Program Documentation: Real Estate Listing System

Task Allocation: Real Estate Listing Program (C)

Naimaer Vlad is responsible for:

- Managing and displaying available listings (from `listings.txt`).
- Implementing the "Buy" functionality:
  - Selecting a listing.
  - Moving the purchased listing from `listings.txt` to `history.txt`.
  - Updating `listings.txt`.
- Displaying the purchase history (from `history.txt`).

Tomescu Robert is responsible for:

- Developing the main menu structure and program navigation flow (`header()`, `menu()`, `main()` loop).
- Implementing the "Add to Favourites" functionality:
  - Selecting a listing.
  - Moving the favourited listing from `listings.txt` to `fav.txt`.
  - Updating `listings.txt`.
- Displaying the favourites list (from `fav.txt`).

I. System Overview

This system allows users to manage and interact with real estate listings. Users can view available listings, add them to a personal favourites list, mark them as purchased (moving them to a purchase history), and view their favourites and purchase history. The system operates through a command-line menu interface and uses text files for data persistence.

II. Core Data Representation (Conceptual C++ Adaptation)

While the current C code uses direct file I/O with string manipulation, a C++ adaptation would likely use classes to represent the core entities.

- **`Listing` Class (Conceptual)**
  - Attributes:
    - `int orderNumber` (e.g., "1.", "2.")
    - `std::string type` (e.g., "House", "Apartment")
    - `std::string price` (e.g., "145000$")
    - `std::string size` (e.g., "85sq")
    - `std::string rooms` (e.g., "3 rooms")

        - `std::string bathrooms` (e.g., "1 bathroom")
        - `std::string parking` (e.g., "parking space included", "no parking space")
    - Responsibilities: Store all details pertaining to a single real estate listing. Methods could include parsing a string to populate attributes and formatting attributes back into a string for display or file storage.

III. File Structure and Data Persistence

The system utilizes several text files to store its data:

- **`listings.txt`**
    - Purpose: Stores details about currently available real estate listings.
    - Format: Each line represents a listing with fields separated by commas, including order number, type, price, size, number of rooms, bathrooms, and parking details.
    - Example line: `4.Apartment, 99000$, 60sq, 2 rooms, 1 bathroom, no parking space.`
- **`fav.txt`**
    - Purpose: Stores listings that the user has marked as favourites.
    - Format: Same as `listings.txt`.
    - Example line: `2.Apartment, 145000$, 85sq, 3 rooms, 1 bathroom, parking space included.`
- **`history.txt`**
    - Purpose: Stores listings that the user has marked as purchased.
    - Format: Same as `listings.txt`.
    - Example line: `1.House, 199999$, 120sq, 4 rooms, 2 bathrooms, no parking space.`
- **`format.txt`** (Documentation File)
    - Purpose: Describes the expected format of a listing line.
- **`temp.txt`** (Temporary File)
    - Purpose: Used internally during file manipulation (e.g., when removing a listing from `listings.txt` after it's added to favourites or purchased).

IV. Application Functionality (Single Executable)

The application provides a menu-driven interface to interact with the listings.

- **Main Menu (`header()` function)**
    - Displays options:
        - 1 - Listings
        - 2 - Favourites
        - 3 - Purchase history
        - 0 - Exit
- **1. Listings Management (`list()` function)**
    - **View Listings:**

        - Reads and displays all entries from `listings.txt`.
    - **User Actions within Listings:**
        - **Add to Favourites:**
            - Prompts the user for the order number of a listing.
            - Reads `listings.txt` to find the specified listing line.
            - Appends the found listing line to `fav.txt`.
            - Removes the listing from `listings.txt` (by copying other listings to `temp.txt`, then replacing `listings.txt` with `temp.txt`).
        - **Buy (Mark as Purchased):**
            - Prompts the user for the order number of a listing.
            - Reads `listings.txt` to find the specified listing line.
            - Appends the found listing line to `history.txt`.
            - Removes the listing from `listings.txt` (using the same temporary file method as above).
        - **Back:** Returns to the Main Menu.
- **2. Favourites Management (`fav()` function)**
    - **View Favourites:**
        - Reads and displays all entries from `fav.txt`.
    - **User Actions within Favourites:**
        - **Back:** Returns to the Main Menu.
- **3. Purchase History Management (`history()` function)**
    - **View Purchase History:**
        - Reads and displays all entries from `history.txt`.
    - **User Actions within Purchase History:**
        - **Back:** Returns to the Main Menu.
- **Navigation (`menu()` function and `main()` loop)**
    - The `main()` function repeatedly displays the header and gets user input.
    - The `menu()` function uses a `switch` statement to call the appropriate function (`list()`, `fav()`, `history()`) based on user input.
    - The "Back" functionality (option '4') in sub-menus is intended to return the user to the main menu display.
    - `system("clear")` is used to clear the screen for better readability between menu transitions.

V. Program Flow

1. The `main()` function starts a loop.
2. Inside the loop, `header()` displays the main menu options.
3. The user is prompted to `Enter option:`.
4. `system("clear")` clears the console.
5. `menu(option)` is called, which directs to `list()`, `fav()`, or `history()` based on the input.
6. Each of these sub-functions displays its specific content and then presents options, including an option to go back (which, due to the `menu()` structure, effectively re-

triggers `header()` if '4' is chosen within `list()`, or simply allows the main loop to reiterate).

7. The loop in `main()` continues as long as the entered option is between 1 and 4 (inclusive of the "Back" option which might lead to re-displaying the main menu or specific sub-menu logic). Option 0 exits.