**YOUR NAME: Steven Bedoya**

**TITLE OF YOUR PROJECT: LimeWired**

**WEBSITE URL#1 OF DATA: wired.com**

For each of the requirements specified in the Requirements Document (emailed prior), describe in your own words how it will be fulfilled in your project. Assume URL#1 above for this proposal. Type in each box. The box will grow as you add lines to it. This isn’t merely a writing exercise. You are expected to follow what you are proposing once approved.

**Requirement1. Online (Search) function.**

**The program can search throughout the wired website by category, links and even through an advance search. Wired contains mainly mainstream news and allows the user to find multiple data of these articles and can easily choose what they want using the advance search feature.**

**Requirement2. Storage of Data.**

**The data is stored in a hash table and I never got to add in a database feature. The hashtable is saved into a folder and once the user closes the program they can later come back and see all their data still saved. The database is separate for each user and not one united.**

**Requirement3. Offline queries.**

**The user can see all their data saved in their hash table and search through it with ID number or even an advance search filter that allows them to find thinfs related to the inputted search term. User can also minimally edit the data and can fully delete any data in the table. More ways to edit the data could have been done and even deleting the enitre database or based category or month would have been a neat feature.**

**Requirement4. Persistence of Data.**

**The data is saved in a hash table can be reopen later when the user logins in into the program. There is currently no way to backup the data from log files or any other way.**

**Requirement5. Processing individual pages.**

**The program can read multiple web pages and read the data and save it into the hash table. It gets everything from the title, date, author, article etc.**

**Requirement6. Input/Output from Command Line.**

**The program functions without the use of a GUI and allows the user to do everything from add, delete, find, edit and even has command flags that allow easier usage of the program.**

**Requirement7. GUIs.**

**Due to time constraints a GUI could not be implemented and only a basic display of buttongs, text field can be seen but not functional. I really wanted to add a GUI after I was done with the basic foundations but ran out of time it was final section to complete.**

**Requirement8.** **Documentation.**

**Comments for methods and classes and lines of code are provided when necessary. An user manual is provided that contains instructions and any issues that the program has. Screenshots are included in the user manual and how to install the program.**

**Requirement9. Transaction Log.**

**A log file that contains the timestamps and transitions of the user outputs after the program closes or logouts. It shows what data was inputted or outputted, deleted, added and what actions the user took. This can not be used to back up data from.**

**Requirement10.** **Code Length.**

**The code is over 1000.**

**Requirement11. User Registration.**

**A very basic user registration was implemented that allows the user to register and login into the program. The permissions for the user and admin are the same and guest is not allowed to add, delete or edit data from the dataabase. A text file was used to store the names and password and isn't the best possible way to do it but was to late to change it during the project. Needs more better permissions and other way to store the user and password.**

**Requirement12. Innovation#1.**

My first innovation was scrap for an advance search that allows the user to find multiple article based on the inputted data from the user. The user can search up to 9 pages if it exist for that given data. The advance search even works with spaces such as "fast car" or "Marvel movie".

**Requirement13. Innovation#2.**

One of the innovations I was going to implement was to allow the user to pick key things they want to be added to their data table after logging in. Didn't get to fully implement that feature.

**Comparison**

I was able to accomplish around 80% of what I wanted to do but was able to get the foundation of the project complete. When I first got the project I had many ideas but it was hard to implement them because I didn't have the skills to make them real. As I worked on the project I learned new things and looked back on the old sections and realized could have done them in a more efficient way such as the login/register and even the hash table. I got stuck around the regex of the wired.com and needed to fully understand regex and see what works on getting the data efficiently. Testing out the regex took some time and then making sure it successfully enter the database also took time. I really wanted to add the GUI and I thought it was going to be easy to make but underestimated the time it takes to learn how to do it properly and have it functioning. The program is set up to make the GUI work but am disappointed that I could never get to it. This project showed me why we need software methodologies and structure to create big projects like this because I went in straight coding and after a while things got messy and was hard to change things without breaking something else. I learned that I should layout my plans and have mini checkpoints/milestones and know how to properly test/debug my program. I never had a complex project like this before and it stressed me out but I learned a lot and will use the experience and knowledge gained when developing my next big project.