Library Management System - README

This is a Windows-based Library Management System developed using C++ and Qt Widgets, designed to manage books and magazines with features like adding, searching, borrowing, returning, and data persistence. This project is a continuation of Assignment 2 for COS2614 - Programming Contemporary Concepts at UNISA.

Project Structure

- `LibraryManagementSystem_part2/`
  - `LibraryManagementSystem/`: Main application directory containing source files, UI forms, and `LibraryManagementSystem.pro`.
  - `LibraryUtils/`: Static library project containing source files and `LibraryUtils.pro`.

Setup Instructions

1. **Install Qt**:
   a. Download and install Qt from qt.io with the MinGW 32-bit compiler.
   b. Set up Qt Creator with the installed Qt version.
2. **Extract the Project**:
   a. Unzip the submitted `.zip` file to a directory (e.g., `C:\Users\Marker\Desktop\68879245_COS2614_assignment3_v2\`).
   b. Ensure the folder structure preserves `LibraryManagementSystem` and `LibraryUtils` as sibling directories under `LibraryManagementSystem_part2`.
3. **Configure the Project**:
   a. Open Qt Creator.
   b. Go to `File` > `Open File or Project` and select `LibraryManagementSystem_part2/LibraryManagementSystem/LibraryManagementSystem.pro`.
   c. In the "Configure Project" dialog, select the appropriate Qt kit (e.g., Desktop Qt 5.3.0 MinGW 32-bit).
   d. **Update Build and Run Settings**:

i. In the "Projects" panel, under "Build & Run" for `LibraryManagementSystem`:
    1. Set the "Build Directory" to a custom path (e.g., `build-LibraryManagementSystem-Desktop_Qt_5_3_0_MinGW_32bit-Release`) within the extracted directory structure.
    2. Set the "Working Directory" under "Run Settings" to the same build directory.
ii. Repeat for `LibraryUtils` by opening `LibraryManagementSystem_part2/LibraryUtils/LibraryUtils.pro` and setting the "Build Directory" to a sibling path (e.g., `build-LibraryUtils-Desktop_Qt_5_3_0_MinGW_32bit-Release`).

e. **Build LibraryUtils First**: Open `LibraryManagementSystem_part2/LibraryUtils/LibraryUtils.pro`, set to "Release" mode, and build the project (`Build` > `Build Project "LibraryUtils"`) to generate the static library. Note the actual build directory (e.g., `build-LibraryUtils-...`).
f. Adjust the `LIBS` path in `LibraryManagementSystem.pro` if needed: Edit the line under `LIBS` to match the `LibraryUtils` build directory (e.g., replace `Desktop_Qt_5_3_0_MinGW_32bit` with the marker's version if different, such as `Desktop_Qt_5_12_0_MinGW_32bit`).

4. **Build the Main Application**:
    a. Return to the `LibraryManagementSystem` project in Qt Creator.
    b. Set the build configuration to "Release".
    c. Clean and build the project (`Build` > `Clean All`, then `Build` > `Build Project "LibraryManagementSystem"`).
    d. Check the "Compile Output" pane for any errors. If a "No rule to make target" error occurs, verify the "Build Directory" settings and ensure `LibraryUtils` is built first.
5. **Run the Program**:
    a. Click the "Run" button (green play icon) in Qt Creator.
    b. Alternatively, navigate to the build directory (e.g., `build-LibraryManagementSystem-Desktop_Qt_5_3_0_MinGW_32bit-Release`) and double-click the executable (`LibraryManagementSystem.exe`), ensuring `library_data.txt` is in the same directory.

Usage Instructions

- **Add a Book**:
  - Enter Title, Author, ID, and Genre in the respective fields.
  - Click "Add Book" to add it to the list and save to `library_data.txt`.
- **Add a Magazine**:
  - Enter Title, Author, ID, and Issue Number in the respective fields.
  - Click "Add Magazine" to add it to the list and save.
- **Search**:
  - Type a partial title in the "Title Book" field and click "Search" to filter the list.
  - Clear the field and click "Search" to show all items.
- **Borrow/Return**:
  - Select an item from the list and click "Borrow/Return" to toggle its status.
  - The status updates and is saved to `library_data.txt`.
- **Exit**:
  - Close the application to save the current list to `library_data.txt`.

Notes

- The program creates `library_data.txt` in the build directory if it doesn't exist.
- Ensure the "Build Directory" and "Working Directory" in Qt Creator's "Projects" settings match the new file path to avoid build errors.
- The code is documented with comments for clarity and compiles without errors in Qt Creator.

Troubleshooting

- **Build Errors**: Verify the "Projects" panel build directory settings match the new file path. Rebuild `LibraryUtils` first. If "No rule to make target" occurs, ensure `LibraryUtils` is built separately and the `LIBS` path in `LibraryManagementSystem.pro` is correct.
- **Missing Library**: Check the `LIBS` path in `LibraryManagementSystem.pro` and rebuild `LibraryUtils` with a matching build directory structure. Adjust the path if the build directory name differs (e.g., due to a different Qt version).
- **File Not Found**: Ensure `library_data.txt` is in the working directory or create an empty file manually.