Sawyer Stanley
0289276
COMP 4911
Bookdex: Project Proposal
February 28, 2024

Overview

Bookdex is an application that allows a client to remotely query a database of select books. A home library will then be accessible from a mobile device. Thus, next time one is browsing a book retailer's shelves, one can use Bookdex to confirm that a desirable book is or is not currently owned ensuring that unnecessary copies are not purchased. Note, the contents of the books are not accessible. Bookdex is a tool for browsing titles and confirming existence.

To make Bookdex a reality, a database server within which to store books will be required. Each user will have an SQLite database on the C# server. The server will be a console application, avoiding unnecessary work on GUI design. When the server recieves a query, it will attempt to pose that query to the database and return the results to the client. Creation and interactions with the database will be streamlined using packages sqlite-net-pcl[3] and SQLitePCLRaw.bundle_green[4]. These packages contain useful attributes for defining relations within C# classes and useful methods for querying an SQL database.

The server will contain an additional SQLite database which will store client login information. This database is essential for not only login purposes, but for ensuring a client gets access to only their database of books. This is because database file names will correspond with a client's distinct username.

The client side of Bookdex will be realized through a .NET MAUI application[1], allowing Bookdex to be easily enjoyed on multiple platforms. Unlike the server, the client will not be a console application but will consist of a simple GUI. This way, a user can intuitively create and send their queries to the server. If the client successfully logs in or creates an account, they will be prompted with a text field within which they can enter a book title or author to send to the server. Database entries that match the query will then be returned to the client and presented as a list. This is the simplest interaction that must be implemented. Additional quality of life features will be covered later.

When researching protocols for querying database servers, one of the more promising results was the tabular data stream protocol (TDS)[2]. Upon looking closer at TDS specifications, only a handful of messages need be supported to make Bookdex functional. On the client side, these messages include

(1) Pre-Login As the name suggests, this message is sent by the client prior to the login message as a "handshake" to set up encryption.

(2) Login Authentication is performed at this step. The client, requesting to make a TDS connection with the server, sends their credentials to be accepted or rejected from the server.

(3) SQL Batch This type of message contians SQL commands to be processed by the server.

(4) Bulk Load While the SQL Batch message is similar to writing an SQL command to *retrieve* data from a database, the Bulk Load message is akin to a bulk *insert* into the database.

(5) Attention This message is sent when the client wishes to interrupt and cancel the current request.

Server side messages include

(1) Pre-Login Response This message is a response to the pre-login message from the client to complete the “handshake” and setting up encryption.

(2) Login Response Having recieved a login message from the client, the server responds with a login response message containing server information.

(3) Row Data A mesage describing row and column (despite the name) information requested by the client. This is similar to a relation being returned after an SQL command.

(4) Response Completion Indicates whether there are more results to still be returned by the server to the client.

(5) Error and Info The naming of these message types are exactly as expected. Error messages are sent to the client when an error occurs, and info messages provide additional information.

(6) Attention Acknowledgment When the server recieves an attention message from the client, an attention acknowledgment is sent in response. The client must read returned data until this message is recieved.

By implementing this subset of messages of TDS, Bookdex can be realized.

Additional Ideas

While my priority will be the successful implementation of part of TDS demonstrated through a minimalistic Bookdex, there are a handful of other ideas I would be interested in attempting if time permits.

In the base application, Bookdex will permit users to search their library by title and/or author. This will make the construction of the SQL queries simple. Adding additional features to how a client can search (filtering results by genre, publisher, etc.) will improve the overall quality of life.

Bookdex users will have access to their own library. It would be interesting to allow users to share their libraries with others, providing other users with read-only access. I imagine this being useful in the scenario where Alice had read most of her books and wants inspiration on what to read next from Bob. So, Bob gives Alice read-only access to his library.

Because I imagine Bookdex’s main purpose being checking the existence of books within one’s library, I’m interested in attempting some form of results caching. Perhaps a user is preparing to browse through some potentially new books in a location where they will not have access to the internet. Storing some search results locally prior to being disconnected to the internet would allow them to continue using the application. This is similar to how Google Maps continues to provide directions so long as you’ve planned the route prior to leaving.

References

[1] *.NET Multi-platform App UI Documentation*. https://learn.microsoft.com/en-us/dotnet/maui/?view=net-maui-8.0. Accessed: 2024-2-28.

[2] *[MS-TDS]: Tabular Data Stream Protocol*. https://learn.microsoft.com/en-us/openspecs/windows_protocols/ms-tds/b46a581a-39de-4745-b076-ec4dbb7d13ec. Accessed: 2024-2-28.

[3] *sqlite-net-pcl*. https://www.nuget.org/packages/sqlite-net-pcl/. Accessed: 2024-2-28.

[4] *SQLitePCLRaw.bundle.green*. https://www.nuget.org/packages/SQLitePCLRaw.bundle_green/. Accessed: 2024-2-28.