Final Report

Team RightAfterDeadline

Summary

Our main customer and stakeholder is Dr. Lunney. Other stakeholders include us the providers and engineers, the instructors of CSCE 606, and other data scraping engineers who would be benefited from our software. The main customer needs a software that takes an excel spreadsheet of book names or song names as input and gets an excel spreadsheet with scrapped results corresponding to each entry as the output. He emphasized on scrapping important data like the price difference between ebooks and paperback, book price, song durations, play counts, and sell figures.
To fulfill his needs to scrape data for both books and songs, we splitted up into three sub teams. The songs team allocated YouTube API and last.fm API as data source and extracted playcounts, listeners, durations of the songs from them. The books team used Amazon API as the data source to extract information about prices for different versions of books (kindle, paperback, hardcover, etc.) for the given books. The UI team implemented the user interface, tested I/O redirection and integrated other parts together.

User Stories

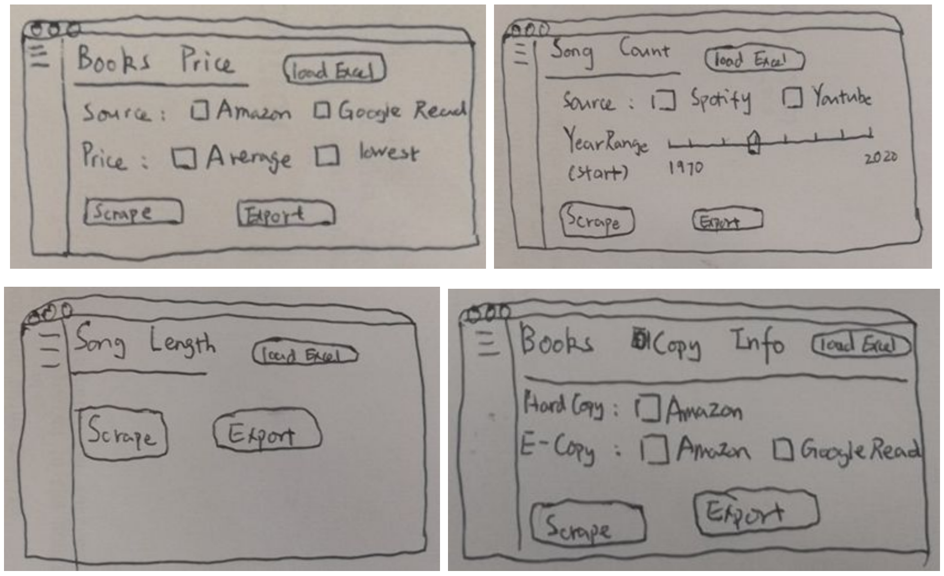

Figure 1: Lo-fi UI mockups. Top: left to right: Feature 1 and 2. Bottom: left to right: Feature 3 and 4

1. Feature: find average and lowest prices for a book

As a researcher in copyrights
So that I can see price difference between books with and without copyrights
I want to see the lowest and average prices of the book from the internet

We gave this user story 2 points because finding average and lowest book prices require some advanced processing on top of data scraping. We eventually dropped it because only limited portions of our test samples could find enough information to determine this feature's usefulness.

2. Feature: find view/listen count for a song

As a researcher in copyrights
So that I can see a song's trend over the time
I want to see the view count/listen of this song from the internet

We originally allocated 2 points for this one but eventually raised to 3 points because view and listen were actually two independent pieces of information. We accomplished that test and were able to find view/play counts from both YouTube API and last.fm API while listener count was only available from last.fm API.

3. Feature: find the length of the song

As a researcher in copyrights
So that I can match songs with similar length
I want to see the length of the song from the internet

We originally allocated 1 point for this one but eventually raised to 2 points because finding duration was not as easy as we thought. We originally tried to use Spotify API but later switched to last.fm API, where it provides duration as a metadata.

4. Feature: find price for both electronic copy and hard copy of a book

As a researcher in copyrights
So that I can see if electronic copy and hard copy make differences if book is copyrighted
I want to see the price of electronic copy and hard copy of a book (if applicable)

We gave 2 points to this feature because it requires intermediate efforts to find since not every book has both versions available. The feature was completed and polished to show prices of 4 common formats of copies (kindle, paperback, hardcover, and audiobook). All of them were extracted with the Amazon API and displayed all available books prices in real time.

5. Feature: I/O interaction: save file and results (w/ custom name)

As a copyright researcher
So that I can view and save the result
I want to save the results for future research

We gave 1 point to this feature because it requires some efforts to design an UI that allows user to import and export files with freedom.

6. Feature: Make tasks running in parallel

As a researcher in copyrights
So that I can efficiently get results for different types of input

I want to scrape the data in parallel

We gave 2 points to this feature but then raised it to 3 points because it requires the UI team to first refactor the program, adding extra work for them to finish. With some efforts to develop parallelism, the program now partitions the input into multithreading and then combines the result to make the program run more efficiently. No low-fi UI mockup since it's embedded.

This project is not a legacy project

Team Roles

Scrum Master: Sicong Huang
Product Owner: Sicong Huang
UI Team: Chen Liang
Songs Team: Fengqiao Wang, Sicong Huang, Joseph Cineros, David Qin
Books Team: Aditya Atul Vijayvergia, Fengqiao Wang

Name	**Team**	**Role**
Sicong Huang	Songs Team	scrum master, product owner, last.fm API implementation
Chen Liang	UI Team	GUI design, multithreading implementation, code merge and refactor
Fengqiao Wang	Songs Team / Books Team	Google Books API implementation, Spotify API implementation, last.fm API implementation
Joseph Cineros	Songs Team / Books Team	Spotify API implementation, Barnes Noble API implementation
David Qin	Songs Team	YouTube API implementation
Aditya Atul Vijayvergia	Books Team	Amazon API implementation

- Fengqiao Wang used to work on Google Books API, but after proving this API is not viable to the project, he moved to implement Spotify API. After realizing Spotify API doesn't provide sufficient data, he eventually moved to last.fm APi.
- Joseph Cineros originally signed up to work on Spotify API and Barnes Noble API, after realizing Bares Noble API was only for business partner, he dedicated his attention on Spotify API

Iteration Summary

Iteration 0: initialized tasks, talked to the client about specifications and user stories, and planned for user stories and tasks.

Iteration 1: UI was finished with dummy entries; finished researching YouTube API; determined that Barnes Noble API doesn't work thus dropped it
Iteration 2: Found duplicate results being a fundamental problem for both songs and books scraping; Customers can import and export excel files with GUI now; Books team changes method of scraping from API calls to GET request; songs team can extract metadata and display it
Iteration 3: Met with client to handle duplicates and demoed current progress with client; Books team now bypassed to use spider scraper module to get an extensive amount of GET requests and process them; UI team now implements progress bar and modified the input data format into pandas dataframe to facilitate more smooth integration; Song team deemed Spotify API was not very useful and decided to switch to last.fm API
Iteration 4: Songs team finished last.fm API and confirmed the UI can successfully use the books scraping feature, also finished the YouTube API data scraping except for some issues with usage quotas; Books team finished touching up the Amazon API and connected it to the UI; UI team finished integrating data scraping with last.fm, YouTube API, and Amazon API, all three features were functional (except YouTube API can run out of quota)

Customer Meeting

Meeting 1: 02/25/2020 from 1:30 PM to 2:00 PM at Evans Library 204E
Introduced the team and discussed the client's needs
Meeting 2: 04/10/2020 from 12:30 PM to 1:00 PM via Zoom
Demonstrated the GUI capable of taking input and output but haven't implement data scraping functionality; discussed about duplicate results and talked about data source
Meeting 3: 05/04/2020 from 5:00 PM to 5:30 PM via Zoom
Demonstrated every component to the client. Except that YouTube API had quota issue, every other demo was successful and received the client's praise; discussed how to use the software and some constraints that needed to be added to the input

BDD/TDD process

BDD

As a behavior driven development process, we follow the user stories and build the habvior of each feature in parallel. For example, when testing last.fm API, our BDD approach was used to make sure the code can take in the specific input and print the output in data frame structure. And later with the input of dataframe structure, we focused on making sure the scrapped results were in a dictionary of dataframes.

TDD

When using the test-driven development process, we focused on passing tests during development. For example, if the input is not in all lower case, the code will modify the input queries with the specific translator code written. In this stage, we focused on whether the test was passed and the condition was triggered, not the output behavior of the formulated queries.

Configuration Management

We have 2 spikes during our project. The first spike is that the books team realized that using Amazon API is unreliable and insufficient and decided to switch to the Spider Amazon Books API (GET requests). The second spike is that when Spotify API failed our expectation, we had a surge to look for replacement and implemented last.fm API as a backup, negating the loss of data from the Spotify API. We have two releases. The first release demonstrated the prototype UI without functioning scraping features. The final release where we demonstrate the functionality of every data scrape feature integrated to the UI.

Since the program is designed to be running locally on Windows, no cloud platform, such as Heroku, is used. Instead, we merged all functionality into a single user interface to provide a one-stop solution to all scraping functions mentioned in this project.

For collaborative coding and version control, GitHub is used as the hosting service for the project, allowing teammates to work in parallel. In addition, the project is divided into 5 iterations, and for each iteration a release will be created on GitHub. At the end, the final project is uploaded as the final release so that the customer can download and use the program directly.

Besides the packages and libraries that are used for achieving the functionality of code, an additional tool named ‘pyinstaller’ is used to pack the whole project. This special tool can pack all dependencies and required files into one project folder and allow the user to execute the program on Windows system without installing Python and the required dependencies.
All codes are up to date (and please follow readme.md to build the project)

Pivotal Tracker: https://www.pivotaltracker.com/n/projects/2437225
GitHub: https://github.com/Innoversa/Law-School-Copyright
Detailed Documentation (also shared with the customer):
https://drive.google.com/file/d/1PKAMdr2ysBlk3kFAf2fVvp40G7EC6DJG/view?usp=sharing
(Download if not shown properly on Google Doc)