

DATE: April 22nd, 2019
TO: Dr. Mary Eberlein, TA's
FROM: Team Eagle - Hiep Nguyen, Michael Flanders, Kunpeng Qin, Nabil Khan, Nelson Levy, Wesley Klock
SUBJECT: Team Eagles' BookBrain Phase 3 Report

We are team Eagles on Canvas, and our project is BookBrain. Our github can be found here: <https://github.com/Flandini/461L-Eagles>, and our webapp is located at <http://bookbrain.club/>. We have 6 members on our team:

- Name: Hiep Nguyen, Email: hiepnguyen8069@gmail.com, github username: hnguyen257
- Name: Michael Flanders, Email: flanders.michaelk@gmail.com, github username: flandini
- Name: Nelson Levy, Email: nlevy2807@gmail.com, github username: nrl538
- Name: Wesley Klock, Email: wesleyklock@gmail.com, github username: wesclock777
- Name: Kunpeng Qin (Phase 3 Leader), Email: qinkunpeng2015@utexas.edu, github username: qkpqkp
- Name: Nabil Khan, Email: nabilliban14@gmail.com, github username: Nabilliban14

TASKS, FEATURES, AND FUNCTIONALITY

For the third phase, we promised five things:

- Dynamic web pages
- Dynamic web page tests
- Improved book searching and filtering
- More sources and thousands of books listed
- A fully functioning web application

For this phase, we focused on finishing our scraper by adding all scraper sources, making our web pages dynamic, scraping more book data, continuing to write tests, and displaying sentiment analysis of book reviews.

For our user interface (UI), we improved the overall mobility of the frontend and information presentation. We added additional UI elements that change without page reload such as the “Save” and “Remove book from favorites” buttons, our dynamic critic review loading for books, and the similar books carousel. Additionally new content regarding sentiment of reviews and an explanation of these metrics are displayed.

We have increased our number of tests from 32 to 58 and have unit tests, integration tests, and dynamic web content tests as promised for this third phase. We decided to implement dynamic testing using Selenium to make sure our front-end was reacting as intended. These tests were implemented in Java using JUnit and ran via Maven.

We have added Amazon, Twitter, and Reddit to our previous scraper sources: Barnes & Noble, IDreamBooks, and GoodReads. We have also added the IDreamBooks widget to each book’s page that have critic reviews available. We are now also displaying a “Polarity” and “Subjectivity” metric for each type of review on each book’s page.

We have implemented extra features such as a user profile page where users can view their recently viewed books and saved/favorited books.

DESIGN

We are using Python2.7 and Flask for our web application. Because there are no classes in our scraper, testing, or web app code, we have not included a UML class diagram as we felt that it would not properly describe the design of our application. Instead, we have included a UML sequence diagram below to describe the design of our web application for this third phase. The diagram was rather large, so you may need to zoom in to properly see the text.

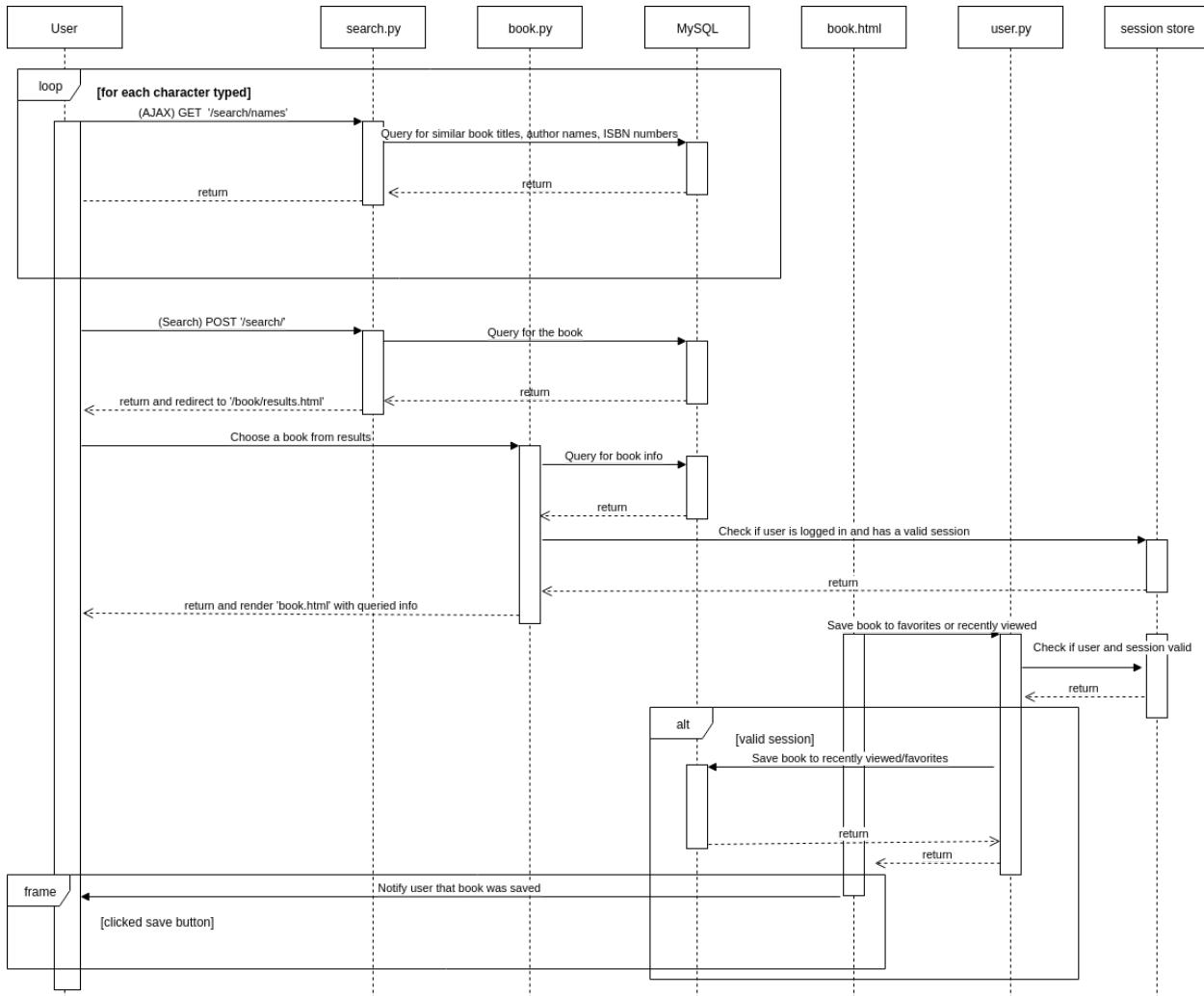


Figure 1. Sequence Diagram depicting autocomplete, search, and book page display with dynamic save to favorites/recently viewed

Python 2.7 was used for scraping code. For scraping we used APIs where possible to simplify and speed up the gathering of information. If no API was available, Selenium was used to navigate the site and gather the information using BeautifulSoup to parse the site's source. The scrapers use an input of the main csv file we are using for all our books and output a csv with key information from the main csv as well as the added information that was gathered.

REQUIREMENTS

We introduced 5 new user stories for this phase that are available on our Github issues board (<https://github.com/Flandini/461L-Eagles/issues/4>):

- As a person looking for trendy books and reviews, I want to be shown reviews and sentiment of a book from social media sites.
- As a person looking for a new book to read, I want to be shown books and their covers similar to the current one I'm looking at.
- As a person who isn't great at remembering every book I've looked at, I would like to be able to register, login, and save my favorite books and their reviews.
- As an analytical person, I would like to know how subjective listed book reviews are.
- As an analytical person, I would like to know the sentiment and emotion of listed book reviews.

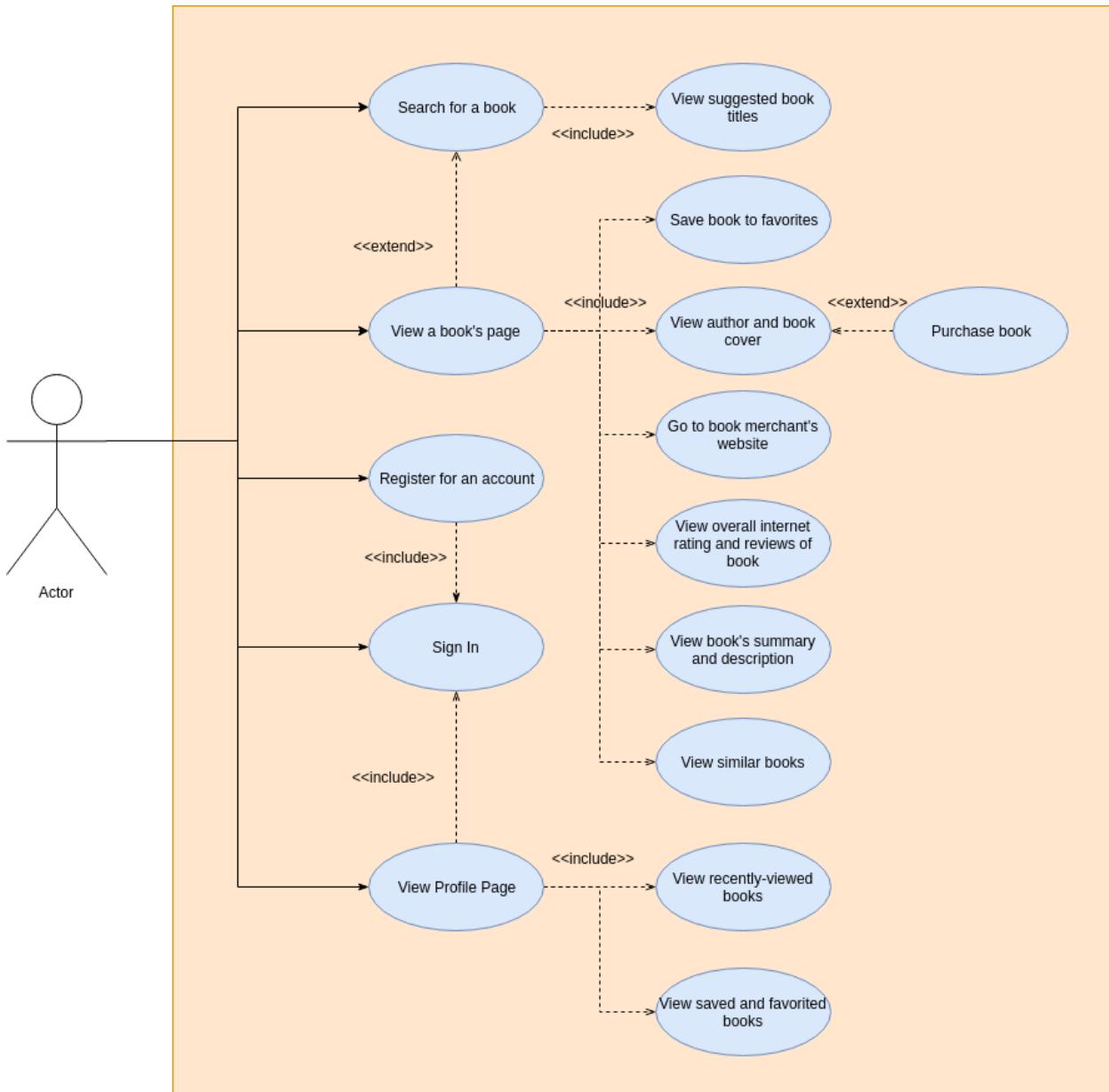


Figure 2. Phase 3 Updated Use Case Diagram

This phase's user stories focused on users being able to make accounts and save books while presenting more data per book to the user. We have implemented and fulfilled all of the requirements outlined by these user stories and have shown this in Figure 2 below. There are differences between Figure 3, our goal use case diagram for our finished web application, and what we have actually implemented shown in Figure 2. The reason for these differences is that

during our Agile development process, we decided to take a more user-oriented approach to the website. As a result, we did not end up fulfilling the use cases of users being shown a list of trendy books on the homepage or the use case of being able to filter by genre. Instead, we have fulfilled new use cases of being able to register for an account, login to that account, save books to a favorites list, and view recently viewed books. We have also implemented functionality to show users similar books to the ones they have viewed, but there is no genre support.

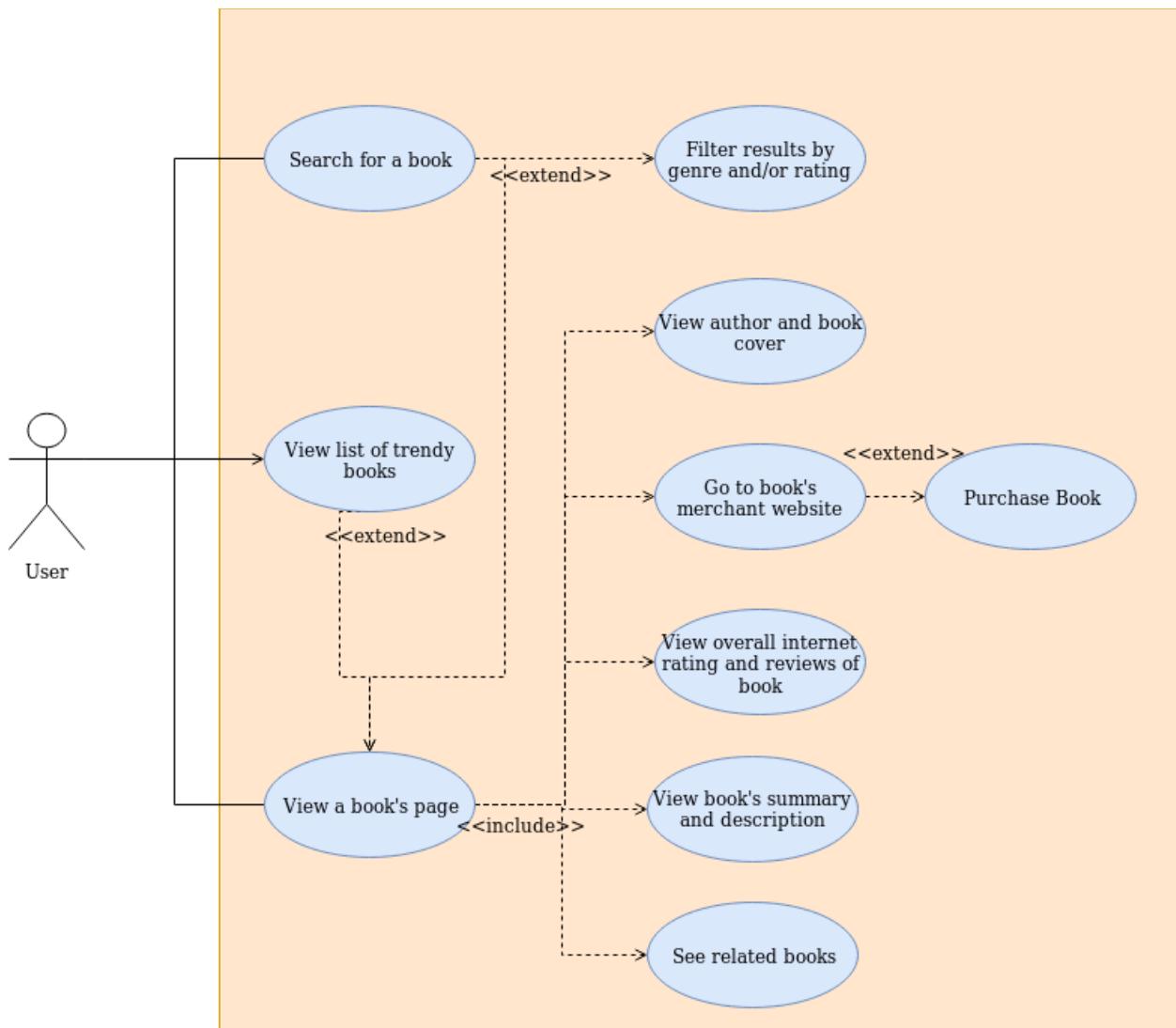


Figure 3. Goal Use Case Diagram/Use Case Diagram of BookBrain at Completion per Proposal

Document

Screenshots of our finished web application, BookBrain:

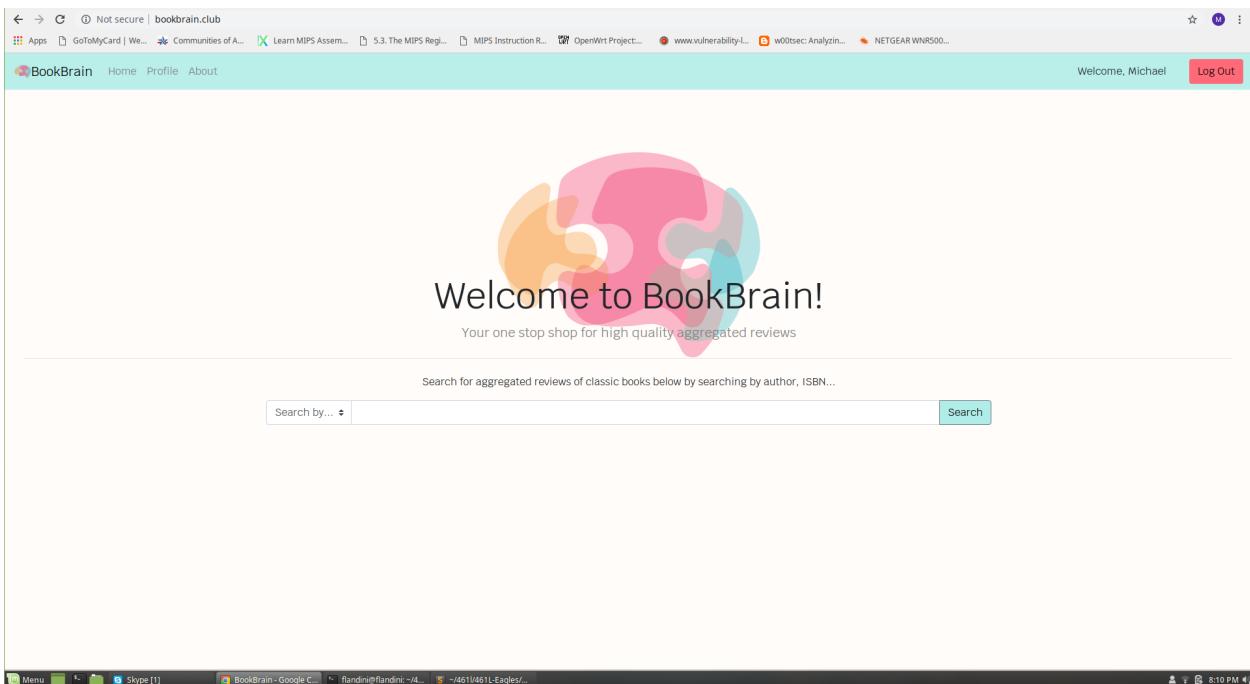


Figure 4. BookBrain Home Page with User Logged In

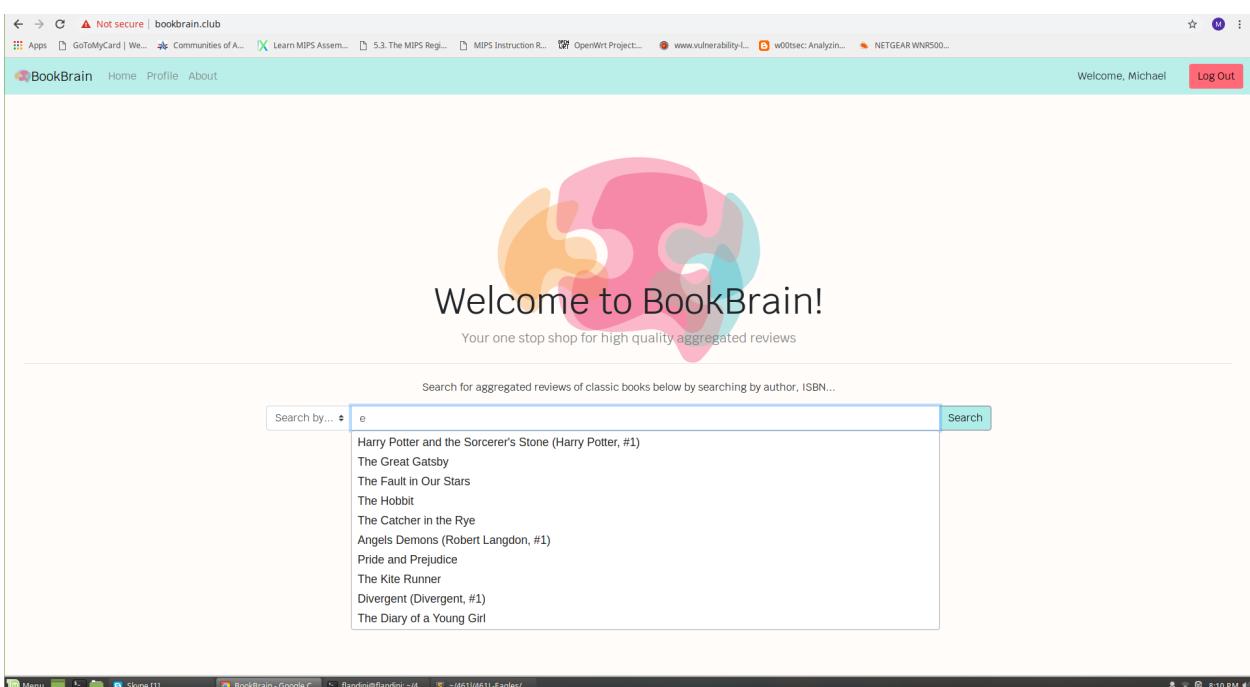


Figure 5. BookBrain Autocomplete Functionality

The screenshot shows a web browser window with the URL bookbrain.club/search/. The page title is "Results". The first result is "Harry Potter and the Sorcerer's Stone (Harry Potter, #1)" by J.K. Rowling on 1997. The second result is "Harry Potter and the Prisoner of Azkaban (Harry Potter, #3)" by J.K. Rowling on 1999. The third result is "Harry Potter and the Order of the Phoenix (Harry Potter, #5)" by J.K. Rowling on 2003. The fourth result is "Harry Potter and the Chamber of Secrets (Harry Potter, #2)" by J.K. Rowling on 1998. The results are presented in a clean, modern interface with a light blue header.

Figure 6. BookBrain Search Results Page

The screenshot shows the BookBrain search results page for the book "Twilight" by Stephenie Meyer. The main image is the book cover, which features a red apple held by two hands. To the right of the image, there is a "Rating" section with the heading "BookBrain Averaged Internet Rating: 3.57" and a note that it is based on a weighted sum of ratings from multiple other websites and sources. Below this is a "Tweets Mentioning This Book" section, which displays several tweets with their polarity and subjectivity scores. The tweets include:

- "I read twilight and I was like what is this shit?" Kristen :) pic.twitter.com/EBycWNJh
Subjectivity: 0.90 ⓘ
Polarity: 0.02 ⓘ
- Today's Twilight pic.twitter.com/JQOckdbsn
Subjectivity: 0.00 ⓘ
Polarity: 0.00 ⓘ
- joint #pic.twitter.com/HvPQaFBQIN
Subjectivity: 0.00 ⓘ
Polarity: 0.00 ⓘ
- Highlights from Saturday's Illini Twilight, featuring the Gary Wienke recognition, the Senior Day ceremony and #Illini competition from throughout the day pic.twitter.com/CdHBCJxaNx
Subjectivity: 0.00 ⓘ
Polarity: 0.00 ⓘ
- (game; twilight.princess hd) pic.twitter.com/NLtoUyxbR2
Subjectivity: 0.40 ⓘ
Polarity: -0.40 ⓘ
- Still better love story than twilight pic.twitter.com/XS2PuozTd3
Subjectivity: 0.55 ⓘ
Polarity: 0.50 ⓘ

Figure 7. Screenshot 1 of 4 of Individual Book Page (No User Logged In)

Similar books

Description

About three things I was absolutely positive. First, Edward was a vampire. Second, there was a part of him and I didn't know how dominant that part might be that thirsted for my blood. And third, I was unconditionally and irrevocably in love with him. In the first book of the Twilight Saga, internationally bestselling author Stephenie Meyer introduces Bella Swan and Edward Cullen, a pair of star-crossed lovers whose forbidden relationship ripens against the backdrop of small-town suspicion and a mysterious coven of vampires. This is a love story with bite. (less)

Reviews from Amazon

This is exactly what I wanted, I'm so glad I ended up buying it. My set is perfect, I don't have bent in edges like others have mentioned. I'm going to have to be quite careful with this set, the white is so gorgeous in person. If anybody reading this is skeptical about buying this collection, I say go for it! It's a great price for what you're getting. This is truly a wonderful gift for any Twilight fan.

Written by Amazon Customer
Subjectivity 0.71
Polarity 0.39

The Kindle set of books was fine, until Breaking Dawn. The whole part of the birth and Jacobs imprinting were missing from the book. Instead they repeated the Bella chapter twice. So it went directly into the chapter of the burning. I own the hard cover books and this is not an issue in them. Needless to say, I was unhappy with this purchase.

Written by Farf Dawg
Subjectivity 0.60
Polarity -0.03

Figure 8. Screenshot 2 of 4 of Individual Book Page (No User Logged In)

Misc. Details

Paperback, 498 pages

Link to Purchase

amazon

Reviews from Barnes and Noble

I love having the full matching box set to add to my library. A part of me wishes I would have stopped by half price books to pick up the novels but it was great having it delivered and matching for 30 some odd dollars. One star off because the back page (the red side) chips away easily. I have only held them and bookmarked them and the back looks worn. Otherwise, I'm happy with my purchase.

Written by Desiree
Subjectivity 0.47
Polarity 0.23

Still reading Twilight. Also love HYPNOTIC by Maria E. another great CRAZE! LOVE LOVE LOVE

Written by Anonymous
Subjectivity 0.63
Polarity 0.60

I had heard about this book from one of my sister's friends and I never thought anything of it. Then two of my closest friends started RAVING about this book so I decided to look it up. I read the first chapter online and was hooked. I quickly bought the first one and I could not put it down. You really grow to love and care for Bella, Edward, and the whole Cullen family. They each have their own personalities which you also grow to love. Twilight is exactly my type of heroine!

Written by FairyDust90
Subjectivity 0.45
Polarity 0.30

I loved reading this book! It was unique and the story was fantastic.

Written by theReader278

Figure 9. Screenshot 3 of 4 of Individual Book Page (No User Logged In)

I loved reading this book! It was unique and the story was fantastic.

Written by theReader278
Subjectivity 0.90
Polarity 0.55

Reddit Posts About This Book

How The Outsiders, Harry Potter and The Hunger Games transformed YA fiction https://www.washingtonpost.com/entertainment/books/how-the-outsiders-harry-potter-and-the-hunger-games-transformed-ya-fiction/2018/12/24/525157da-03c8-11e9-9122-82e98f91ee6f_story.html

Written by largeheartedboy
Subjectivity 0.00
Polarity 0.00

What books have you read that have a great concept but the story is meh?
https://www.reddit.com/r/books/comments/aitcsp/what_books_have_you_read_that_have_a_great/

Written by jeffoh
Subjectivity 0.75
Polarity 0.80

The Books That Wouldnt Die: Theyre alive, despite being rebutted, criticized, and cast out of the disciplines from which they came!
https://www.chronicle.com/article/The-Books-That-Wouldn-t-Die-245879?key=yCOpnBLrqCSy_JORBofCgEAHJ5yaQsqWlc_SIQBRY3HYRT5ZkIZvC2-IGcR6WYk5GTovWR3djVkoRmluUIA3VVBJavOU2ejOdmpIcnUwb2JlWmgwaw

Written by davideiss66
Subjectivity 0.40
Polarity 0.12

Information on The Wheel of Time Amazon TV Series <https://www.comingsoon.net/tv/news/1041027-amazons-wheel-of-time-uta-brisewitz-to-direct-first-two-episodes>

Figure 10. Screenshot 4 of 4 of Individual Book Page (No User Logged In)

Rating

BookBrain Averaged Internet Rating: 4.44

This rating is based on a weighted sum of ratings and reviews pulled from multiple other websites and sources.

Critic Reviews From iDreamBooks

Source	Date	Review Excerpt
Guardian	27 Jun 2014	I can read this book over and over again. From the very beginning until the end J.K. Rowling has me gripped! There is never a dull moment... Full Review
BookPage	1 Oct 1998	Designated for ages 8 to 12, but written for anyone who loves a good tale well told, this is a book to engage the mind and grab the heart . . . and J.K. Rowling is a writer to watch and remember. Full Review
About.com	12 Feb 2018	Later books in the series would be more intricate, more involved, contain more moments of sheer terror and sharper social satire. But this book still remains one of my favorites in the series, partly for its warmth, partly for its mystery, partly for some of its marvelous lines. Full Review
Tor	14 Jun 2011	It is hardly believable that Harry would forget the cloak, but this plot device is necessary to send Harry to the Forbidden Forest where he first faces true peril. Despite this flaw, adults and children will enjoy the underlying

Similar books

Tweets Mentioning This Book

Save

Figure 11. Screenshot of Individual Book Page with User Logged In and Save Button Below

Book Title

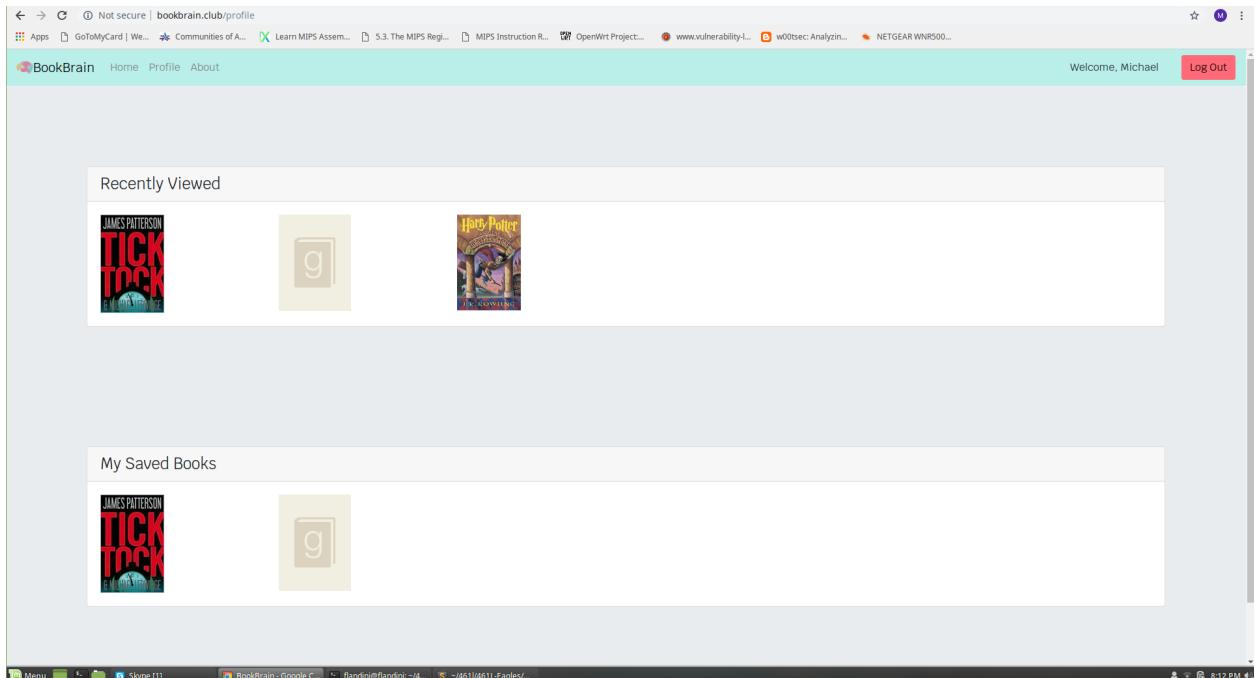


Figure 12. Screenshot of User Profile Page Showing Three Recently Viewed Books and Two Saved Books

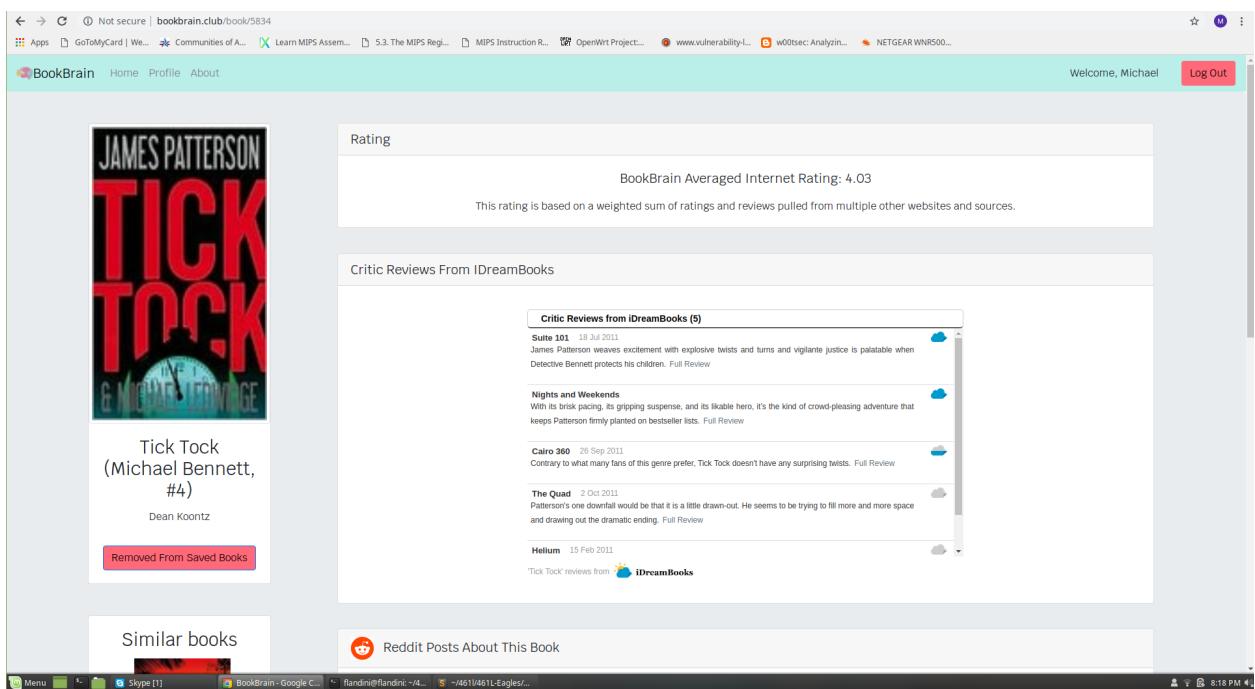


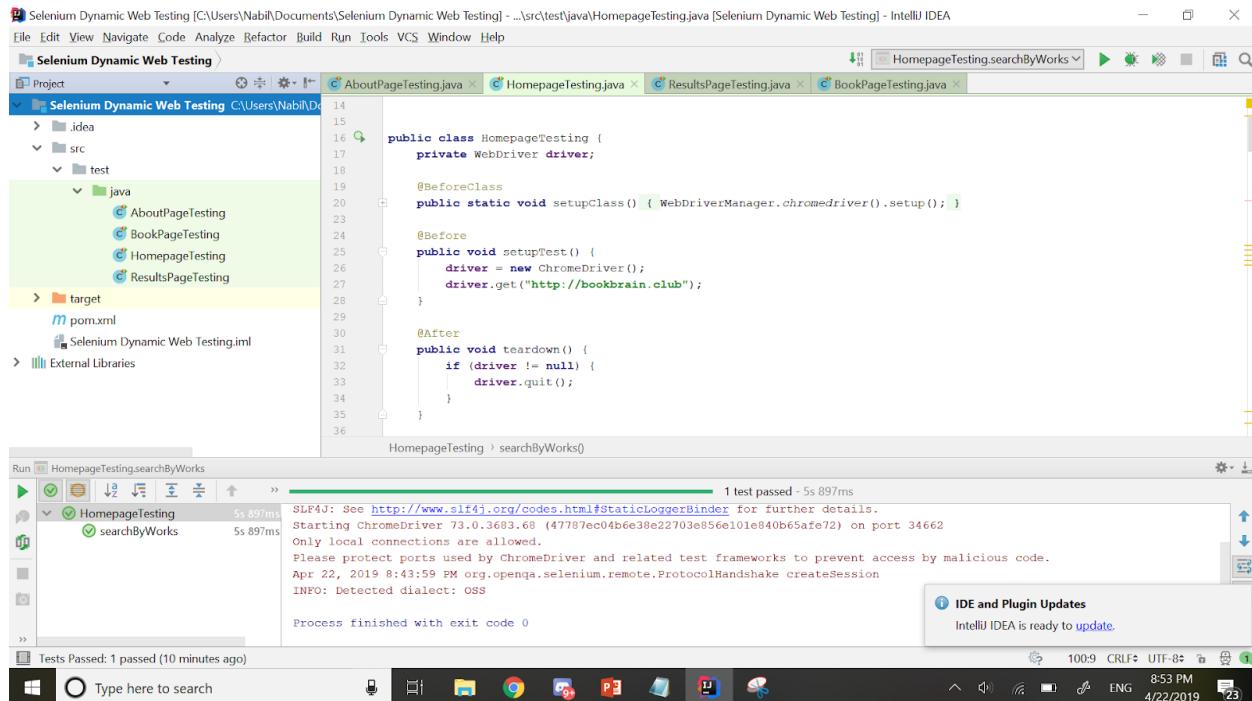
Figure 13. Screenshot of Individual Book Page with ‘Remove From Saved Books’ Button and Critic Review Widget

TOOLS, SOFTWARE, AND FRAMEWORKS

In addition to all of the tools, software, and frameworks used in previous phases, we have added only a few more tools to our web application. In this phase, we have used Maven and Selenium for testing. We have also added sentiment to each review by using TextBlob. To manage our project, we have created an issue board on Trello. We have used Google App Engine to hold our web app and Google Cloud SQL to hold our Mysql database.

TESTING

In the previous phases we had only incorporated static tests, but in this phase we utilized Selenium, a framework tool for testing websites, to incorporate dynamic tests. More specifically, we utilized Selenium WebDriver to load our web application and interact with its web elements. Figure 14 shows the project file structure and the code for creating and destroying a web driver. To organize our tests, we created a test java file for each major web page on our website: HomepageTesting.java, ResultsPageTesting.java, and BookPageTesting.java, AboutPageTesting.java. In the following paragraphs, we will describe each test file in detail.



The screenshot shows the IntelliJ IDEA interface with the following details:

- Project Structure:** The left sidebar shows the project structure under "Selenium Dynamic Web Testing". It includes a .idea folder, a src directory containing test and main Java files, a target folder, pom.xml, and Selenium Dynamic Web Testing.iml.
- Code Editor:** The main window displays the content of the `AboutPageTesting.java` file. The code defines a class `HomepageTesting` with methods for setup and teardown of the WebDriver.
- Run Tab:** The bottom tab bar shows the run configuration for `HomepageTesting` with the test `searchByWorks`. The status indicates "1 test passed - 5s 897ms".
- Output Tab:** The right side of the interface shows the output of the test execution, including logs from SLF4J and ChromeDriver, and a message about detected dialect: OSS.
- Status Bar:** The bottom right corner shows system information: 100:9 CRLF, UTF-8, 8:53 PM, ENG, 4/22/2019, and a battery icon.

Figure 14. Test Project Structure and WebDriver Code

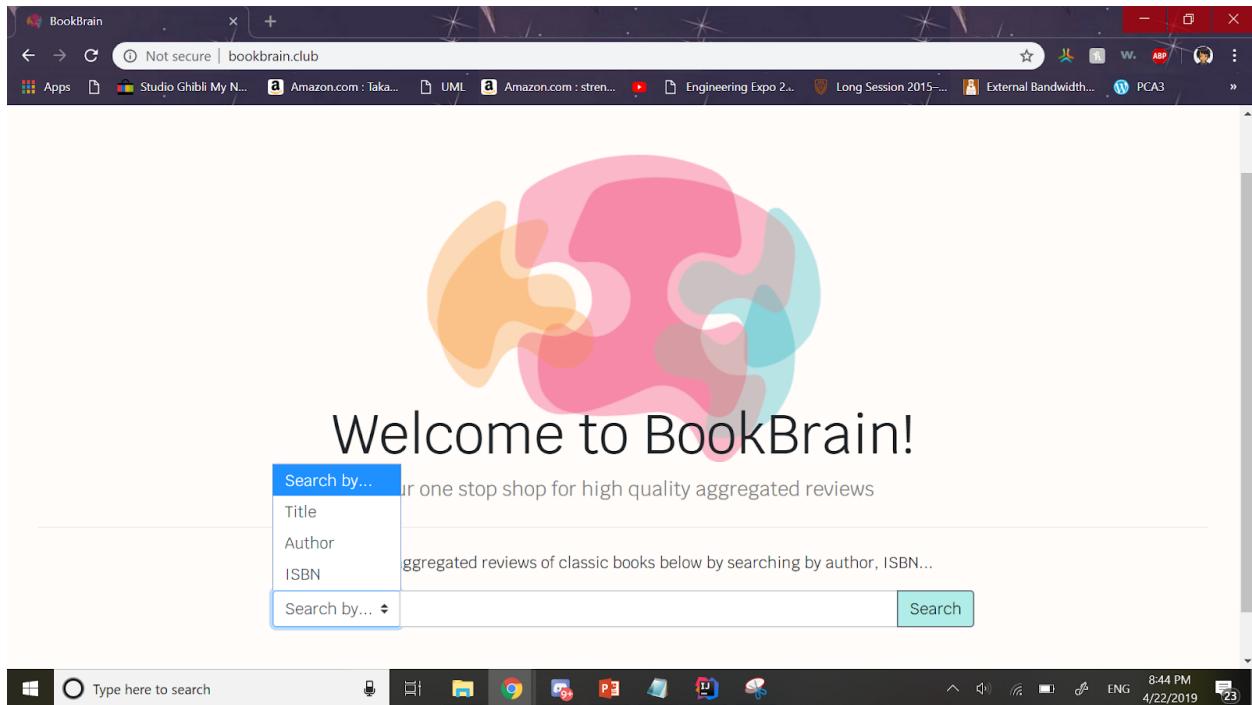


Figure 15(a). Search By Element Test

```

@Test
public void searchByWorks() {
    WebElement searchOptions = driver.findElement(By.id("searchOptions"));
    Select searchSelect = new Select(searchOptions);

    assertEquals( expected: "Search by...", searchSelect.getFirstSelectedOption().getText().toString());

    searchSelect.selectByVisibleText("Title");
    assertEquals( expected: "Title", searchSelect.getFirstSelectedOption().getText().toString());

    searchSelect.selectByVisibleText("Author");
    assertEquals( expected: "Author", searchSelect.getFirstSelectedOption().getText().toString());

    searchSelect.selectByVisibleText("ISBN");
    assertEquals( expected: "ISBN", searchSelect.getFirstSelectedOption().getText().toString());

    searchSelect.selectByVisibleText("Search by...");
    assertEquals( expected: "Search by...", searchSelect.getFirstSelectedOption().getText().toString());
}

```

Figure 15(b). Search By Element Test Code

In `HomepageTesting.java`, we made tests to ensure that every clickable web element redirects the user to the correct link. Most of our detailed testing for this webpage went into the search form. The search options were tested for responsiveness when requested to change, seen in Figure 15(a) and Figure 15(b). The autocomplete function of the form was tested for correctness, such

as ensuring it did not offer title suggestions when a user typed in numbers with the ISBN search by option selected.. It was also tested to ensure it offered at most 10 suggestions, because we wouldn't want users to be given an excessively long list.

In ResultsPageTesting.java, we mainly focused on three things. Results for a search are displayed, results are clickable, and the pagination buttons on the bottom of the page are functioning as they should. For web elements in general, we need to find the web element tag that will allow Selenium's webdriver to interact with it. This is usually done by using the inspect option on Google Chrome, and then applying the ID given in our code, as seen in Figure 16(a) and Figure 16(b).

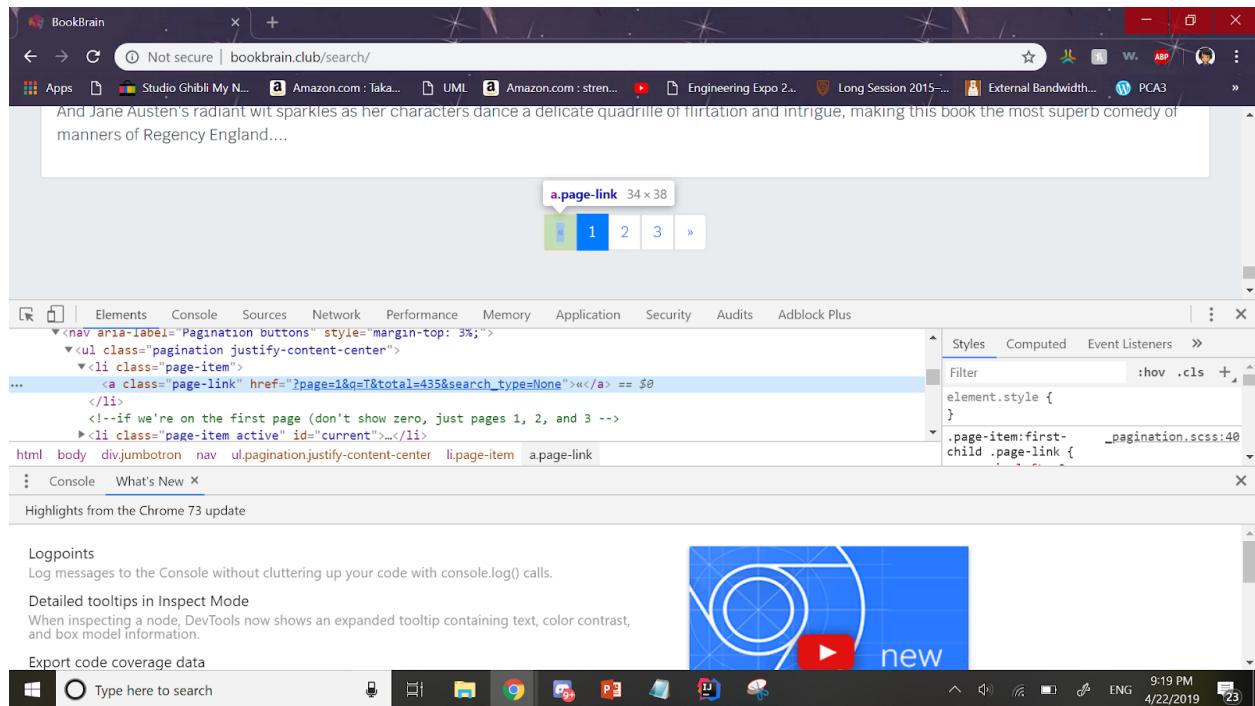


Figure 16(a). First Page Pagination Test and ID

```

@Test
public void paginatorClickingFirstPageWorks() {
    WebElement paginator = driver.findElement(By.cssSelector("ul.pagination.justify-content-center"));
    WebElement lastPageButton = paginator.findElements(By.tagName("li")).get(0);

    //make sure button I am clicking is the << button
    assertEquals(expected: "<<", lastPageButton.findElement(By.tagName("a")).getText().toString());
    lastPageButton.click();
    driver.manage().timeouts().implicitlyWait(3, TimeUnit.SECONDS);

    paginator = driver.findElement(By.cssSelector("ul.pagination.justify-content-center"));
    String pageItShouldBe = paginator.findElements(By.tagName("li")).get(1).getText().toString();
    String currentPage = driver.findElement(By.cssSelector("li#current.page-item.active")).getText().toString();

    assertEquals(pageItShouldBe, currentPage);
}

```

Figure 16(b). First Page Pagination Test and ID Implementation

In BookPageTesting.java, we made tests to ensure that links and buttons worked just like the previous files. In this case, it was specifically the buy button and similar books icon as seen in Figure 8 and Figure 9 respectively. On top of that, however, we created further tests to ensure that the iDreamBooks widget (seen in Figure 13) was working and interactive.

Finally, in AboutPageTesting.java, we made sure the hyperlinks listed would link to the correct website. This was simple as we simply had to compare the Selenium webdriver's current URL after clicking the hyperlink with what we expect it to be, and using AssertTrue to ensure the test passes the way we want to.

By breaking the web application into its four major web components and testing the dynamic elements in each of those major web components, we were able to do a full test coverage of the different things the user could interact with. In total, we performed 26 different Selenium Dynamic Webpage tests on BookBrain. Below are the figures representing the passed test cases.

The screenshot shows the IntelliJ IDEA interface with the project 'Selenium Dynamic Web Testing' open. The 'HomepageTesting.java' file is selected in the editor. The code contains 13 test methods, all of which have passed. The run history at the bottom shows 'All 13 tests passed'.

```

    assertEquals(expected: "ISBN", searchSelect.getFirstSelectedOption().getText());
    searchSelect.selectByVisibleText("Search by...");
    assertEquals(expected: "Search by...", searchSelect.getFirstSelectedOption().getText());
}

@Test
public void autocompleteForSearchByIsSizeAtMost10() {
    WebElement searchBar = driver.findElement(By.id("autocomplete"));
    //type in T
    searchBar.sendKeys(...charSequences: "T");
    driver.manage().timeouts().implicitlyWait(3, TimeUnit.SECONDS);
}

```

Run HomepageTesting
All 13 tests passed - 1m 7s 401ms

Tests Passed: 13 passed (moments ago)

Figure 17. HomepageTesting.java 13 Test Cases All Passed

The screenshot shows the IntelliJ IDEA interface with the project 'Selenium Dynamic Web Testing' open. The 'ResultsPageTesting.java' file is selected in the editor. The code contains 6 test methods, all of which have passed. The run history at the bottom shows 'All 6 tests passed'.

```

    WebElement bookResults = driver.findElement(By.cssSelector("div.list-group"));
    bookResults.findElement(By.cssSelector("a.list-group-item.list-group-item-action")).click();

    String currentURL = driver.getCurrentUrl();
    assertEquals(expected: "http://bookbrain.club/book/1", currentURL);
}

@Test
public void paginatorDefaultIsPage1() {
    WebElement currentPage = driver.findElement(By.cssSelector("li#current.page-item.active"));
    String number = currentPage.getText().toString();
    assertTrue(number.equals("1"));
}

```

Run ResultsPageTesting
All 6 tests passed - 39s 899ms

Tests Passed: 6 passed (moments ago)

Figure 17. ResultsPageTesting.java 6 Test Cases All Passed

Selenium Dynamic Web Testing [C:\Users\Nabil\Documents\Selenium Dynamic Web Testing] - ..\src\test\java\ResultsPageTesting.java [Selenium Dynamic Web Testing] - IntelliJ IDEA

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

Selenium Dynamic Web Testing > src > test > java > ResultsPageTesting

Project: Selenium Dynamic Web Testing C:\Users\Nabil\Documents\Selenium Dynamic Web Testing

test > java > ResultsPageTesting

ResultsPageTesting.java

```

public void resultsClickable() {
    WebElement bookResults = driver.findElement(By.cssSelector("div.list-group"));
    bookResults.findElement(By.cssSelector("a.list-group-item.list-group-item-action")).click();

    String currentURL = driver.getCurrentUrl();
    assertEquals(expected: "http://bookbrain.club/book/1", currentURL);
}

@Test
public void paginatorDefaultIsPage1() {
    WebElement currentPage = driver.findElement(By.cssSelector("li#current.page-item.active"));
    String number = currentPage.getText().toString();
    assertTrue(number.equals("1"));
}

```

Run BookPageTesting

All 2 tests passed - 16s 989ms

BookPageTesting 16s 989ms

- similarBooksIsClickable 6s 918ms
- buyButtonsClickable 10s 77ms

SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See <http://www.slf4j.org/codes.html#StaticLoggerBinder> for further details.
Starting ChromeDriver 73.0.3683.68 (47787ec04b6e38e22703e856e101e840b65afe72) on port 20520
Only local connections are allowed.
Please protect ports used by ChromeDriver and related test frameworks to prevent access by malicious code.
Apr 22, 2019 10:50:52 PM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: OSS
Starting ChromeDriver 73.0.3683.68 (47787ec04b6e38e22703e856e101e840b65afe72) on port 19000
Only local connections are allowed.
Please protect ports used by ChromeDriver and related test frameworks to prevent access by malicious code.
Apr 22, 2019 10:50:59 PM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: OSS

Process finished with exit code 0

Tests Passed: 2 passed (moments ago)

Type here to search

57:87 CRLF: UTF-8 10:51 PM 4/22/2019

Figure 17. ResultsPageTesting.java 2 Test Cases All Passed

Selenium Dynamic Web Testing [C:\Users\Nabil\Documents\Selenium Dynamic Web Testing] - ..\src\test\java\AboutPageTesting.java [Selenium Dynamic Web Testing] - IntelliJ IDEA

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

Selenium Dynamic Web Testing > src > test > java > AboutPageTesting

AboutPageTesting.java

```

goodReads2Link.click();

String currentURL = driver.getCurrentUrl();
assertEquals(expected: "https://www.goodreads.com/", currentURL);

@Test
public void iDreamBooksWorks() {
    WebElement iDreamBooksLink = phasesAndDocumentation.get(1).findElements(By.tagName("a")).get(2);

    assertEquals(expected: "IDreamBooks", iDreamBooksLink.getText().toString());

    iDreamBooksLink.click();

    String currentURL = driver.getCurrentUrl();
    assertEquals(expected: "https://idreambooks.com/", currentURL);
}

@Test
public void githubWorks() {
    WebElement githubLink = phasesAndDocumentation.get(2).findElement(By.tagName("a"));

    iDreamBooksWorks();
}

```

Run AboutPageTesting

All 5 tests passed - 59s 588ms

AboutPageTesting 59s 588ms

- iDreamBooksWorks 7s 839ms
- goodReadsSecondLinkW 9s 256ms
- barnesAndNoblesLinkW 27s 493ms
- goodReadsLinkWorks 8s 971ms
- githubWorks 6s 29ms

SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See <http://www.slf4j.org/codes.html#StaticLoggerBinder> for further details.
Starting ChromeDriver 73.0.3683.68 (47787ec04b6e38e22703e856e101e840b65afe72) on port 26227
Only local connections are allowed.
Please protect ports used by ChromeDriver and related test frameworks to prevent access by malicious code.
Apr 22, 2019 10:41:01 PM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: OSS

Tests Passed: 5 passed (2 minutes ago)

Type here to search

81:17 CRLF: UTF-8 10:43 PM 4/22/2019

Figure 17. AboutPageTesting.java 5 Test Cases All Passed

FINAL REFLECTIONS

The biggest difficulties we faced with launching BookBrain were communication, infrastructure, collaboration, and large scrape times. Initially, we had problems with team communications such as setting due dates and meeting times, and there were also problems with collaboration on GroupMe. After meeting with Dr. Eberlein, we fixed these issues by using a Trello board to keep track of all issues and deadlines and by setting more in person group meetings.

Another collaboration and infrastructure issue that we ran into was our Github repository usage. Except for the end of the third phase, we all checked code into the repository by directly pushing commits to the main master branch. Sometimes we would end up overwriting or deleting others' commits whether by force pushing or just honest git mistakes. However, we realize that we should have forked the repo and used pull requests; this is something that we would do differently starting over.

We also faced difficulties in keeping data compatible and correct between different formats and people. For instance, it was difficult synching up everyone's databases with the current database schema in the Github repository. Even after getting everyone's databases on the same page, it was tough loading data from CSV files into MySQL from scrapers that came from multiple people using different encodings and formats. If we could start development over, we would probably use something like Docker or Vagrant to enforce the similarity of development environments across team members and also define a specific format for data before distributing work.

Another one of our largest issues was dealing with large scrape times and incomplete scrape data. For this project, we had to scrape a lot of data from several different sites. We started out with scraping data from GoodReads and then using this scraped data to help scrape other sites. For example, after pulling book ISBN's from GoodReads, we used these ISBN's for our critic review widget and to reference which books we needed to scrape from other sites. However, it took a while to scrape data from any of the sites we specified, and if the scraped data from one of

the sites (especially GoodReads) was messed up, it meant having to re-scrape that site and others due to coupling of our data. Because of this coupling and the amount of sites that had to be scraped with each addition of new books, we did not get as many books scraped as we had hoped to.