**I. Scope**

**A. System objectives**

BookShare is website to connect people interested in exchanging used textbooks.

BookShare assists those within the DTCC community in loaning, or selling, used college textbooks to others in the community. Users will post multiple books for sale or loan. Others can search the postings using a Google-like keyword search. If an interesting post is found, the user requests their email address be sent to the poster. After that the two individuals work out an arrangement for selling or loaning the book through private email.

We believe this would encourage students to help each other out in the struggle for finding cheap textbooks by providing textbooks for a fraction of the cost in a small localized location. The service would be available for all who have a DTCC email address.

**B. Human interfaces**

There are three tasks that occur in Bookshare. Users may *Post* books, searching the postings, or change their account/profile information. The landing page should reflect the posting or search choice, and then lead to either a login page or a search view. Users may search the postings without creating an account. The login and profile pages will be reachable through a button on the upper right of the screen on the search page.

Once in the search view, the

* User will type ISBN, book title, author, or key word into search box and click a submit button. The page will re-write including the results of the search (standard search engine interface).  Each book, or posting, will have a *Request* button. If the user presses the *Request* button next to a book, the owner of that book will get an email from the site stating that so and so user would like to use their book.

In the posting views:

* The user will enter an ISBN number into a search form. The system will look up that book by ISBN number in the Google Book API and return title, author, alternate ISBN numbers, edition, year, and a thumbnail image of the book. The user will specify additional things like the condition of the book, and their Wants (Sell, Load, Free, Barter).  Then this information will be processed into the database and the posting will be available.

In the login/registration and account views:

* When registering for the account, the user will have to verify an email address by returning a code string before being able to *Post* or make a *Request* on a book.
* The user will be able to view their postings and remove any books no longer available.  This will remove the information from the database.
* The user can change their email address and password
* The user can delete their account. This will delete their account and all their book postings.

.

**C. Major software functions**

* Easily create and manage book listings
* User customizable profile page
* Manage connections between users with automated email messages
* Login, password, and email verification

**E. Major design constraints**

* ***Server Software***: The server needs to run on a standard Linux, Apache, MySql, PHP (LAMP) stack available through a low-end shared hosting site.
* ***Database***: The size of the database needs to be kept a minimum size to make maintenance easier.
* ***Front End***: The system should look good on the front-end so it can be shown-off to friends and future employers

**II. Reference Documents**

* *Murach’s PHP and MySQL, 2nd Ed*. Joel Murach and Ray Harris. 2014
* *PHP and MySQL Web Development(5th Ed.)* Luke Welling and Laura Thomson 2016
* *HTML and CSS: Design and Build Websites.* Jon Duckett 2011
* *JavaScript and JQuiery.* Jon Duckett 2014

**III. System Design and Team Responsibilities**

Model View Controller design will be used so that two development teams can work somewhat asynchronously. The *Front-End* team will be responsible for the design of all graphic elements, and any modifications of the human interface described above. They also will implement the controller (index.php) and all the view pages. The *Back-End* team will build the Model including the database, Google Books API, and email subsystem. They will provide the front-end team will sample views (in HTML) that demonstrate how to interpret and display database results. They also will provide functions for the controller where it needs to interface with the database.

### A. Front-End Team

A landing page will direct users to either post or browse listings. Both choices will lead to an index.php page that acts as the controller/dispatcher for all the views. Other than the search View and the registration pages, all pages will require the user to login and have a session id to view the page. This access test will occur in the index.php file; the Back-End team will provide the necessary function(s) to accomplish this test.

Login/Registration:

If a session ID is required is not available, the user will be redirected to the login/registration page. The successful login will create the session ID. The Registration will be a two-step process, where the user provides a username, email address, and password which is registered in the database but marked disabled. A ULA agreement will be displayed and acknowledged. At that point the email subsystem will send a verification message, including a numeric code, to the new user’s address. A link in that email will direct the user where to verify the email address by entering the code. Registration will be complete when this is done, and the login will be enabled.

Search View:

As described above in the Human Interface section. Additional details to be added here.

Account View:

Will allow the user to change email address, password, delete account, delete book postings, and delete their account. This may be implemented as multiple views/pages. All clicks on this page will lead back to the controller where the tasks will be completed and then the Account View re-displayed. The back-end team will provide the functions to complete all these tasks.

Post View:

As described above in the Human Interface section. Additional details to be added here.

Support View:

A support button will appear on most pages that leads to a form allowing the user to report problems to the support team. Users will be given a choice of subject lines including “Report Abuse”, “Report Website Problem”, “Username/Password Problems”, and “Other”. An email will be sent to the support team email address, and from there forwarded to support team members.

**B. Backend Team**

Database Design:

The database will have a user table, and a book listing table, plus any auxiliary tables needed for proper normalized design.

The user table will need to include Username, Password (preferably encrypted), a system assigned UID, a field indicating whether registration is complete, and a session id (SID) if there is an active session. Functions to be provided include (the names and argument lists to be negotiated between teams):

addUser(username, password, email) – adds user to the database

activeSession(SID) - A Boolean function that indicates if a session is active.

openSession(username, password) - returns SID if successful, else returns -1

removeUser(UID) - removes all book listings owned by the user, then removes the user

getEmail(UID) - returns the email address of a user

(more to be added here)

The book listing table will include listing id (LID), title, author, edition, year, listing user’s UID, wants, condition, number of requests, and The following functions will be provided to the front-end team:

* keywordSearch()
* searchByUser()
* addListing()
* removeList()
* incrementRequests()

Database Utility Programs:

These SQL or PHP/SQL programs will be run by administrators only. They are stored in a non-web accessible area on the webhost as well as the GitHub repository.

* Delete listings that are 12 months old
* Delete listings by UID

Google Book API:

This module is responsible for interfacing with the Google Book API, and providing info about the book. A sample of how to parse/use the result will be provided to the front-end team. Functions include:

* search(ISBN) - does a lookup for both 10 and 13 digit ISBN numbers and returns a list of results.

Email:

TBD

Log System:

* Keep the log as a flat file. Open a new file Monthly
* Log all emails with UID and LID
* Log software errors

Additional Functions:

The front-end team may require additional functions. The back-end team will be responsibility for any request for functions accessing the database, the Google API, or the email subsystem. The definition of those functions will be negotiated by both teams.

**IV. Personas**

**Pat:**

Pat is an 18 year old freshman at Durham Tech taking courses for the first time. Pat is computer literate, and accessing the site via a mobile device. Pat will search the listings, and after finding a needed book complete the registration from a phone or tablet. Pat may push the *Request* button on 2-10 books, and will only access the site in August and December.

**Alex:**

Alex is an 55 year full-time instructor at Durham Tech. Alex has been teaching 10 years, is slightly computer-phobic but competent at using online resources. Alex will access the system from a desktop, and probably list 5-10 books to lend or give away. System usage could occur any time of the year.

**Adrian:**

Adrian is 35 years old and working part-time as a car mechanic. Adrian is interested in getting books cheaply and selling those no longer needed. Adrian would prefer to borrow or barter as cash is very short. Postings are likely to occur quickly after the semester is over, and searches are likely to happen the week after classes start.