**Glossary**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Description** | **Author** |
| Inception Draft | July 20, 2018 | First draft. To be refined primarily during elaboration phase. | Katy Atchison |
| Elaboration 1 Draft | July 29, 2018 | Second draft. Changes include detailing a greater amount of detail and including new terms that appeared since the last draft. | Cody Dowell |
| Elaboration 2 Draft | August 1, 2018 | Third draft. Changes include adding minor terms that were recently discovered during elaboration 1. | Cody Dowell |

|  |  |
| --- | --- |
| **Term** | **Definition** |
| **All** | A Category-like list that stores a collection of books. This is where all books created are stored and can be used to generate a list of all of the books at any time. This is also where the user goes to find any books that they have not otherwise categorized. This category will be shown to a user by default. |
| **Account** | An account is the object a user will use to log into the system, and each account will have their own books, categories, ratings, and reviews. An account consists of a username and a password, and for someone to use our application, they will first need to create an account with a valid username and password. |
| **Book** | A class representing a physical book that the user may have read or wants to read. Will have attributes, such as Title, Author, and number of pages corresponding to the physical book. A book can be given a personal rating, as well as an aggregate review. A book object can also be added to s number of different categories that a user defines. All books belong in the “All” category. |
| **BookID** | A hidden attribute in the Book class that distinguishes the Book objects from one another and acts as a primary key in the database holding all books. A book’s BookID is automatically set when the book object is created and fulfills all other requirements that a primary key needs to function correctly. |
| **BookKeeper** | BookKeeper is the name of the application we are developing, and this term will collectively refer to the all of the software working to implement the functions of our application as a whole. |
| **Category** | Collection of references to Book objects. Users create and name these, and they are displayed in the user interface as a means of giving the Users organization in navigating through their saved books. A user is allowed to create as many categories as they want, and are also allowed to add as many books to category as they want. |
| **CategoryID** | A hidden attribute in the Category class that distinguishes the Category objects from one another and acts as a primary key in the database holding all categories. A categories’ CategoryID is automatically set when the category object is created and fulfills all other requirements that a primary key needs to function correctly. |
| **Classification** | All books have a classification of either “Read”, “To-Read”, or “Currently-Reading”. This affects what associations they have. Only “Read” books can have a Rating or Review, and only “Currently-Reading” books can have Progress. However, a book can change classification- for instance, it can first be classified as “Want-To-Read” and then, when the user starts the book, the classification can be changed to “Currently-Reading”. |
| **Database** | The database is a central part of the software that is responsible for storing data on all the users, books, and categories created on the software. The database is updated anytime an account is registered, when any changes are made to a book, or whenever a category is created or edited. The database will stay up at all times and will have preventive measures to stop total system failure in the case of the database going down. |
| **Database Modification Authorization** | The database modification authorization is the validation by the database and by the software to ensure that any information being created, edited, or deleted is valid before it is pushed onto the database, to prevent any system failures or a lack of comprehension for the user. In the event of information not being valid, it rejects the information and tells the user interface to tell the user what has occurred. |
| **Database Modification Authorization Request** | The database modification authorization request is an action done by the user interface when information is entered in an attempt to create, edit, or delete something on the database. It sends information to the database modification authorization, which will ensure that information entered is valid. |
| **Password** | The password is a log in credential that users will have to enter to log in, along with their username. The password field will display astericks when being typed in, and the actual password itself will be encrypted inside the database as to protect it from any outside attacks. Passwords will have some restrictions regarding the strength of it, so that people trying to gain access to other's accounts will not be able to guess a password eaily or quickly. |
| **Progress** | The User’s progress in a book they are currently reading. User will enter a page number, and Progress is calculated by dividing their page number by the total pages in the book. It is displayed as a percentage. Progress is stored in the database, and is able to be modified at any time, as long as the information is entered correctly. |
| **Rating** | The User’s personal rating out of ten for a book they have read. Only applies to books classified as “Read”. Ratings, once confirmed and validated, are stored in the database and are available for the user to view at any time. The rating for any book can be changed at any time after it has been initially set. |
| **Read** | This is a status that will indicate that the user has finished reading a specific book, and that the user's progress of the book will be at 100%. When a book's status is read, that user will be able to enter a rating and a review for the book. Until a book reaches read status, ratings and reviews are not available for that user. |
| **Review** | The User’s personal “Review” of a book they have read. Stored as plain text that they enter through a form. Reviews, once confirmed and validated, are stored and are available for the user to view at any time. The review can also be changed at any time after it is initially created, as long as the new review fits the restrictions provided. |
| **Submit** | When a user decides to click on a submit button while using the application, it most likely means that they are attempting to create, edit, or delete some type of data that the database is currently storing or wishes to store. So anytime a submit button is activated, a change in the database is soon to follow, if all data entered is valid and there are no system failures that occur. |
| **User** | A class representative of the user that summons their credentials, such as their name and username. A user logs into the system by entering their username and password. The password is encrypted to protect the user account. A user can add books, create categories, and overall has control of several features that the software offers. |
| **UserID** | A hidden attribute in the User class that distinguishes the User objects from one another and acts as a primary key in the database holding all users. A user’s UserID is automatically set when the user object is created and fulfills all other requirements that a primary key needs to function correctly. |
| **Username** | The username is a log in credential that will be entered into the system when trying to log in, along with the password. The user name will be shown as plain text when being typed into it's appropriate log in field, and the every user must have a unique username. |
| **+** | The + symbol is what a user will click on when trying to use any of the create functions of the application. This includes attempting to add a book to their all category, trying to add a book to a specific category, as well as adding a new category for books to be added to. |