Adam Klein, Mark Macki, Alex Morton

ACS 560

Book Tracker

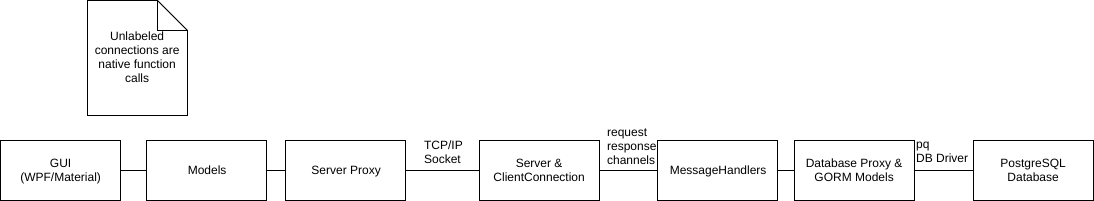
Follows is a design document for the Book Tracker application. Diagrams in the document are listed below.

* Modules Diagram
* Use case Diagram
* Client Diagram
* Server Models
* Server Connections
* Messaging Diagram
* Creating Client Connection
* Client Request Routing
* Add Book Sequence

Note: For better readability all diagrams are also available as individual files in the Design folder.

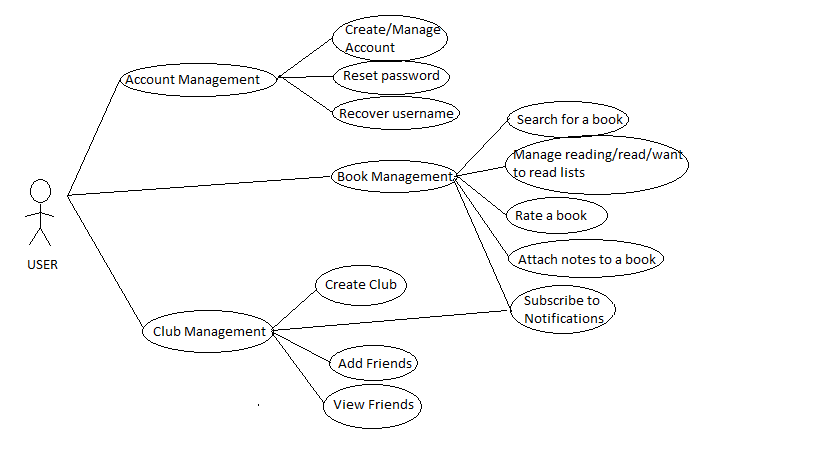
Modules:

An overview of the application subdivided into layers. Represents each piece of the client-server architecture and the connections between them.



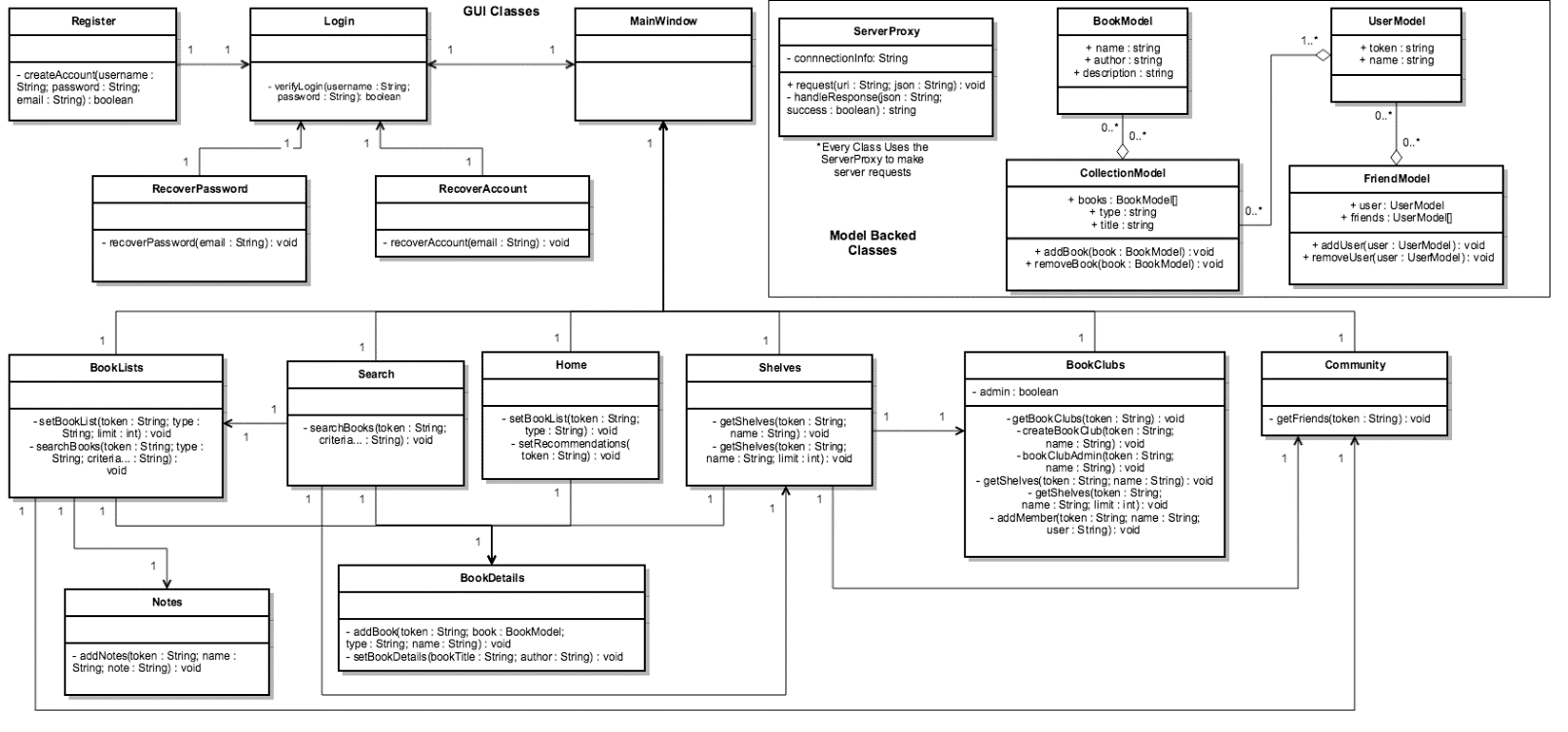
Use Case:

The user at the GUI will have 3 main functions: account management, book management and club management. The use case lists those functions and further breaks them down based on sub-functions.



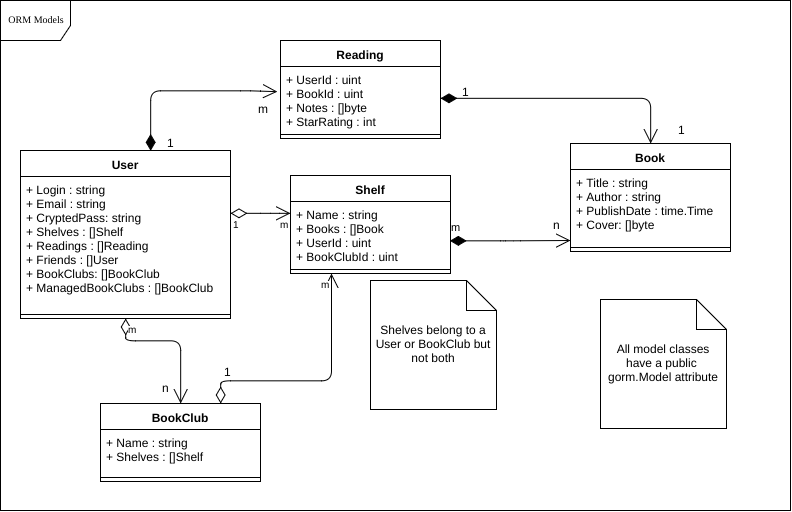
Client Diagram:

The Client UML diagram represents the classes in C# for the client. It is separated into two parts: the classes for the GUI and the model classes. The model classes represent books, users, and collections of both of them. There is also a ServerProxy class that manages requests and responses to the server.



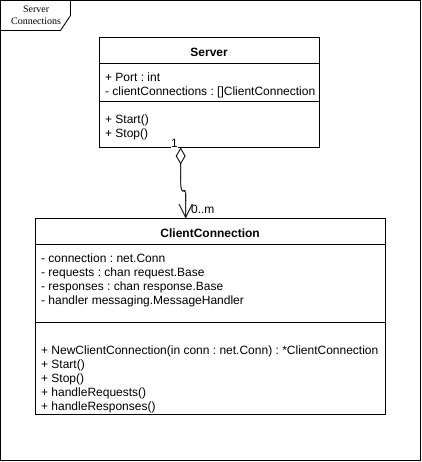
Server Models:

This diagram represents the tables in the PostgreSQL database. This includes a table to for a user, the reading lists for a user, book clubs the user is a member or a manager of, shelves for users and book clubs, and books in Reading lists and Shelves.



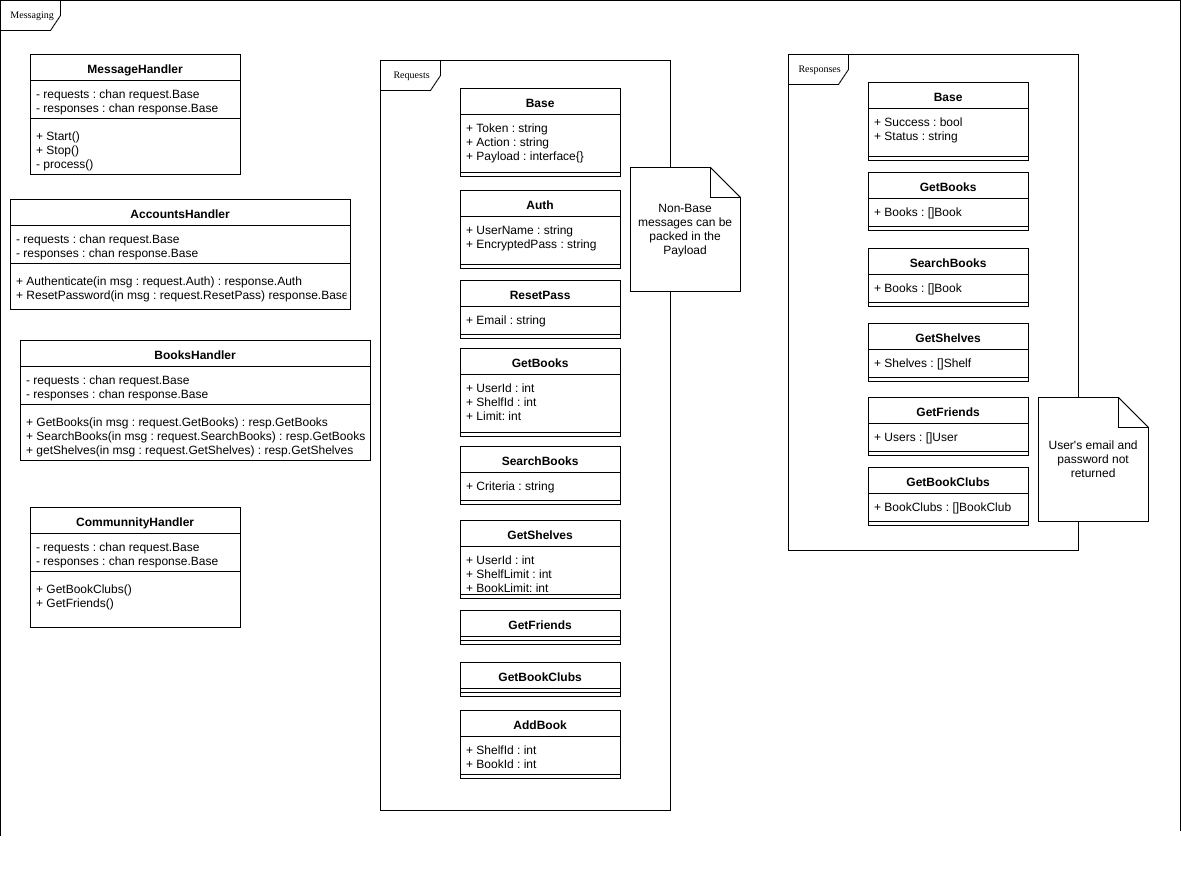
Server Connection:

A UML Class diagram of the basic components in client connections to the server. The Server listens on the base socket port. For each new connection, a ClientConnection is created and started. ClientConnections handle individual requests and send responses on the TCP connection.



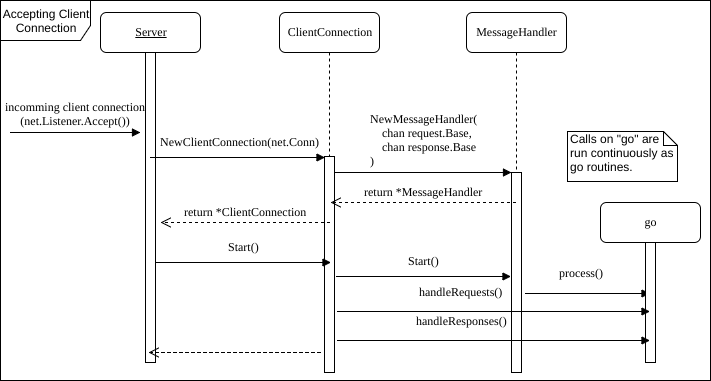
Messaging:

This includes the basic messaging components. ClientConnections each have a MessageHandler who parses messages via JSON off the underlying TCP connection. That JSON is then converted into a request.Base message. The Base message's payload is then converted into a more specific message type and sent to a more specific message handler. The message is processed and a response message is returned.



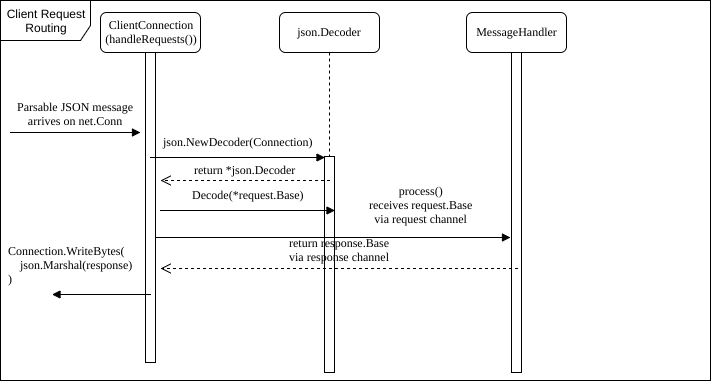
Creating Client Connection:

Sequence diagram showing the steps to handling a client connection. Shows a sequence of events during a new client connection. ClientConnections are constructed and started. This in turn creates and starts a corresponding MessageHandler. ClientConnections and MessageHandlers communicate on typed channels to asynchronously handle messages and responses.



Client Request Routing:

Shows a sequence of events when handling generic client requests on an established connection. JSON is parsed from the socket and onto a request.Base message. The MessageHandler processes this message, sent on the request channel, and puts its response message on the response channel. This response is then converted back to JSON to send back to the client.



Add Book Sequence:

Shows a sequence of events during an 'AddBook' request inbound on a ClientConnection. The connection's MessageHandler processes the message, performing more fine-grained operations with the help of a DBProxy. The DBProxy abstracts these requests using GORM models to perform database interactions. Responses are then sent back on the response channel to the ClientConnection.

