Megan Webb
HCI 584 | Summer 2024

Link

Developer Documentation for the Weekly Top Books app

Overview

The NYT Best Sellers Project is a web application that allows users to view the top New York Times best-selling books in various genres. Users can select a genre from a dropdown menu to see the list of best sellers, including the title, author, a brief description, and a link to purchase the book on Amazon.

Final Planning Specs

Project Description

The project aims to create a user-friendly app that displays the most popular books each week across various genres. The app aggregates data from the New York Times Books API and presents it in a clean, modern interface.

Features

- Genre Selection: Dropdown menu for selecting a genre.
- Book Listings: Display top books for the selected genre, including title, author, description, and a link to purchase on Amazon.
- Published Date: Display the date for which the list of books is relevant.

Technology Stack

- Frontend: HTML, CSS, Bootstrap
- Backend: Flask (Python)
- API: New York Times Books API

Installation, Deployment, and Administration

Prerequisites

- Python 3.x
- New York Times API key

Setup Instructions

1. **Clone the Repository:**
   ```
   git clone https://github.com/yourusername/nyt-best-sellers.git
   cd nyt-best-sellers
   ```

2. **Install Dependencies:**

```
pip install flask requests
```

3. **Set Up API Keys**
   Create a file named API_keys.py in the project directory and add your New York Times API key and URL: # API_keys.py

   ```
   NYT_API_KEY = 'your_nyt_api_key'
   NYT_API_URL = 'https://api.nytimes.com/svc/books/v3/lists.json'
   ```

4. **Run the Application:**

   ```
   python app.py
   ```

5. **Access the Application**
   Open your web browser and go to http://127.0.0.1:5000/.

User Interaction and Flow

User Interaction and UX Flow

1. Homepage: Users are greeted with a dropdown menu to select a genre.
2. Genre Selection: Users select a genre from the dropdown menu.
3. Book Listings: The app displays the top books for the selected genre, including titles, authors, descriptions, and Amazon purchase links.
4. Published Date: The date for which the list of books is relevant is displayed.

Walkthrough of Code

app.py

- **Imports and Configuration:**

```
from flask import Flask, render_template, request
import requests
from API_keys import NYT_API_KEY, NYT_API_URL

app = Flask(__name__)
```

- Function to Fetch Best Sellers List

```
def get_best_sellers_list(list_name):
    params = {
        'api-key': NYT_API_KEY,
        'list': list_name
    }
    response = requests.get(NYT_API_URL, params=params)
    if response.status_code == 200:
        results = response.json().get('results', [])
```

```
        if results:
            date = results[0].get('published_date', '')
        else:
            date = ''
        return results, date
    return [], ''
```

- **Route to Handle Genre Selection and Display Results**

```
@app.route('/', methods=['GET', 'POST'])
def index():
    genres = ["combined-print-and-e-book-fiction", "combined-print-and-e-book-nonfiction",
     "hardcover-fiction", "hardcover-nonfiction", "trade-fiction-paperback", "mass-market-paperback",
     "paperback-nonfiction", "e-book-fiction", "e-book-nonfiction", "hardcover-advice",
     "paperback-advice", "advice-how-to-and-miscellaneous", "chapter-books",
     "childrens-middle-grade", "childrens-middle-grade-e-book", "childrens-middle-grade-hardcover",
     "childrens-middle-grade-paperback", "paperback-books", "picture-books", "series-books",
     "young-adult", "young-adult-e-book", "young-adult-hardcover", "young-adult-paperback",
     "animals", "audio-fiction", "audio-nonfiction", "business-books", "celebrities",
     "crime-and-punishment", "culture", "education", "espionage",
     "expeditions-disasters-and-adventures", "fashion-manners-and-custo...
    selected_genre = request.form.get('genre') if request.method == 'POST' else genres[0]
    best_sellers, published_date = get_best_sellers_list(selected_genre)
    return render_template('index.html', genres=genres, best_sellers=best_sellers,
     selected_genre=selected_genre, published_date=published_date)

if __name__ == '__main__':
    app.run(debug=True)
```

index.html

- **HTML Structure and Bootstrap Integration**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>NYT Best Sellers</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
    <link rel="stylesheet"
     href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
</head>
<body>
    <div class="container">
    <h1>New York Times Best Sellers</h1>
    <form method="POST">
      <div class="form-group">
        <label for="genre">Select Genre:</label>
        <select class="form-control" name="genre" id="genre" onchange="this.form.submit()">
          {% for genre in genres %}
            <option value="{{ genre }}" {% if genre == selected_genre %}selected{% endif %}>
              {{ genre.replace('-', ' ').title() }}
            </option>
          {% endfor %}
        </select>
      </div>
    </form>
    <p>Published Date: {{ published_date }}</p>
    <h2>{{ selected_genre.replace('-', ' ').title() }}</h2>
    <ul class="list-group">
      {% for book in best_sellers %}
        <li class="list-group-item">
          <h5>{{ book.book_details[0].title }} by {{ book.book_details[0].author }}</h5>
          <p>{{ book.book_details[0].description }}</p>
          <a href="https://www.amazon.com/s?k={{ book.book_details[0].title | urlencode }}+{{ book.book_details[0].author | urlencode }}" target="_blank">Buy on Amazon</a>
        </li>
      {% endfor %}
    </ul>
  </div>
</body>
</html>
```

style.css

- **CSS for Styling**

```
body {
  font-family: 'Arial', sans-serif;
  margin: 0;
  padding: 0;
  background-color: #f4f4f4;
  color: #333;
}

.container {
  max-width: 800px;
  margin: 0 auto;
```

```
    padding: 20px;
    background-color: #fff;
    border-radius: 10px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
  }

  h1, h2 {
    text-align: center;
  }

  form {
    margin: 20px 0;
    text-align: center;
  }

  select {
    padding: 10px;
    font-size: 16px;
    margin-left: 10px;
    border: 1px solid #ddd;
    border-radius: 5px;
  }

  p {
    text-align: center;
    font-size: 16px;
    color: #666;
  }

  ul {
    list-style-type: none;
    padding: 0;
  }

  li {
    background-color: #f9f9f9;
    margin: 5px 0;
    padding: 10px;
    border: 1px solid #ddd;
    border-radius: 5px;
  }

  li:nth-child(even) {
    background-color: #f1f1f1;
```

```
}

h5 {
  margin: 0;
  font-size: 18px;
  color: #333;
}

a {
  color: #007bff;
  text-decoration: none;
}

a:hover {
  text-decoration: underline;
}
```

Known Issues

- Rate Limiting: The New York Times API may enforce rate limits, which can affect the app's ability to fetch data if too many requests are made in a short period.

- Error Handling: The app currently has basic error handling. More robust error handling could improve user experience by providing clear feedback on issues.

- Descriptions: Some book entries may not have descriptions, leading to empty description fields in the app.

Future Work

- Detailed Book Profiles: Expand the app to include more detailed book profiles with additional information like reviews, ratings, and related books.

- Enhanced Error Handling: Improve error handling to provide more user-friendly error messages and better logging for debugging.

- User Accounts: Implement user accounts to allow users to save their favorite books and genres.

- Mobile Optimization: Further optimize the app