Memorandum

To: Alka Harriger, CNIT 25500 Professor

From: The openLibrary team:  
Tai Gunter  
Ethan Madden

Date: April 23, 2013

Subject: Team openLibrary‘s CNIT 25500 Team Project Summary Evaluation

Enclosures: APPENDIX A: Table of Individual Contributions   
APPENDIX B: Testing Instructions   
APPENDIX C: List of Team files (actual files are included with the submission as a zip file)  
APPENDIX D: Final Usability Tests with results and analyses  
APPENDIX E: Client Evaluation/Feedback (using required template. If client sends to Prof. Harriger directly, we will add it in the location of the placeholder you have herein.)  
APPENDIX F: Team's Time Logs

Related: *To be sent under SEPARATE cover:* Individual evaluations from each team member

This report summarizes our experiences in developing an operational C# desktop application for **Chaz Brown**. It also provides information on our individual contributions to various aspects of the project. In addition to the technical aspects of developing and implementing this project, we also learned the following general, valuable lessons:

* .
* .
* .

# Use of Feedback from Each Source

Getting feedback from several different people was absolutely critical when developing this application. When programming, it is easy to get caught up in your own features and forget that not everyone is as familiar with the program as you are. Furthermore, not everyone using the program will have as much computer skill as you do. Having users test the application really helped us know what we were doing well and what we could improve on. Not all features that we thought were easy to use and intuitive were viewed the same way by our user base. Also, having the peer evaluations was a big help. Getting feedback from other programmers is useful too, because they might have ideas for different features that we simply didn’t think of. Finally, instructor and TA feedback is helpful because they’ve done this before and know what to look for. Getting feedback from them helped us to improve our application even further.

|  |  |
| --- | --- |
| SOURCE | FEEDBACK SUMMARY AND IMPACT ON PROJECT |
| Client | Told us what he was looking for, allowed us to expand into a more business-centered application |
| Users (usability tests) | Gave us feedback from users that are not as familiar with computers; this helped  us to develop an application that worked well and was easy to use, even if the  user isn’t familiar with the program. |
| Class (peer evaluations) | Gave us good insight to what other programmers and developers thought of the application, they also had good ideas from improvements. |
| Instructor and TA (milestone feedback,  office hours,  presentation feedback) | The feedback we received helped us to know that we were doing everything as  expected and that there shouldn’t be any surprises when the project was graded. |
| Team | Only having two members lessened this effect, but it was nice to have someone  else to talk to when we ran into a problem. If one person developed a new  feature, it was likely that the other had a way to make it better. |

# Discussion of Positive Lessons Learned

Replace this paragraph with an overall paragraph description. Minimally, list at least one positive learning experience for the team. Address design issues to better serve the target user’s needs as identified by the client or user feedback. Describe your planning and interaction with the client and each other. Explain the new things that each team member learned to implement all/part of the project.

# Discussion of Suggested Improvements

After development of the application had completed, we looked back on what we would have liked to accomplish had we been given more time to code. The thing that we both noticed was lacking from our project was the ability to fine people based on how overdue their book was. Currently, our program is set up to give overdue notices to people who have overdue books, but this is the only way that overdue books are pointed out.

The application could very easily implement a feature that told the user how much a given customer owed in fees, but we chose not to include that feature because we didn’t have the time to *fully* implement it. In our opinion, fully implementing this feature would mean that we would have to add an entire sales system to the application. There would be some situations where customers would simply pay the fines and be on their way, but other situations that would involve the book needing replaced. We are pulling the cost of the book from Amazon, so that wouldn’t be hard to implement, but the sales system would be complicated enough (different payment methods, tax, etc.) that we simply didn’t have time to implement it. As a group, we felt that choosing not to implement a feature at all was a better decision than implementing the feature but not doing a very good job.

# Discussion of Future Implementation

Our project is fully featured at this time and could be used by the client for his intended purposes. As a home system, it’s fully ready to go. It meets all requests of the client and does not lack any features necessary for home use. Since it is an open-source project, our client as well as other people that would like to use the application are free to use it and modify it.

This also means that a corporate library could, in theory, download and use our application. While it has a lot of features, we do not feel that it is ready for use in a corporate environment. There are a few features that would still need to be added (security, fining system), before we as a team would feel comfortable releasing this for recommended use in a corporate environment.

# Conclusion

Conclude with a summary paragraph. Be sure that all team members sign the header section and initial all attached pages confirming their agreement with the contents of this memo.

APPENDIX A: Individual Contributions

**Individual work contribution percentages**

|  |  |  |  |
| --- | --- | --- | --- |
| **Team Member** | **Total** | **Tai Gunter** | **Ethan Madden** |
| **Milestone reports and related documentation** | 100% | 40% | 60% |
| **Development of specific methods/classes (listed below) (See note 1 at the bottom of this table)**  **frmAbout.cs**  **frmAddBook.cs**  **frmAddCustomer.cs**  **frmAddEmployee.cs**  **frmAddGame.cs**  **frmAddMovie.cs**  **frmAddMusic.cs**  **frmAmazonKeys.cs**  **frmBugReport.cs**  **frmClockIn.cs**  **frmClockOut.cs**  **frmCurrentlyClocked.cs**  **frmEditBook.cs**  **frmEditCustomers.cs**  **frmEditEmployee.cs**  **frmEditGames.cs**  **frmEditMovies.cs**  **frmEditMusic.cs**  **frmFindMedia.cs**  **frmHomeScreen.cs**  **frmOverdueItems.cs**  **frmPrint.cs**  **frmViewBooks.cs**  **frmViewCustomers.cs**  **frmViewEmployees.cs**  **frmViewGames.cs**  **frmViewMovies.cs**  **frmViewMusic.cs**  **bookLookup.cs**  **databaseHandler.cs**  **otherLookup.cs**  **settings.cs**  **SignedRequestHelper.cs**  **TrackLookup.cs** | 100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100%  100% | 60%  100% | 40% |
| **Usability Testing** | 100% | 35% | 65% |
| **Presentation** | 100% | 50% | 50% |
| **Cool Feature 1: DataGridView Implementation** | 100% | 80% | 20% |
| **Cool Feature 2: Amazon API Book Lookup** | 100% | 20% | 80% |
| **Cool Feature 3: Amazon API Game Lookup** | 100% | 20% | 80% |
| **Cool Feature 4: Amazon API Movie Lookup** | 100% | 30% | 70% |
| **Cool Feature 5: Amazon API Music Lookup** | 100% | 30% | 70% |
| **Cool Feature 6: Music Preview** | 100% | 85% | 15% |
| **Cool Feature 7: Instant Search** | 100% | 50% | 50% |
| **Cool Feature 8: Email Notifications** | 100% | 30% | 70% |
| **Cool Feature 9: Receipt Printing** | 100% | 100% | 0% |
| **Cool Feature 10: Auto-Updating Login Menu** | 100% | 100% | 0% |
| **Total Contribution** | 100% |  |  |
| **Total hours on project-related tasks** | XX total man hours | XX | XX |
| **Letter grade based on contribution** | Team grade | X | X |

Note 1: A count using the Visual Studio Object Browser revealed about **340** methods in our program. Writing a percentage breakdown for each one of these methods would be extremely time consuming and not reveal any valuable information. Instead, we have listed a breakdown for each form or class file in the project, bringing the number down to a much more reasonable **35.**

APPENDIX B: Testing Instructions

A step-by-step list that explains how we can access the team’s project & assess its conformance to project requirements. In the event passwords/access codes/keystrokes/special links/etc. are required, they are provided in the numbered list.

| **Evaluation Criteria** | **Testing**  Describe how we should assess each criterion (may be as simple as listing which step below handles this). Indicate the places in the app where this can be tested | **Developed By**  Explain the role each team member played in achieving this goal |
| --- | --- | --- |
| **Access Instructions** | | |
| * Identify all passwords and special links required to use the application and all of its features. If access for any section is via databases/text files, please indicate the section and where to get the login from. |  |  |
| **Navigation Structure** |  |  |
| **Application design and layout** |  |  |
| **Usage of tools and technologies taught in class.** | | |
| * Using databases to lookup records. | **View > Books** or  **View > Movies** or  **View > Music** or  **View > Games** |  |
| * Using databases to add records. | **Add > Books**  or  **Add > Movies** or  **Add > Music** or  **Add > Games** |  |
| * Using databases to delete records. | **View > Books** or  **View > Movies** or  **View > Music** or  **View > Games**  …then…  Right Click a record > **Delete** |  |
| * Using databases to update records. | **View > Books** or  **View > Movies** or  **View > Music** or  **View > Games**  …then…  Right Click a record > **Edit** |  |
| * Using text files to read data. |  |  |
| * Using text files to write data. |  |  |
| **Additional features not demonstrated in class**  Elaborate on the unique/cool features you added. For each feature, describe the specific needs that the feature addressed for the client. Relate how you implemented the feature(s) and describe the challenge level in its implementation. Did you incorporate the features illustrated in class? Did you research the web and locate sample code? Did you develop the code completely from scratch? Finally, describe the added value it brought to your client/user/application. |  |  |
| **Student-Client communication** |  |  |
| **Project Documentation** |  |  |

1. Testing instructions for demonstrating uniform application
2. Testing instructions for demonstrating reading from text files and exception handling.
3. Testing instructions for demonstrating writing to text files and exception handling.
4. Testing instructions for demonstrating database usage for lookup and exception handling.
5. Testing instructions for demonstrating database usage for add and exception handling.
6. Testing instructions for demonstrating database usage for delete and exception handling.
7. Testing instructions for demonstrating database usage for update and exception handling.
8. Testing instructions for demonstrating collection of user information with appropriate validation.
9. Testing instructions for cool features we added. (Repeat for each feature)

APPENDIX C: List of Team files (all non-graphic files)

A hierarchical listing of all files in the team folder’s directory listings that identifies all files included with the submission

APPENDIX D: Final Usability Tests with results and analyses

APPENDIX E: Client Evaluation/Feedback

APPENDIX F: Time Logs

|  |  |  |  |
| --- | --- | --- | --- |
| **USER** | **DATE** | **DESCRIPTION** | **HOURS** |
|  |  |  |  |
| Ethan Madden | 3-18 | Created Github account, repository, and working directory | 1 |
| Ethan Madden | 3-19 | Created database model in Oracle SQL developer | 3 |
| Ethan Madden | 3-21 | Created Access database  Created **library.mdb** | 2 |
| Ethan Madden | 3-21 | Created relationships in Access Database | 1 |
| Ethan Madden | 3-26 | Fixed issues in BOOK table of database  Began working on API pull for books | 2 |
| Ethan Madden | 3-26 | Continued work on database and Amazon API pulls | 1 |
| Tai Gunter | 3-26 | Added C# project to working directory: openLibrary 1.0  Created **frmMain.cs** | 1 |
| Tai Gunter | 3-26 | Added database connections, began work on adding books  Created **databaseHandler.cs** | 2 |
| Ethan Madden | 3-26 | Amazon API successfully pulls data about books | 4 |
| Tai Gunter | 3-26 | Began working on adding books to the database  Created **frmAddBook** | 2 |
| Ethan Madden | 3-26 | Simplified and increased efficiency of Amazon API lookup  Created **lookup.cs** | 2 |
| Ethan Madden | 3-27 | Added functionality to pull more things from Amazon API | .5 |
| Ethan Madden | 3-27 | Added error handling if the ISBN was of an incorrect format | .5 |
| Tai Gunter | 4-1 | Created new working directory for the project: openLibrary 2.0 | .25 |
| Tai Gunter | 4-1 | Continued work to add books to the database | 1.5 |
| Tai Gunter | 4-2 | Added handling to fix apostrophes in the textboxes | 1 |
| Ethan Madden | 4-2 | Added functionality to pull CD information from Amazon API  Created **otherLookup.cs** | 3 |
| Tai Gunter | 4-2 | Added functionality to view the database contents from within the app using DataGridViews  Created **frmViewBook.cs** | 3.5 |
| Ethan Madden | 4-2 | Increased functionality of CD lookup through Amazon API | 1.5 |
| Ethan Madden | 4-3 | Added functionality to look up movies and games through Amazon API | 4 |
| Ethan Madden | 4-3 | Improved functionality of movie and game lookup from Amazon API | 2 |
| Tai Gunter | 4-4 | Attempted to fix issues with DataGridView not refreshing when a new book is added to the database | 3 |
| Tai Gunter | 4-9 | Fixed issue where DataGridView was not refreshing after a new book was added. Added functionality to view games, customers, and employees. Added forms to add employees, customers, music, games, and movies.  Created **frmAddCustomer.cs,**  **frmAddEmployee.cs,**  **frmAddGame.cs,**  **frmAddMovie.cs,**  **frmAddMusic.cs,**  **frmViewGames.cs,**  **frmViewCustomers.cs,**  **frmViewEmployees.cs** | 6 |
| Ethan Madden | 4-10 | Moved functionality to add items into a new menu, as per our client’s request | 1 |
| Ethan Madden | 4-10 | Made changes to the database by adding tables for new media and changing the required tables | 1.5 |
| Ethan Madden | 4-10 | Added functionality to add customers and employees to the database | 2 |
| Tai Gunter | 4-11 | Added instant book search | 2.5 |
| Tai Gunter | 4-11 | Added music preview functionality | 3.5 |
| Ethan Madden | 4-12 | Modified Amazon lookup to pull track information for CDs | 3 |
| Ethan Madden | 4-12 | Added track numbers to the tracks being pulled from Amazon | 2 |
| Ethan Madden | 4-13 | Added support for CDs that have multiple discs | .75 |
| Ethan Madden | 4-13 | Added track information to the main project (from prototype project) | 1 |
| Tai Gunter | 4-13 | Added insert statement to add tracks to the database when scanned | 2.5 |
| Ethan Madden | 4-13 | Modified Amazon lookup to pull star actors for movies scanned into the database | 3 |
| Tai Gunter | 4-13 | Working on getting the tracks to show up when an album is clicked in the DataGridView | 2 |
| Ethan Madden | 4-14 | Actors are now added to the database when a movie is scanned and added | 2.5 |
| Ethan Madden | 4-15 | Fixed relationships within the database now that actors and tracks are being added | .5 |
| Tai Gunter | 4-15 | Tracks now successfully show up in a listbox when an album is clicked in the DataGridView | 2 |
| Ethan Madden | 4-15 | Made a new Amazon lookup call to find the ASID of a track when it is clicked in the listbox, will be used with music preview | 2 |
| Tai Gunter | 4-15 | Music preview is now successfully integrated and functional | 3 |
| Tai Gunter | 4-16 | Added functionality to clock in, clock out, log in, log out, and created checkout interface  Created **frmLogin.cs,**  **frmLogout.cs,**  **frmCurrentlyClocked.cs** | 4 |
| Tai Gunter | 4-17 | Added functionality to checkout and clocking in / clocking out. Time clock information is now stored in the database | 2 |
| Tai Gunter | 4-18 | Worked on checkout and storing this information in the database | 1.5 |
| Tai Gunter | 4-19 | Worked on checkout some more; | 1 |
| Tai Gunter | 4-19 | Ability to renew books added | 1.5 |
| Tai Gunter | 4-20 | Checkout is now fully functional | 2 |
| Ethan Madden | 4-20 | Setup created to make email notifications  Created **mail.cs** | 2 |
| Tai Gunter | 4-21 | Integrating email into the main app (from prototype)  Created **frmOverdueItems.cs** | 2 |
| Ethan Madden | 4-21 | Created prototype for settings file, used to store Amazon API key information  Created **settings.txt** | 2 |
| Ethan Madden | 4-21 | Integrated settings file so that the Amazon API lookup uses the information | 1 |
| Tai Gunter | 4-21 | Began to redo checkout to improve functionality and make it more organized | 1 |
| Tai Gunter | 4-21 | Checkout fully redone, writes to new tables in the database, does not record in the media tables anymore | 2 |
| Tai Gunter | 4-22 | Checkout bugs fixed, Check in works (redesigned), Renew works (redesigned), made changes to time clock setup, overdue items report works completely, settings.txt fully integrated with GUI, email notifications fully integrated, added ability to look up a customer if without their ID number, added music search, added game search, fixed bugs in addMovie, actors are now being added correctly to the database  Created **frmAmazonKeys.cs** | 9 |
| Tai Gunter | 4-22 | Created PowerPoint for presentation  Created **openLibrary.pptx** | 1 |
| Ethan Madden | 4-23 | Fixed a bug in music preview where the program would crash if three songs were selected within 30 seconds of each other | 2 |
| Ethan Madden | 4-23 | Added search functionality for movies | 2 |
| Ethan Madden | 4-24 | Added functionality to delete things from the database through DataGridView via right click | 3.5 |
| Ethan Madden | 4-24 | Began work on edit functionality through DataGridView via right click  Created **frmEditBook.cs** | 2 |
| Tai Gunter | 4-24 | Added functionality to print receipts  Created **frmPrint.cs** | 4 |
| Ethan Madden | 4-25 | Completed edit functionality for books | 2 |
| Ethan Madden | 4-25 | Added edit functionality for music  Created **frmEditMusic.cs** | 1 |
| Ethan Madden | 4-25 | Fixed settings file to no longer ask for email address, also will create the file if it does not exist (bugfix) | 1 |
| Ethan Madden | 4-25 | Added edit functionality for movies  Created **frmEditMovies.cs** | 1 |
| Ethan Madden | 4-25 | Added edit functionality for games  Created **frmEditGames.cs** | 1 |
| Tai Gunter | 4-26 | Added “about” page  Added bug reporting feature  Created **frmAbout.cs**  Created **frmBugReport.cs** | 3 |
| Ethan Madden | 4-26 | Added validation to make sure that only employees that are in the database can clock in. | 1 |
| Ethan Madden | 4-26 | Added availability Boolean to media items | 2 |
| Ethan Madden | 4-26 | View media menus now show things in a better order and show if the item is available to be checked out. | 2 |
| Ethan Madden | 4-27 | Added validation to make sure items must be checked in before they can be checked out | 1 |
| Tai Gunter | 4-27 | Changed available column to mark new items as available when they are added to the database  Small bugfix in viewMusic | 1 |
| Tai Gunter | 4-27 | You can now find an item in the database if you don’t know what the UPC/ISBN is.  Created **frmFindMedia.cs** | 2.5 |
| Tai Gunter | 4-27 | Implemented searching for customers and employees by name or card code | 1.5 |