Jeffery Cui, jyc2ra@virginia.edu
John Zhang, jwz2kn@virginia.edu
CS 4720 Web/Mobile Development

Milestone 2 Progress

Thus far, we've implemented several screens of our textbook exchange app for iOS. The app is split into two main features, Buy and Sell books, accessible using the tab navigation at the bottom.

Buying a book:

In the Buy screen, you can refresh the books that have been put up for sale by pulling the screen downwards. You'll see book data displayed from an amazon site provided from AWS: http://ec2-52-91-193-208.compute-1.amazonaws.com/textbooks/.

Click on one of the books to see more detailed info about it. The detailed screen is not finished yet---- We have yet to finish our implementation of pictures for it. The grey square at the top is our placeholder imageview. Most notably, in the detailed view of a book, you can see that we've displayed the location/GPS of where the current owner of the book wishes to sell the book.

Selling a book:

In the Sell screen, you can see books that you've previously sold. This is currently filled with dummy data.

You can click on the + Add a book row to add a book, which takes you to a view where you must fill out info required to sell a book. Only a few of those fields (Title, ISBN, Author, Price, Image) are actually required--- You can't submit your entry to sell unless they're filled out. The required text fields highlight red if you edit them and don't fill them out.

If you go back to the main part of the Sell screen, you can see a little person icon in the top right corner. This is where the user can enter in their contact info so that it stays the same for each book they sell. This information will be stored on the server as well, but we thought it made sense to store it in local data using NSUserDefaults(), so that the data can be shared from that screen to other screens, and so that it stays the same between sessions.

Database:

You'll see book data displayed from an amazon site provided from AWS: http://ec2-52-91-193-208.compute-1.amazonaws.com/textbooks/.

For our webservice we set up a backend SQL database using a remote Windows image provided by AWS EC2. We used Razor C# and HTML to create the website that will pull info from the database as well as posting information to the database. As of right now, the webpage will pull data from the database and then print it out in JSON format. Our app will access this page and then parse the JSON to create objects we can then use more easily in the app. We see this implementation in the buy screen of the tab view. Pulling up will refresh the data if anything is updated.

For the posting (which we have not fully implemented), we will be using POST variables in order to send info and update the database. As of right now, we are just doing a hardcoded

GET variable url in order to update information. The takes place in the sell screen when you click submit. If, during testing, you need confirmation that this set up actually works, please don't hesitate to email us, and we can manually add new entries to the database.