceStr = trim(choiceStr);
    try { choice = stoi(choiceStr); } catch(...) { choice = -1; }

    if (choice == 1) {
        cout << "Enter search query: ";
        string query;
        getline(cin, query);
        query = trim(query);
        string queryLower = toLower(query);
        vector<const vector<string>*> matches;
        for (const auto& fields : rows) {
            if (fields.size() > 1) {
                string name = toLower(fields[1]);
                string nameTrimmed = name;
                nameTrimmed.erase(0, nameTrimmed.find_first_not_of(" \t\n\r"));
```

Future Features / Improvements

- Recommendation system
- Game statistics and analytics
- Improved password security (hashing)
- Sorting and ranking features
- Visual User Interface Design

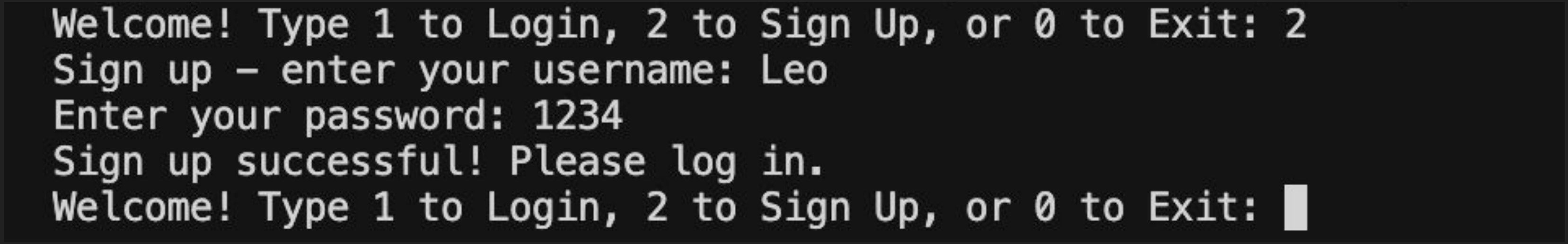
Design choices make these easy to add later



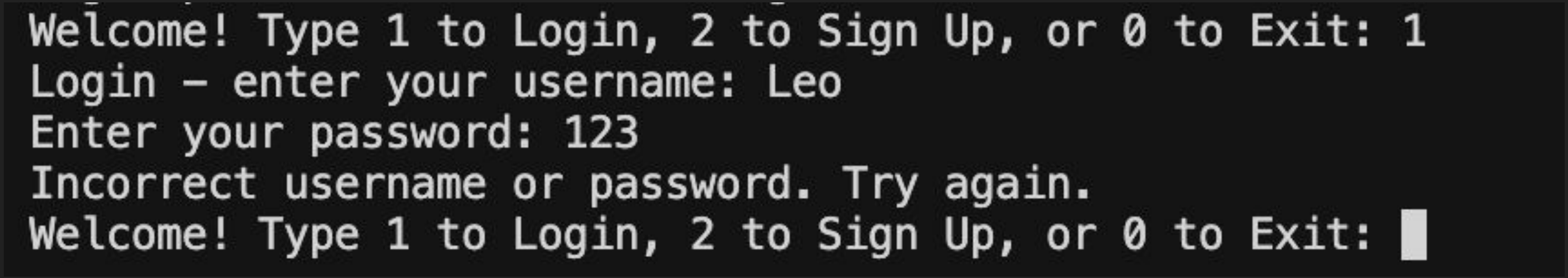
```
        }
    }
    else if (choice == 6) {
        cout << "Feature coming soon: Recommendations is in the works and will be released soon!\n";
    }
    else if (choice == 7) {
        cout << "Feature coming soon: Statistics is in the works and will be released soon!\n";
    }
```

DEMO :)

1) SignUp & Login

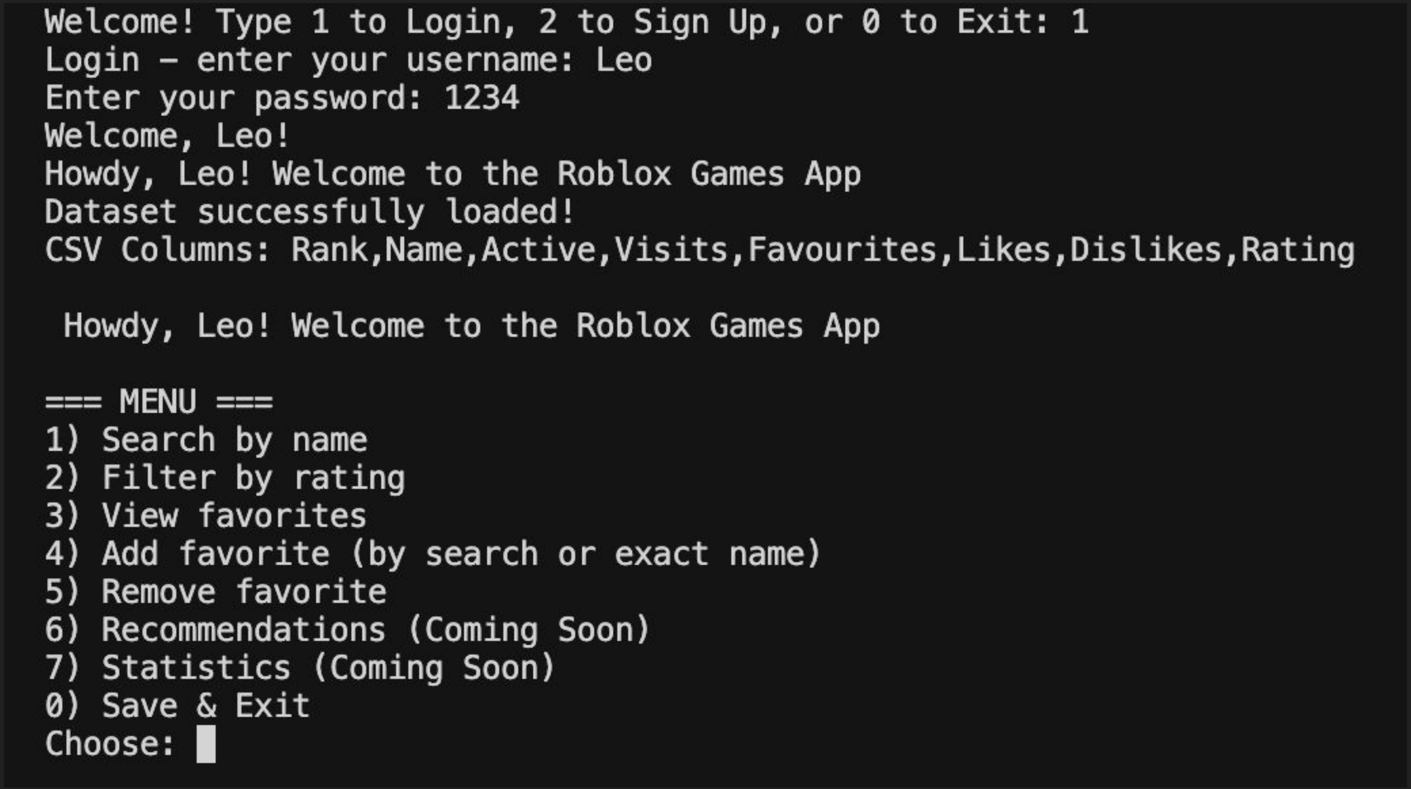


```
Welcome! Type 1 to Login, 2 to Sign Up, or 0 to Exit: 2
Sign up - enter your username: Leo
Enter your password: 1234
Sign up successful! Please log in.
Welcome! Type 1 to Login, 2 to Sign Up, or 0 to Exit:
```



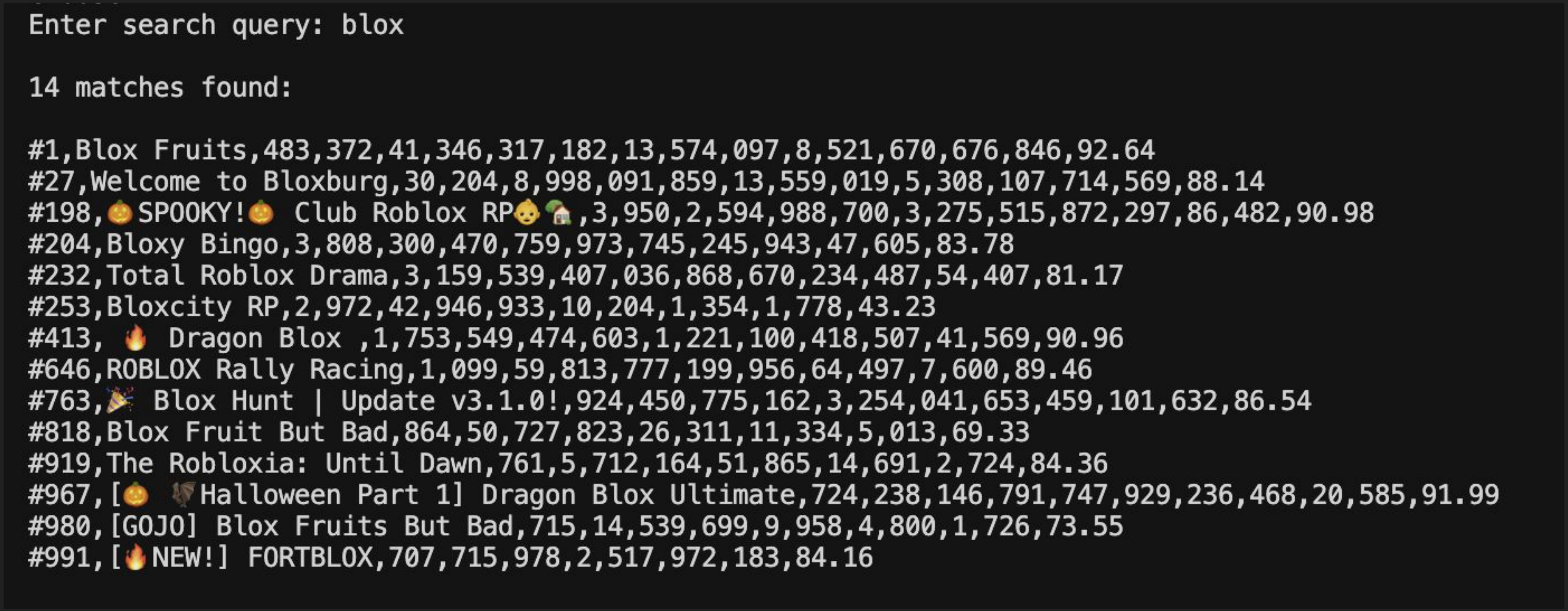
```
Welcome! Type 1 to Login, 2 to Sign Up, or 0 to Exit: 1
Login - enter your username: Leo
Enter your password: 123
Incorrect username or password. Try again.
Welcome! Type 1 to Login, 2 to Sign Up, or 0 to Exit:
```

2) Menu Options



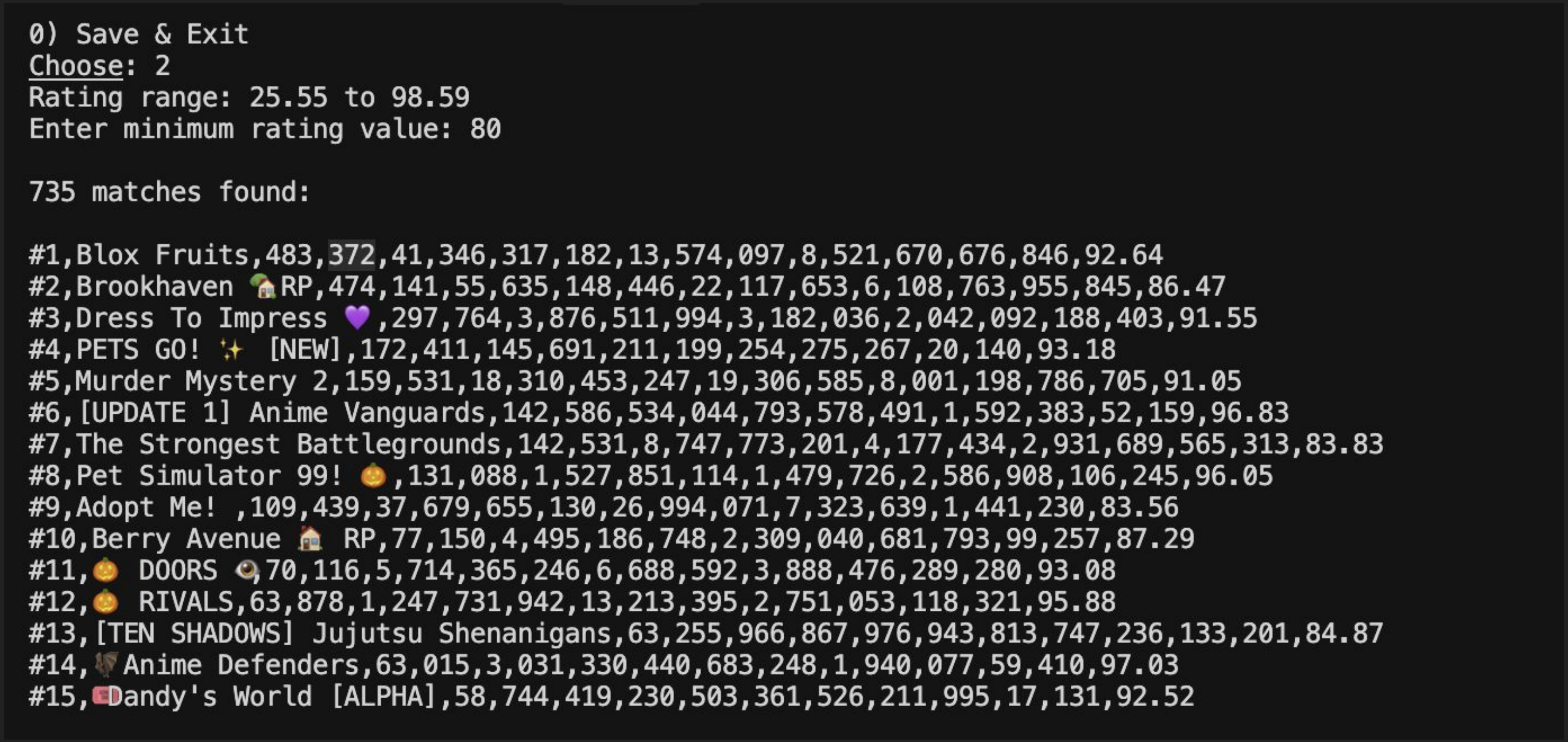
```
Welcome! Type 1 to Login, 2 to Sign Up, or 0 to Exit: 1
Login - enter your username: Leo
Enter your password: 1234
Welcome, Leo!
Howdy, Leo! Welcome to the Roblox Games App
Dataset successfully loaded!
CSV Columns: Rank,Name,Active,Visits,Favourites,Likes,Dislikes,Rating

 Howdy, Leo! Welcome to the Roblox Games App

=== MENU ===
1) Search by name
2) Filter by rating
3) View favorites
4) Add favorite (by search or exact name)
5) Remove favorite
6) Recommendations (Coming Soon)
7) Statistics (Coming Soon)
0) Save & Exit
Choose: █
```

3) Search Games by Name



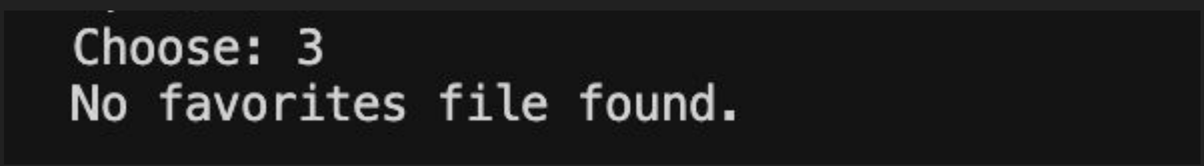
```
Enter search query: blox

14 matches found:

#1,Blox Fruits,483,372,41,346,317,182,13,574,097,8,521,670,676,846,92.64
#27,Welcome to Bloxburg,30,204,8,998,091,859,13,559,019,5,308,107,714,569,88.14
#198,🎃SPOOKY!🎃 Club Roblox RP👶🏡,3,950,2,594,988,700,3,275,515,872,297,86,482,90.98
#204,Bloxy Bingo,3,808,300,470,759,973,745,245,943,47,605,83.78
#232,Total Roblox Drama,3,159,539,407,036,868,670,234,487,54,407,81.17
#253,Bloxcity RP,2,972,42,946,933,10,204,1,354,1,778,43.23
#413, 🔥 Dragon Blox ,1,753,549,474,603,1,221,100,418,507,41,569,90.96
#646,ROBLOX Rally Racing,1,099,59,813,777,199,956,64,497,7,600,89.46
#763,🎉 Blox Hunt | Update v3.1.0!,924,450,775,162,3,254,041,653,459,101,632,86.54
#818,Blox Fruit But Bad,864,50,727,823,26,311,11,334,5,013,69.33
#919,The Robloxia: Until Dawn,761,5,712,164,51,865,14,691,2,724,84.36
#967,[🎃 🦇Halloween Part 1] Dragon Blox Ultimate,724,238,146,791,747,929,236,468,20,585,91.99
#980,[GOJO] Blox Fruits But Bad,715,14,539,699,9,958,4,800,1,726,73.55
#991,[🔥NEW!] FORTBLOX,707,715,978,2,517,972,183,84.16
```

4) Filter Games by Minimum Rating

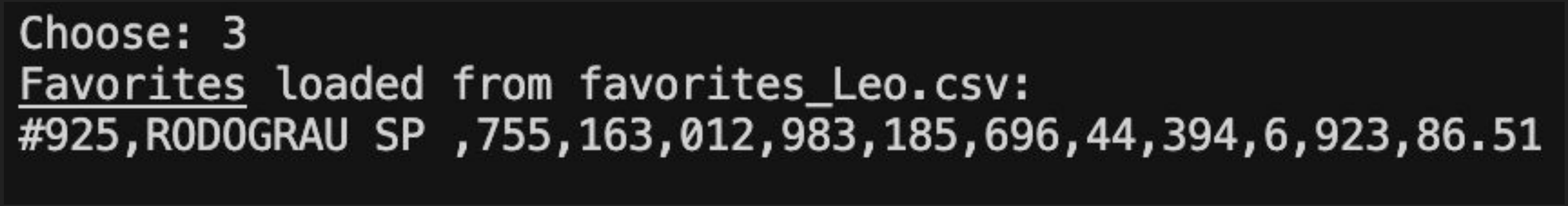


```
0) Save & Exit
Choose: 2
Rating range: 25.55 to 98.59
Enter minimum rating value: 80

735 matches found:

#1,Blox Fruits,483,372,41,346,317,182,13,574,097,8,521,670,676,846,92.64
#2,Brookhaven 🏡RP,474,141,55,635,148,446,22,117,653,6,108,763,955,845,86.47
#3,Dress To Impress 💜,297,764,3,876,511,994,3,182,036,2,042,092,188,403,91.55
#4,PETS GO! ✨ [NEW],172,411,145,691,211,199,254,275,267,20,140,93.18
#5,Murder Mystery 2,159,531,18,310,453,247,19,306,585,8,001,198,786,705,91.05
#6,[UPDATE 1] Anime Vanguards,142,586,534,044,793,578,491,1,592,383,52,159,96.83
#7,The Strongest Battlegrounds,142,531,8,747,773,201,4,177,434,2,931,689,565,313,83.83
#8,Pet Simulator 99! 🎃,131,088,1,527,851,114,1,479,726,2,586,908,106,245,96.05
#9,Adopt Me! ,109,439,37,679,655,130,26,994,071,7,323,639,1,441,230,83.56
#10,Berry Avenue 🏠 RP,77,150,4,495,186,748,2,309,040,681,793,99,257,87.29
#11,🎃 DOORS 👁️,70,116,5,714,365,246,6,688,592,3,888,476,289,280,93.08
#12,🎃 RIVALS,63,878,1,247,731,942,13,213,395,2,751,053,118,321,95.88
#13,[TEN SHADOWS] Jujutsu Shenanigans,63,255,966,867,976,943,813,747,236,133,201,84.87
#14,🦇Anime Defenders,63,015,3,031,330,440,683,248,1,940,077,59,410,97.03
#15,🎟️Dandy's World [ALPHA],58,744,419,230,503,361,526,211,995,17,131,92.52
```

5) Load & Display Saved Favorites

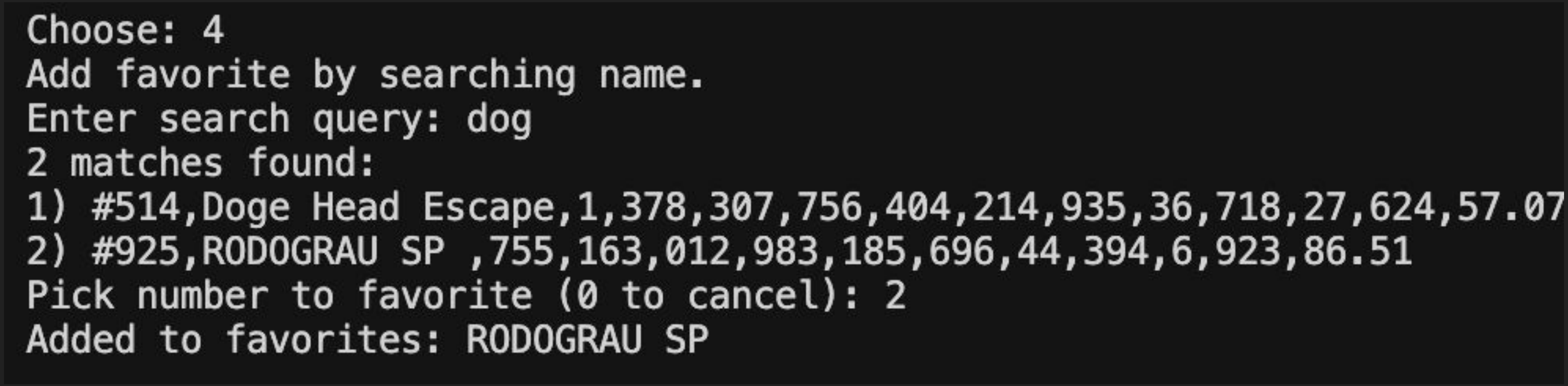


```
Choose: 3
No favorites file found.
```



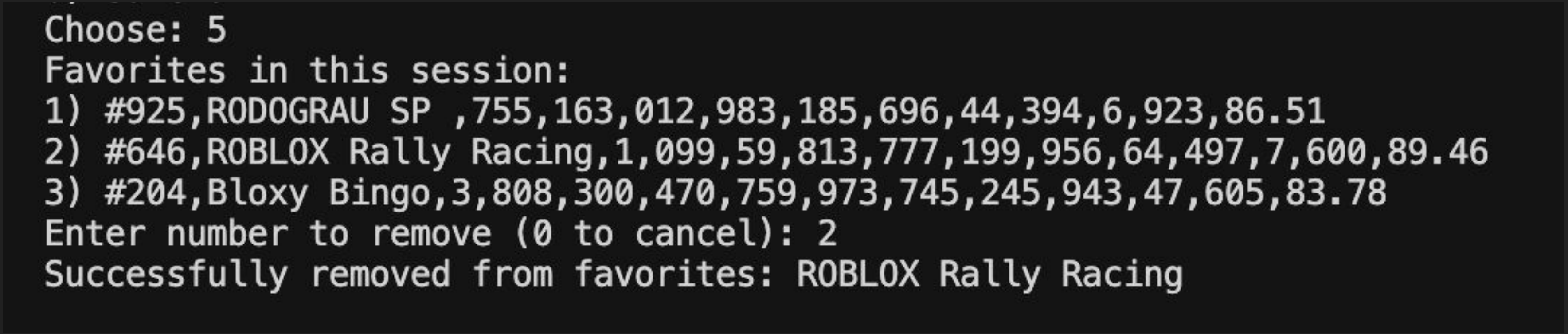
```
Choose: 3
Favorites loaded from favorites_Leo.csv:
#925,RODOGRAU SP ,755,163,012,983,185,696,44,394,6,923,86.51
```

6) Add New Favorite



```
Choose: 4
Add favorite by searching name.
Enter search query: dog
2 matches found:
1) #514,Doge Head Escape,1,378,307,756,404,214,935,36,718,27,624,57.07
2) #925,RODOGRAU SP ,755,163,012,983,185,696,44,394,6,923,86.51
Pick number to favorite (0 to cancel): 2
Added to favorites: RODOGRAU SP
```

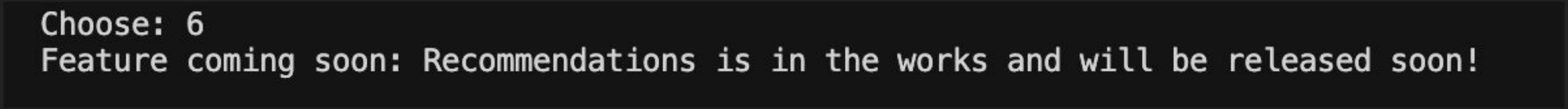
7) Remove Favorite



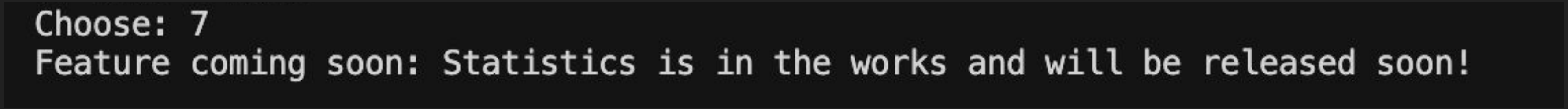
```
Choose: 5
Favorites in this session:
1) #925,RODOGRAU SP ,755,163,012,983,185,696,44,394,6,923,86.51
2) #646,ROBLOX Rally Racing,1,099,59,813,777,199,956,64,497,7,600,89.46
3) #204,Bloxy Bingo,3,808,300,470,759,973,745,245,943,47,605,83.78
Enter number to remove (0 to cancel): 2
Successfully removed from favorites: ROBLOX Rally Racing
```

8) Future Features

Friendly message about future availability



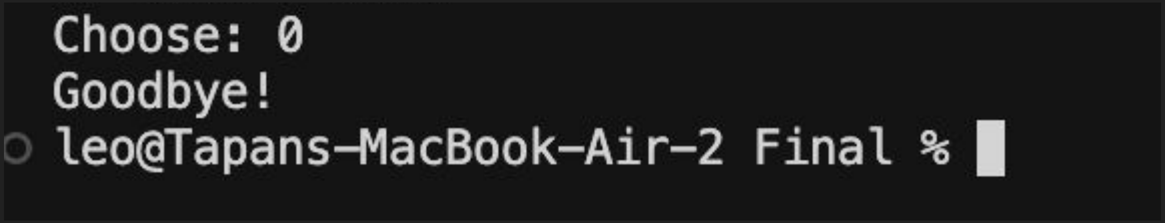
```
Choose: 6
Feature coming soon: Recommendations is in the works and will be released soon!
```



```
Choose: 7
Feature coming soon: Statistics is in the works and will be released soon!
```

9) Save & Exit

Saves and says goodbye, ending while-loop



```
Choose: 0
Goodbye!
leo@Tapans-MacBook-Air-2 Final %
```