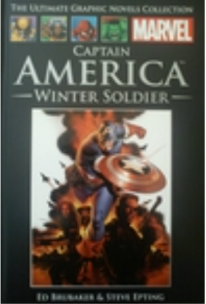


Feature 1

Recommendation system based on similar books. This feature adds a section called “Similar Books” to each book. This section recommends books to the user which are similar to the book that the user is currently looking at.



Captain America: Winter Soldier (The Ultimate Graphic Novels Collection: Publication Order, #7)

ISBN: N/A | Author: Ed Brubaker

★★★★★

Description

The questions plaguing Captain America's dreams and memories have been answered in the most brutal way possible. And in the wake of this brutality, General Lukin makes his first all-out assault - tearing open old wounds and threatening to make new scars that will never heal!




Publisher: Hachette Partworks Ltd. | Release Year: 2012

Unavailable as e-book


Write Review

Reviews >


Similar Books




Similar Books




[The Incredible Hulk: Silent](#)




[The New Avengers, Volume 4:](#)




[Doctor Strange: The Oath](#)



[Captain America: Man Out of](#)



[The Invincible Iron Man,](#)



[Thor, by J. Michael Straczynski,](#)

How it works (Feature 1):

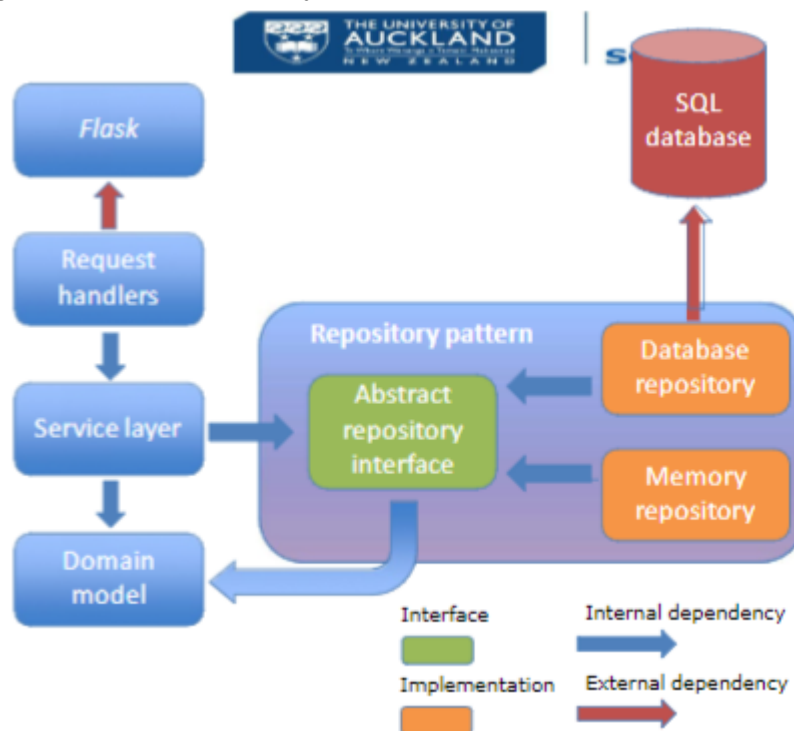
In the comic_books_excerpt.json file, the book data includes a dictionary of similar books. We have used this to create a recommendation of similar books for our users.

```
"similar_books": ["17277814", "13537835", "40452", "22812127", "13528792", "17277783",
```

Then this data is extracted by using jsondatareader.py

```
for book_id in (book_json['similar_books']):  
    book_instance.add_similar_book(int(book_id))
```

The book_id for the similar books is added to the book's similar_books, which is a list of integer book_ids. Repository pattern was used to implement this feature.



First, we created an abstract method in our AbstractRepository.

```
@abc.abstractmethod  
def get_similar_books(self):  
    raise NotImplementedError
```

Then we extended this method in the MemoryRepository.

```
def get_similar_books(self, book: Book):  
    matching = []  
    for book_id in book.similar_book:  
        book = self.get_book(book_id)  
        if book != None:  
            matching.append(book)  
    return matching
```

We made sure that when we're appending to the matching list, if we cannot find a book in our repository that matches the similar book's book_id, we will still keep that book_id in

book.similar_books. This is so the books can be more easily added in the future, when more books are added to our comic_books_excerpt.json file with book_id that matches in other books' similar_books list.

Then we create a method in our service.py, where the service layer interacts with the memory repository by the abstract repository interface to find a list of similar books.

```
def get_similar_books(book: Book, repo: AbstractRepository):
    similar_books = repo.get_similar_books(book)
    return similar_books
```

Books.py then uses the method by passing in a list of books as services.get_similar_books() in render_template(). The user is then navigated to the book info page.

```
@books_blueprint.route('/book_info', methods=['GET'])
def book_info():
    if (session.get('logged_in')==True):
        user_name = (session.get('user_name'))
    else:
        user_name = None
    form = BookSearch()
    book_id = int(request.args.get('book_id'))

    return render_template(
        'book_info.html',
        form=form,
        user_name=user_name,
        book=services.get_book(book_id, repo.repo_instance),
        similar_books=services.get_similar_books(services.get_book(book_id), repo.repo_instance),
        home_url=url_for('home_bp.home'),
        list_url=url_for('books_bp.list_book'),
        numbers={"numbers": range(5)}
    )
```

Feature 2

User review slideshow, The user review slideshow is just an appealing addition to the user review area of our website. It allows the user to easily browse through previous reviews and even allows for users to rate the books.



The Switchblade Mamma

ISBN: N/A | Author: Lindsey Schussman



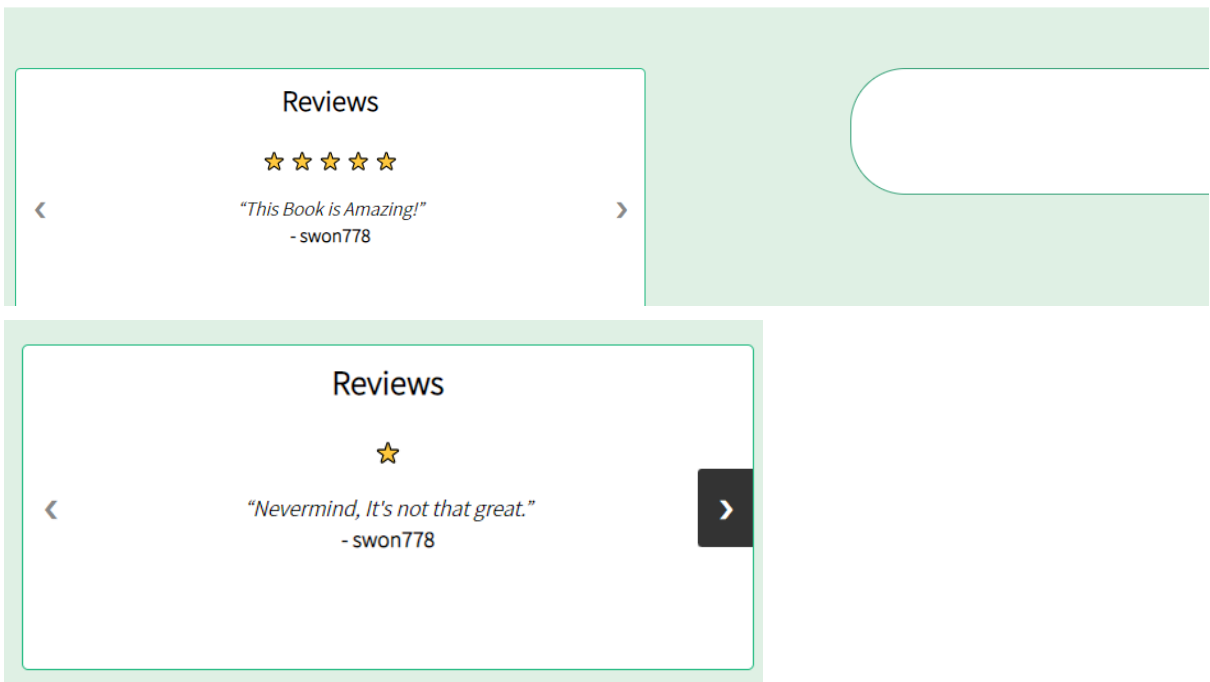
Description

Lillian Ann Cross is forced to live the worst nightmare world. Her life was abruptly turned upsidedown forever and bets are made upon their bloodshed.Lillian is forced Lillian finds difficulty grasping her new functions. As she procedure which has taken the lives of a few before he strengthen her bones, giving her strength and an upper

Pulisher: N/A | Release Year: None

Available as e-book!

Write Review



How it works (Feature 2):

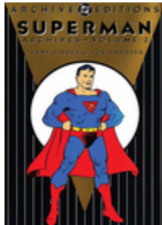
When the user clicks on a book they would like to view, the user is rendered to a page where the information for a single book is displayed. When the page is rendered in, it is also passed in a Book which we use to display its information on the web page using django notations.

```
<div class="review-display">
  <h2>Reviews</h2>
  {% for review in book.reviews %}
    <div class="review-slides">
      <p style="padding-bottom: 20px;">
        {% for number in numbers['numbers'] %}
          {% if number< review.rating %}
            ★
          {% endif %}
        {% endfor %}
      </p>
      <q>{{ review.review_text }}</q>
      <br><strong>- {{review.user.user_name}}</strong>
    </div>
  {% endfor %}
  <a class="prev-review" onclick="plusSlides(-1)">⏮</a>
  <a class="next-review" onclick="plusSlides(1)">⏭</a>
</div>
```

Feature 3

“I’m Feeling Feature”, this feature adds a random book that changes on page refresh. This feature was inspired by the “I’m Feeling Lucky” feature on Google where clicking on the “I’m Feeling Lucky” would take the user to a random site from what they’ve searched

🥒 "I'm Feeling Lucky!" 🥒



Superman Archives, Vol. 2

Author: Jerry Siegel, Joe Shuster

Description

These are the stories that catapulted Superman into the spotlight as one of the world's premier heroes of fiction. These volumes feature his earliest adventures, when the full extent of his powers was still developing and his foes were often bank robbers and crooked politicians.

On refresh:

🥒 "I'm Feeling Lucky!" 🥒



Doctor Strange: The Oath

Author: Brian K. Vaughan, Marcos Martin

Description

Doctor Stephen Strange embarks on the most important paranormal investigation of his career, as he sets out to solve an attempted murder - his own! And with his most trusted friend also at death's door, Strange turns to an unexpected corner of the Marvel Universe to recruit a new ally.

How it works (Feature 3):

In our website utilities, we added a method to retrieve a random book from our repository.
Under folder Library -> Utilities -> services.py

```
def get_random_book(repo: AbstractRepository):  
    books = repo.get_all_books()  
    book_count = len(books)  
    random_ids = (randrange(book_count))  
  
    book = books[random_ids]  
  
    return book
```

Under folder Library -> Utilities -> utilities.py, we use get_random_book to interact with our repository and get a random book from our repository.

```
def get_selected_book():  
    book = services.get_random_book(repo.repo_instance)  
    return book
```

Then the random selected_book will be passed on when rendering the template for the homepage. The book will be randomized everytime the user refreshes the page.

```
@home_blueprint.route('/', methods=['GET'])  
def home():  
    if (session.get('logged_in')==True):  
        user_name = (session.get('user_name'))  
    else:  
        user_name = None  
    form = BookSearch()  
    return render_template(  
        'home.html',  
        form= form,  
        user_name=user_name,  
        selected_book = utilities.get_selected_book(),  
        home_url = url_for('home_bp.home'),  
        list_url = url_for('books_bp.list_book'),  
        book_url = url_for('books_bp.book_info'),
```