



Team 10: Ctrl-Alt-DaHeat

Bao Ngoc Nguyen, Said Mazaheri, Kevin Li, Kieran Serra

UMass Amherst CS426 - Scalable Web Systems

March 28th, 2025



# The Project's Vision

**NewsBridge** is a news aggregator that collects articles from both liberal and conservative sources. Using NLP, it identifies similar stories and an LLM combines them into a single unbiased article, removing biases and presenting a balanced perspective.

Since Milestone 1 is to implement the frontend, key functionalities include:

- **Page Routing:** Implement routing to navigate between pages such as the homepage, articles page, etc.
- **Mock data:** For this initial phase, the frontend will display mock data simulating articles, user details, etc.
- **UI + Layout:** Utilize Tailwind CSS to style components and design responsive a UI
- **Modularization:** Utilize React components to create reusable, modular code, ensuring scalability

**Repository:** <https://github.com/ysmazaheri/NewsBridge>

**Milestone:** <https://github.com/ysmazaheri/NewsBridge/milestones>

**Issues:** <https://github.com/ysmazaheri/NewsBridge/issues>



# The Builders

Kevin Li

Role: Frontend Developer

Primary Contributions:

- Bias Rating Component
- Bookmark Component
- ArticleList Component
- Profile Page
- Bookmarked Page



Said Mazaheri

Role: Frontend Developer

Primary Contributions:

- Sign-Up, Sign-In, & Sign-Out Pages
- Reset-Password Page
- Open-Article Page
- Comment Section Component
- View Model + DTO Entities



Bao Nguyen

Role: Frontend Developer

Primary Contributions:

- Reusable Fundamental Components
- Display Comment Component
- Article Comment Section Component
- Article Preview Component
- Search Page



Kieran Serra

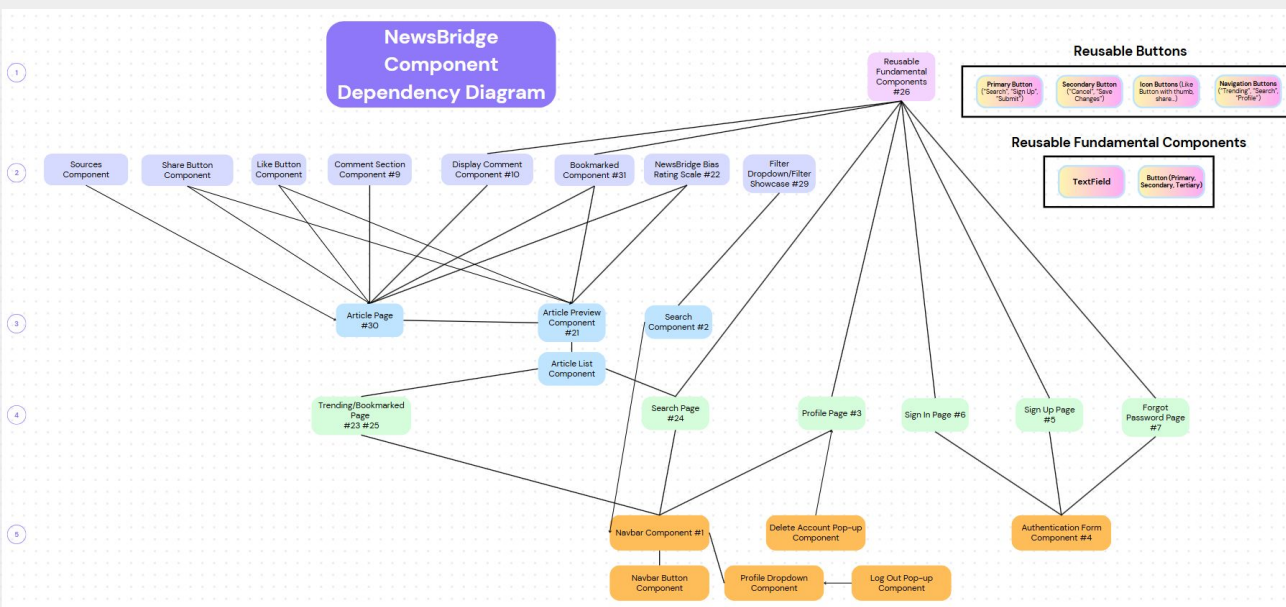
Role: Frontend Developer

Primary Contributions:

- Navbar Component
- Search Component
- Sources Component
- Filter/Dropdown Component
- Trending Article Page



# Software Architecture Overview



Here we have a diagram showcasing the component hierarchy of our UI. For example, for our **Bookmarked Page**, we utilize the **Navbar** component, as well as the **Article List** component (which themselves utilize other components)

The UI is divided into a reusable components hierarchy to ensure scalability

**Pages:** These are the highest-level components, serving as primary views and incorporating multiple mid-level components.

**Mid-Level Components:** Reusable building blocks like ArticleList, Navbar, and ArticlePreview, responsible for rendering content dynamically.

**Low-Level Components:** Fundamental UI elements such as BookmarkButton and BiasRatingScale, which handle specific user interactions.

**UI Communication Component Interaction:** Components communicate via props and callback functions, as well as useContext where necessary to avoid excessive prop drilling.

**Routing:** Uses React Router for navigating between different views, ensuring a seamless user experience.

# Historical Timeline

[illegible]



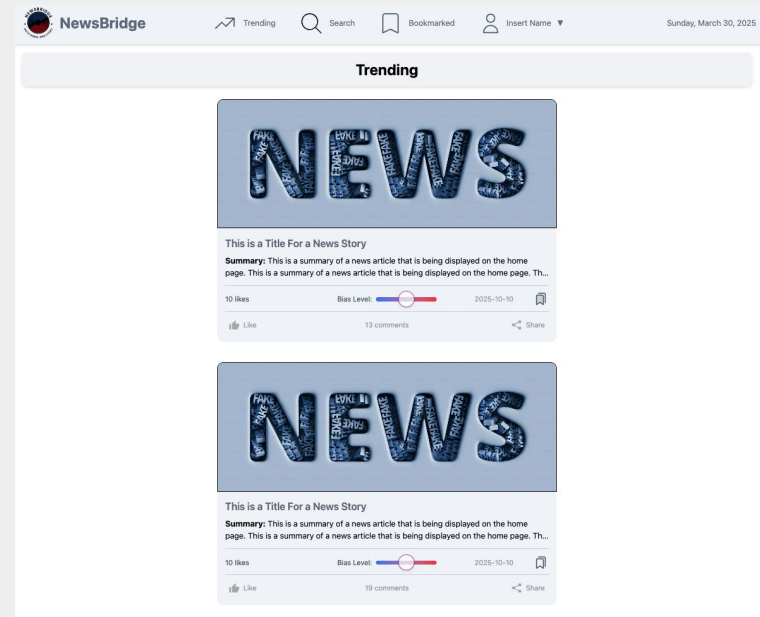
# Design and Styling Guidelines

**Color Scheme:** Uses TailwindCSS across components such as buttons and text areas for a smooth user experience. Primary actions (ex. Sign-in) use the primary color, while secondary actions (Ex. Cancel) use a secondary color, making navigation intuitive for user.

**Responsive Design:** Subtle button animations to provide users with feedback while avoiding discomfort for motion sensitive users. Utilizes Flexbox and Grid for a responsive layout across screen sizes.

**High Contrast Text:** Provides readability across backgrounds for users to get their data.

[Styling Guidelines Document](#)

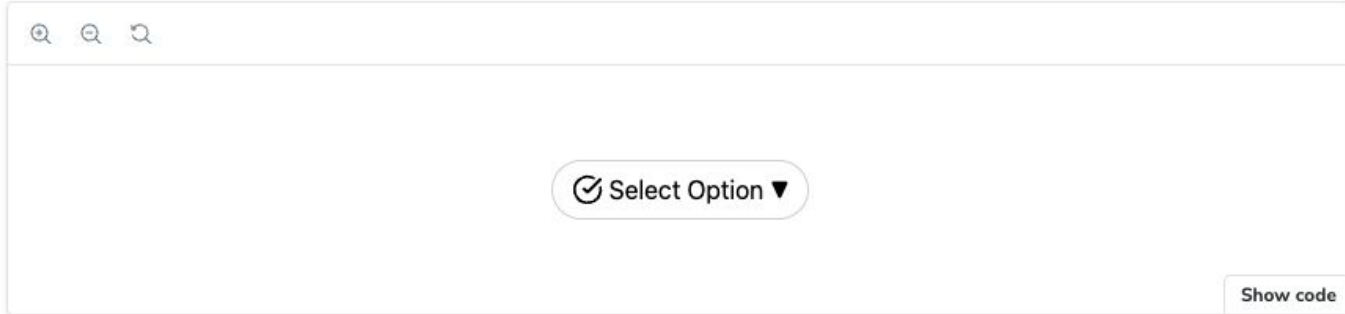


Simplistic modern layout, ensuring users can get exactly what they want without any over complications. We mainly went with a neutral color scheme with some red and blue mixed in to account for a U.S. user theme.

# Component Documentation

## Dropdown

Purpose: A reusable dropdown component that supports different styles (icons) and is used for selecting search criteria for articles.



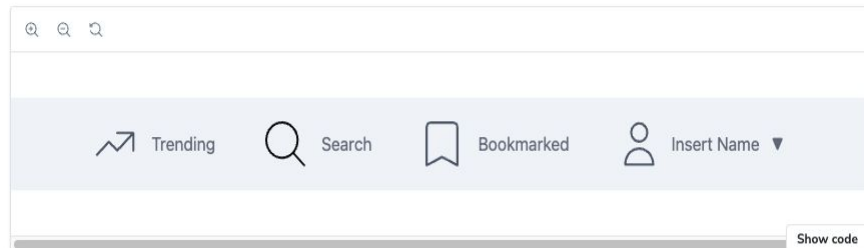
The screenshot shows a documentation interface for a 'Dropdown' component. At the top, there's a header 'Dropdown' and a paragraph describing its purpose: 'A reusable dropdown component that supports different styles (icons) and is used for selecting search criteria for articles.' Below this is a large rectangular box representing the component. Inside the box, on the left, are three magnifying glass icons. In the center, there is a rounded button with a checkmark icon and the text 'Select Option' followed by a downward arrow. In the bottom right corner of the box, there is a small button labeled 'Show code'.

[Link to Storybook](#)

# Component Documentation

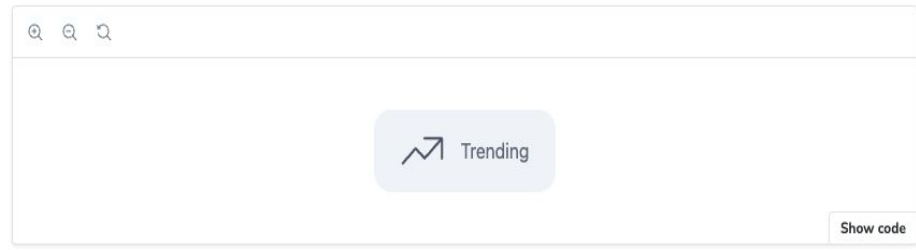
## Navbar

Purpose: A reusable navbar component that can be used to navigate between different sections of the application when user is logged in.



## NavbarButton

Purpose: A reusable button component that is used in the Navbar to allow users to navigate between pages.



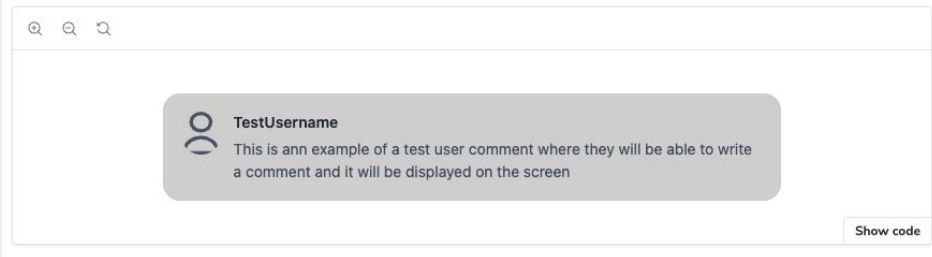
[Link to Storybook](#)



# Component Documentation

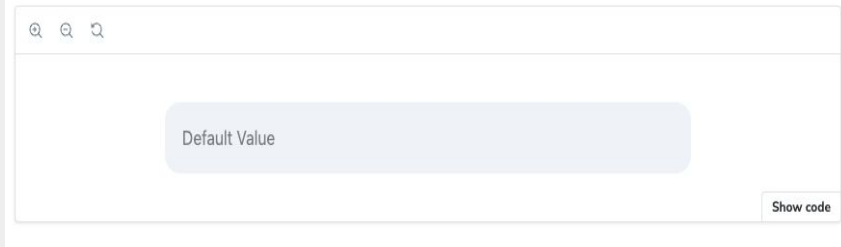
## CommentDisplay

Purpose: A component to display user comments on news articles. It shows the username, comment text, and optionally the user's profile image



## TextField

Purpose: A reusable text field component that can be used in various parts of the application such as login, sign up, and comment sections.



[Link to Storybook](#)

# Component Documentation

## ResetPassword

Purpose: Reset password component that allows users to reset their password by entering their email address.

### Reset Password

Please provide your email address to reset your password.

Reset Password

Remember your password? [Sign In](#)

## Search

Purpose: A search component that allows users to search for news articles based on key input, filter with various options, and sort by date or relevance.



Posted At ▼

Category ▼

Bias Rating ▼

Relevance ▼

Search

Show code

[Link to Storybook](#)

# Component Documentation

## SignIn

Purpose: A sign in page component that allows users to sign in to the application using their email and password.

### Welcome Back 🖐️

Welcome to **NewsBridge**, where diverse perspectives converge to deliver balanced, transparent news for an informed citizenry.

Sign In

Don't have an account? [Sign Up](#)

Forgot your password? [Reset Password](#)

## SignUp

Purpose: Sign up page component that allows users to sign up for the application using their email and password and includes dynamic user validation for password.

### Sign Up

Sign up now to enjoy a community of informed readers and enjoy balanced, transparent news from every side.

- ✖ At least 8 characters
- ✖ At least one uppercase letter
- ✖ At least one lowercase letter
- ✖ At least one special character

Sign Up

Already have an account? [Sign In](#)

[Link to Storybook](#)

# Performance Considerations

- **Pagination for Articles**
  - Reduces load by fetching a limited number of articles per request.
  - Improves response times and UI performance.
- **Search Strategy**
  - Fetch all articles and then filter
    - Speeds up client operations
    - Increases memory usage and initial load
  - Filter within the query
    - Reduces amount of api calls
    - Limits flexibility of searches
- **Using Context for User Data**
  - Minimize database queries by containing data across components in memory.
  - Increases app memory usage and overhead
- **Image Storage**
  - Will use only links instead of full images on the frontend
    - Reduces memory usage
    - Speeds up overall frontend
    - Can allow for dynamic image updates

# Assigned Work Summary - Kieran Serra

## 1. Navbar Component Development

- **Issue:** [Implement Navbar Component](#)
- **Closed:** Yes
- **Description:** Created the Navbar component with the following features:
  - Logo
  - Trending Button
  - Search Button
  - Profile dropdown with Profile and Logout
  - Dynamic Date Display
- [Commit Link](#)

## 3. Filter Dropdown Component

- **Issue:** [Implement Filter Dropdown](#)
- **Closed:** Yes
- **Description:** Developed filter dropdown components with options like category, date, bias rating, and relevance.
- [Commit Link](#)

## 5. Implement Trending Page

- **Issue:** [Implement Trending Page](#)
- **Closed:** Yes
- **Description:** Implement page that maps through articles and displays article preview component allowing users to look through list of articles
- [Commit Link](#)

## 2. Search Component Development

- **Issue:** [Implement Search Component](#)
- **Closed:** Yes
- **Description:** Developed search component allowing users to search and sort articles based on various filters and sort options. (Ex. Relevance, US News, etc.)
- [Commit Link](#)

## 4. Implement Navbar Scaling

- **Issue:** [Implement Navbar Scaling](#)
- **Closed:** Yes
- **Description:** Allow navbar to scale properly on all types of screens and prevent overflow.
- [Commit Link](#)

## 6. Implement Sources Component

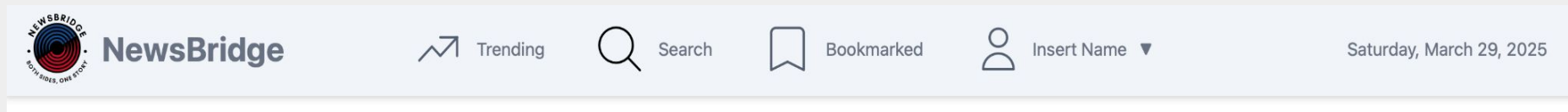
- **Issue:** [Implement Sources Component](#)
- **Closed:** Yes
- **Description:** Create dropdown for all sources used in article allowing users to understand where their article data is coming from.
- [Commit Link](#)

# Assigned Work Summary - Kieran Serra

## Pull Requests:

1. [PR: Implement Navbar with Dynamic Elements](#)
2. [PR: Implement Search Component and Dropdown Filter](#)
3. [PR: Fix Navbar Scaling](#)
4. [PR: Display Dynamic Password Requirements as user types](#)
5. [PR: Add Audience Bias Rating to DTO](#)
6. [PR: Implement Sources Component](#)
7. [PR: Add Home/Trending Page](#)

# Code & UI Explanation - Kieran Serra



## Navbar Component

**Purpose:** Global component used across all pages at the top of the screen, enabling navigation between pages.

**Integration:** Acts as an anchor for the app, connecting all pages and features together.

**Challenge:** Ensuring the navbar scales properly across different screen sizes without overflowing.

**Solution:** Used TailwindCSS for responsive design, eliminating unnecessary components in Navbar when minimized

**Design:** Component follows the app's style guidelines by utilizing the global primary and tertiary color scheme, ensuring it blends into the rest of the UI.

```
return 
<div>
  <nav className="flex items-center justify-center p-2 bg-tertiary text-primary shadow-md w-screen">
    /* Left Section */
    <div className="basis-1/3 flex justify-start items-center hidden xl:flex">
      <div className="flex flex-row items-center space-x-2 pl-2">
        <img
          src={logoSrc}
          alt="NewsBridge Logo"
          className="max-w-full max-h-full object-contain w-17 h-17"
        />
        <span className="text-3xl font-bold">NewsBridge</span>
      </div>
    </div>

    /* Center Section */
    <div className="basis-1/3 flex justify-center items-center w-full space-x-8 md:space-x-0">
      <NavbarButton
        value="Trending"
        textColor="text-primary"
        bgColor="bg-tertiary"
        borderColor="border-transparent"
        img="/trending.svg"
        height="h-16"
        handleClick={() => navigate("/home")}
      />
      <NavbarButton
        value="Search"
        textColor="text-primary"
        bgColor="bg-tertiary"
        borderColor="border-transparent"
        img="/searchicon.svg"
        height="h-16"
        handleClick={() => setShowSearch(!showSearch)}
      />
      <NavbarButton
        value="Bookmarked"
        textColor="text-primary"
        bgColor="bg-tertiary"
        borderColor="border-transparent"
        img="/bookmarkicon.svg"
        height="h-16"
        handleClick={() => console.log("Bookmarks clicked")}
      />
      <ProfileDropdown />
    </div>

    /* Right Section */
    <div className="basis-1/3 flex justify-end items-center px-5 hidden xl:flex whitespace-nowrap">
      <span className="text-md text-primary">{curDate}</span>
    </div>
  </nav>

  <div>{showSearch && <SearchComponent />}</div>
</div>

```



# Challenges and Insights - Kieran Serra

## Challenges:

- **Component Development:**
  - Building reusable components like the Navbar and dropdowns was challenging because they required ensuring they were not only functional but also adaptable across all pages.
- **Team Coordination:**
  - Ensuring that components I created fit well with work done by other team members. It wasn't just about coding my part and moving on, but required close coordination to ensure all components worked together.
- **Connecting Pieces Together:**
  - Once our team was finishing up on our respective components, we had to ensure we could wire everything together strategically to allow for a smooth user experience.

## Insights:

- **Effective Communication is Key:**
  - Through regular team meetings and clear communication, we made sure we were always on the same page allowing for smooth collaboration.
- **Prioritization of Development**
  - We learned the importance of effective prioritization. Understanding the dependencies between components and tasks allowed us to coordinate better, ensuring that the right features were built at the right time without blocking others from working.
- **Collaboration across Components**
  - We ensured that components were not only functional individually but interacted well with other components. This allowed us to craft a scalable clean UI.

# Future Improvements & Next Steps - Kieran Serra

## Enhanced Search Functionality:

- Implement **caching** to speed up the search feature, allowing articles to be retrieved more quickly.
- Implement Search **autocomplete** to allow users to get their desired articles faster.
- Allow for searches based on **user preferences** to let users get the content they are personally interested in.

## Performance Optimizations

- Implement **lazy loading** for non essential content. For ex. When a user is scrolling through an individual article, don't fetch the entire content when component mounts.
- Apply **caching** for article lists, so that calls to the database can become less frequent and allow the application to speed up even more.

# Assigned Work Summary - Bao Nguyen

## 1. Reusable Fundamental Components Development

- **Issue:** [Implement Reusable Fundamental Components](#)
- **Closed:** Yes
- **Description:** Develop reusable fundamental components (small pieces of components that is reused in other pages):
  - TextField: Text box for getting user input, taking in: default input message, color, box size, type - primary, secondary, ...
  - Button: Button pressed, taking in: text, color, size (w, h), action after pressed
- [Commit Link](#)

## 2. Display Comment Component Development

- **Issue:** [Implement Display Comment Component](#)
- **Closed:** Yes
- **Description:** Develop a small concise component that contains:
  - User Icon
  - User's Name
  - User's Comment
- [Commit Link](#)
- [Commit Link](#)

## 3. Article Comment Section Component

- **Issue:** [Implement Article Comment Section Component](#)
- **Closed:** Yes
- **Description:** Develop a comment section component that will be displayed at the bottom of each article. It will contain the following:
  - Text Input for each user to add their own comment
  - Submit Button for user to add comment
- [Commit Link](#)

## 4. Search Results Page Development

- **Issue:** [Implement Search Results Page](#)
- **Closed:** Yes
- **Description:** Implement Search Results Page with the following components / elements:
  - Implement Navbar Component #1
  - Search bar with search terms + filters: Implement Search Component #2
  - Implement Article Preview Component / Card #21
- [Commit Link](#)

# Assigned Work Summary - Bao Nguyen

## 5. Implement Article Preview Component / Card

- **Issue:** [Implement Article Preview Component / Card](#)
- **Closed:** Yes
- **Description:** Create an article preview card component which will be used in the home, bookmarked, and search results pages. The component should include the following fields:
  - Image
  - Article Title
  - Dynamic Sources + Source Images
  - Like Count (Displays current likes of article)
  - NewsBridge Bias Rating Component (our calculated bias rating for this article) Implement NewsBridge Custom Scales / Ratings #22
  - Age of the article (that the article was generated)
  - Bookmark status button Implement Bookmark Component #31
  - Like Button
  - Comment Section component Implement Article Comment Section Component #9 Implement the Display Comment Component #10
  - Share Button
- [Commit Link 1](#)

## Pull Requests Merged (PR):

1. [Implement Reusable Fundamental Components](#)
2. [Implemented article preview component card](#)
3. [Displayed user comment box](#)
4. [Completed the written comments section for user to enter comment](#)
5. [Minor fix folder design](#)
6. [Implement Reusable Fundamental Components](#)

# Code & UI Explanation - Bao Nguyen

```
1 import React, { useState } from "react"; // You, yesterday • Completed all basic structure of the...
2 import Bookmark from "../Bookmark";
3 import { NewsBridgeBiasScale } from "../BiasScale";
4 import LikeButton from "../Partials/LikeButton";
5 import ShareButton from "../Partials/ShareButton";
6 import { UnbiasedArticlePreviewViewModel } from "../.../entities/viewmodels/UnbiasedArticlePreviewVM";
7 import { useNavigate } from "react-router-dom";
8
9 Codeium: Refactor | Explain | Generate JS/Doc | X
10 const ArticlePreview: React.FC<UnbiasedArticlePreviewViewModel> = ({
11   title,
12   summary,
13   imageUrl,
14   likeCount: initialLikeCount,
15   biasRating,
16   daysAgo,
17   isBookmarked,
18   commentCount,
19   id,
20 }) => {
21   const [currentLikes, setCurrentLikes] = useState(initialLikeCount);
22   const [hasLiked, setHasLiked] = useState(false);
23   const navigate = useNavigate();
24
25   Codeium: Refactor | Explain | Generate JS/Doc | X
26   const handleClick = () => {
27     if (hasLiked) {
28       setCurrentLikes(currentLikes - 1);
29       setHasLiked(false);
30     } else {
31       setCurrentLikes(currentLikes + 1);
32       setHasLiked(true);
33     }
34   };
35
36   Codeium: Refactor | Explain | Generate JS/Doc | X
37   const handleClick = () => {
38     navigate(`/article/${id}`);
39   };
40 }
```

```
35 return (
36   <div className="max-w-2xl mx-auto bg-white rounded-xl font-sans">
37     {""}
38     {/* Article Image */}
39     <img src={imageUrl} alt="Article image" data-bbox="369 188 566 231"/>
40   </div>
41   <div className="h-full h-64 object-cover rounded-t-xl border border-black">
42     <div className="w-full h-64 bg-gray-200 rounded-t-xl border border-black flex items-center justify-center">
43       No image available
44     </div>
45   </div>
46   <div className="bg-tertiary rounded-b-xl p-4">
47     {/* Title */}
48     <h3 className="text-xl font-semibold text-primary mb-2 line-clamp-1 cursor-pointer hover:brightness-90">
49       {title}
50     </h3>
51
52     {/* Summary */}
53     <p className="text-base mb-3 line-clamp-2">
54       <span className="font-bold">Summary:</span> {summary}
55     </p>
56
57     {/* Line */}
58     <hr className="border-gray-300 mb-3"/>
59
60     <div className="flex items-center text-sm text-gray-600">
61       {/* Likes */}
62       <div className="w-1/3">
63         <span>{currentLikes} likes</span>
64       </div>
65     </div>
66   </div>
67 </div>
68 </div>
69 </div>
70 </div>
71 </div>
```

```
72 {/* Bias Level */}
73 <div className="w-1/3 flex items-center justify-center gap-2">
74   <span className="text-sm">Bias Level:</span>
75   <div>
76     <div>
77       className="flex items-center justify-center flex-shrink-0"
78       style={{ width: "120px", height: "24px" }}
79     >
80       <NewsBridgeBiasScale rating={biasRating} />
81     </div>
82   </div>
83
84   {/* Days Ago & Bookmark */}
85   <div className="w-1/3 flex justify-end items-center gap-10">
86     <span className="text-gray-400">{daysAgo}</span>
87     <Bookmark isBookmarked={isBookmarked} />
88   </div>
89 </div>
90
91 {/* Line */}
92 <hr className="border-gray-300 mt-2"/>
93
94 <div className="flex items-center text-sm text-gray-600 mt-2">
95   {/* Like Button */}
96   <div className="w-1/3">
97     <LikeButton onClick={handleLike} hasLiked={hasLiked} />
98   </div>
99   {/* Comment Count */}
100   <div className="w-1/3 flex justify-end">
101     <span className="text-gray-400">{commentCount} comments</span>
102   </div>
103   {/* Share Button */}
104   <div className="w-1/3 flex justify-end">
105     <ShareButton onClick={() => console.log("Shared!")} />
106   </div>
107 </div>
108 </div>
109 </div>
110 </div>
111 </div>
112 </div>
113 export default ArticlePreview;
```

## Article Preview Component

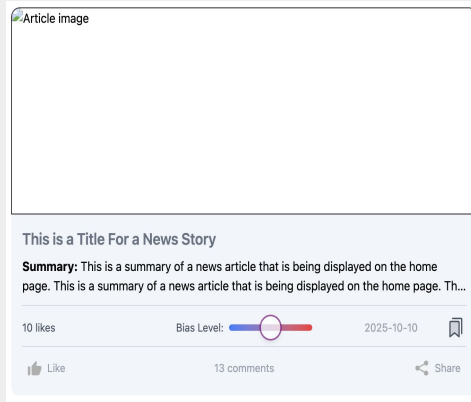
**Purpose:** Component is used in Trending, Bookmark, and Search page. The preview listed out the content and functionality of the article for the users to see

**Integration:** Acts as a central UI element connecting various parts of the application. It integrates into multiple pages, serving as entry points for full article details

**Challenge:** Ensuring interactive elements (buttons, icons) are accessible and maintain visual harmony within a dynamic layout.

**Solution:** Developed the component as a modular, reusable unit that can accept dynamic data via props.

**Design:** Component follows the style guidelines, using the global primary and tertiary color scheme, also incorporating animations and hover effects to enhance engagement



# Challenges and Insights - Bao Nguyen

## Challenges:

- **Responsive Design Complexity**
  - When we tried to implement the Navigation Bar, it was very easy to assemble all of the components in it. But later on, we realized that for different screen sizes, the Navigation Bar does not fit and align well.
- **Collaborative Coordination**
  - Sometimes, we have conflicting ideas about a particular functionality. Active communication and clear explanation helps to clear out confusion, and the team comes forward with a common agreement.
- **Reviewing Pull Requests (PR)**
  - Reviewing other PR can be challenging because as a reviewer, we must ensure that the code produced matches with the functionality. If we accept the malfunction code, later branches can be disrupted, causing delays and fixes.

## Insights:

- **Common Tailwind CSS theme**
  - This helps to render the UI better because rendering with props for Tailwind CSS does not work well. Therefore, I created a file for themes in App.csx for others to use.
- **Iterative Development Benefits**
  - Regular communication with the team and quick responses are crucial for identifying and resolving UI confusion.
- **Creation of Subfolders**
  - Subfolders are very important in component development because for some pages, some components are used and some are not used. Creating subfolders can help developers identify which components are related to what page, and this can also be effective if someone else wants to look at the code.

# Future Improvements & Next Steps - Bao Nguyen

## Admin Trackboard:

- Our current design is entirely user-based. However, some of the articles might be offensive, and the admin is required to intervene and delete them. The admin account can also be used to track user engagement, popular topics, and other users' complaints. The complaints can later be used to further develop and improve the website.

## Testing Optimizations

- Currently, testing and checking are done manually by team members. This is not as fast and not as reliable as unit testing and automated testing. Writing automated testing can help to catch issues early and much faster than manual testing. This will also maintain the key functionality of pages and components



# Assigned Work Summary - Kevin Li

## 1. Bias Rating Component Development

- **Issue:** [Implement NewsBridge Custom Scales / Ratings](#)
- **Closed:** Yes
- **Description:** Implement two types of bias scales / ratings to display or rate bias rating of an article. NewsBridge Bias Scale and User Bias Scale
- [Commit Link](#)
- [Commit Link](#)

## 3. Bookmark Component Development

- **Issue:** [Implement Bookmark Component](#)
- **Closed:** Yes
- **Description:** Implement the Bookmark component with on and off states.
- [Commit Link](#)

## 2. Profile Page Development

- **Issue:** [Implement Profile Page](#)
- **Closed:** Yes
- **Description:** Develop a Profile Page that allows users to alter their profile information.
- [Commit Link](#)

## 4. Bookmarked Articles Page Development

- **Issue:** [Implement Bookmarked Page](#)
- **Closed:** Yes
- **Description:** Implement Bookmarked Page with the following components: Navbar and ArticleList. Ensure that all articles' bookmarks are turned on
- [Commit Link](#)

## 5. Article List Component Development

- **Issue:** [Implement Article List Component](#)
- **Closed:** Yes
- **Description:** Implement Article List Component, which utilizes a list of Article Previews and possibly filters out some. To be used by trending, bookmarked, and search results page.
- [Commit Link](#)

## Pull Requests Merged:

1. [PR: Implemented the bias scale components](#)
2. [PR: Implemented bookmark component, added samples in for elements page](#)
3. [PR: Implemented profile page with updated requirements](#)
4. [PR: Refactored Home.tsx to utilize an ArticleList component](#)

# Code & UI Explanation - Kevin Li

Welcome, Kieran

Kieran Serra

kieranserra@umass.edu

Save Changes

First Name

Your First Name

Last Name

Your Last Name

Email

Your Email Address

Date of Birth

mm/dd/yyyy

Delete Account

## Profile Page

**Purpose:** Page that allows users to alter and view their profile Information, such as name, email, profile picture, etc.

**Integration:** If a user deletes their account, this page will route back to the sign-up page.

**Challenge:** Ensuring real-time updates when a user modifies their profile without excessive API calls.

**Solution:** Used local state to instantly reflect profile changes before sending updates to the backend. In the code snippets, whenever a user uploads a file, we create a temporary object URL to update the pfp.

**Design:** This page features a clickable avatar with a hover effect indicating an upload option. There is also an additional confirmation popup if a user clicks the delete account button.

```
<input
  ref={fileInputRef}
  type="file"
  accept="image/*"
  onChange={handleFileChange}
  className="hidden"
/>
```

```
const fileInputRef = useRef<HTMLInputElement>(null);

const handleFileChange = (event: React.ChangeEvent<HTMLInputElement>) => {
  const file = event.target.files?.[0];
  if (file) {
    // Currently : Updates the user's profile picture in state
    // To Do: Upload to the backend
    console.log("File selected:", file);
    const imageURL = URL.createObjectURL(file);
    setDisplayData((prevState) => ({
      ...prevState,
      profilePicUrl: imageURL,
    }));
  }
};
```

# Challenges and Insights - Kevin Li

## Challenges:

- **Merge Conflicts & Version Control:**
  - Coordinating code changes within a team sometimes led to conflicts, requiring rebasing, resolving conflicts and branch management.
- **State Management:**
  - Ensuring that UI updates reflected real-time changes. Initially for the profile page, my thought process was to only send changes to backend. However, this meant that the user must refresh / re-query the backend to see the changes reflected.
- **Backend Integration Delays:**
  - Initially working with mock data knowing that we would have to transition to API-based data fetching led to challenges in how to best format our mock data to resemble the data from endpoints.

## Insights:

- **Team Coordination is Vital:**
  - We leveraged effective communication and collaboration when resolving merge conflicts. By discussing changes with team members and understanding the context behind their code, we were able to minimize merge conflicts and quickly identify the best solutions if we did have conflicts.
- **Combining State with Backend calls**
  - One thing I learned was the need to strike a balance between local state management and backend synchronization. Minimizing the amount of API calls would lead to better scalability. As a result, for the profile page I utilized a combined approach, which leveraged local state for instant feedback while ensuring backend updates for long-term consistency.
- **Utilizing DTOs and Viewmodels**
  - This was Said's initiative, as I personally never had experience working with DTO and VM objects when working with frontend. Learning how to utilize DTOs with the data retrieved from the database was very informative on how to create the most reflective mock data.

# Future Improvements & Next Steps - Kevin Li

## Additional user-specific features:

- Add functionality for user accounts, allowing users to set **preferences** for news sources, topics, or categories they are interested in.
- **Dark / Light** mode for the UI.
- Implement a feature where users can review their **reading history**, making it easy to track previously read articles or revisit important stories.

## Notifications:

- If a new article was uploaded to the database, or a comment was submitted by another user, it will require re-fetching from some endpoint(s) to load this new data. (refreshing page)
- We can add a **notification** to the UI whenever a database update occurs (A new article has been generated!), so that users can be kept informed. This will of course require back-end integration.

# Assigned Work Summary - Said Mazaheri

## 1. Sign-In Component

- **Issue:** [Implement Sign-In Component](#)
- **Closed:** Yes
- **Description:** Sign-In page with the following:
  - Header + display line
  - Email input
  - Password input
  - Forgot password? button (routes to reset password)
  - Don't have an account? button (routes to sign up)
- [Commit Link](#)

## 2. Sign-Up Component

- **Issue:** [Implement Sign-Up Component](#)
- **Closed:** Yes
- **Description:** Sign-Up page with the following:
  - Header + display line
  - Email input
  - Password input + confirm password
  - Password constraint validation display
  - Already have an account? button (routes to sign in)
- [Commit Link](#)

## 3. Reset Password Component

- **Issue:** [Implement Reset Password Component](#)
- **Closed:** Yes
- **Description:** Reset-Password page with the following:
  - Header + display line
  - Email input
  - Cancel button
  - Enter button
- [Commit Link](#)

## 4. Sign-Out Component

- **Issue:** [Implement Sign-Out Component](#)
- **Closed:** Yes
- **Description:** Sign-Out popup with the following:
  - Image of question mark
  - "Are you leaving?" message
  - Cancel button → close pop-up
  - Yes button → routes to sign-in
- [Commit Link](#)

# Assigned Work Summary - Said Mazaheri

## 5. Comment-Section Component

- **Issue:** [Implement Comment Section Component](#)
- **Closed:** Yes
- **Description:** Previously implemented, we had progress to display comments within the comment section, and we had a text field to write text. Goal is to create a text field or area that produces comment displays.
- [Commit Link](#)

## 7. View Model + DTO Entities

- **Issue:** [Database Entity Development](#)
- **Closed:** Yes
- **Description:** Created DTOs for users, images, user bias ratings, news sources, bookmarks, comments, articles, likes, article mappings, and unbiased articles. Created view models for users, news sources, articles, comments, unbiased articles, and unbiased article previews.
- [Commit Link](#)

## 6. Open-Article Page

- **Issue:** [Implement Article Page](#)
- **Closed:** Yes
- **Description:** When articles are opened, display...
  - Article title
  - Bookmark button
  - Sources
  - Image
  - Summary
  - Article content
  - Audience bias rating
  - User bias rating (draggable)
  - Comment section
- [Commit Link](#)

### Pull Requests:

- [Improve Mocking & Correct Entities](#)
- [Implement Article Page](#)
- [Add UI/UX Design And Styling Guidelines](#)
- [Incorporate Bookmark Feature](#)
- [Comment Section Component](#)
- [Develop More Scalable Logo](#)
- [Implement Basic Database Entities \(DTOs and View Models\)](#)
- [Implement Log Out Component](#)
- [Implement Sign Up and Reset Password Components](#)
- [Design Reusable Authentication Form Module and Implement Sign In Component](#)
- [Route Component Building Pages](#)

# Code & UI Explanation - Said Mazaheri

## Article Page

**Purpose:** Sign-in page that gives user the full access to NewsBridge if provided correct information.

**Integration:** Acts as the first page the user sees. I created a near 1:1 copy of our wireframe.

**Challenge:** While this was one of the easier components I was responsible for, I realized that there was a strong similarity behind sign-in, sign-up, and reset password. I wanted to reduce code duplication.

**Solution:** I created the AuthenticationForm component. Since all pages listed above had headers, descriptions, fields, and routing footers, I created one component to prevent duplication.

**Design:** Consistent with our wireframes by matching the UI mock-ups. Consistent with our coding practices by reducing code duplication.

```
frontend > newsbridge > src > pages > @ Signin.jsx > @ SigninPage > @ handleFooterLinkClick
1 import React, { useState } from "react";
2 import { useNavigate } from "react-router-dom";
3 import AuthenticationForm from "../components/Authentication/AuthenticationForm";
4 import { ToastContainer, toast } from "react-toastify";
5 import "react-toastify/dist/ReactToastify.css";
6
7 const SigninPage: React.FC = () => {
8   const [formData, setFormData] = useState({ email: "", password: "" });
9   const navigate = useNavigate();
10
11   const handleEmailChange = (e: React.ChangeEvent<HTMLInputElement>) => {
12     setFormData({ ...formData, email: e.target.value });
13   };
14
15   const handlePasswordChange = (e: React.ChangeEvent<HTMLInputElement>) => {
16     setFormData({ ...formData, password: e.target.value });
17   };
18
19   const handleSubmit = (e: React.FormEvent) => {
20     e.preventDefault();
21     if (!formData.email || !formData.password) {
22       toast.error("Please fill out all fields.");
23       return;
24     }
25     if (!formData.email) {
26       toast.error("Email is required.");
27       return;
28     }
29     if (!formData.password) {
30       toast.error("Password is required.");
31       return;
32     }
33     // Handle form submission
34     console.log("Submitted data:", formData);
35   };
36
37   const handleFooterLinkClick = () => {}
38   navigate("/sign-up");
39
40   const handleResetPasswordLinkClick = () => {
41     navigate("/reset-password");
42   };
43
44   return (
45     <form onSubmit={handleSubmit}>
46       <AuthenticationForm
47         header="Welcome Back 🐼"
48         description="Welcome to NewsBridge, where diverse perspectives converge to deliver balanced, transparent news for an informed citizenry."
49         fields={
50           [
51             { label: "Email", type: "email", value: formData.email, onChange: handleEmailChange, showIcon: false },
52             { label: "Password", type: "password", value: formData.password, onChange: handlePasswordChange, showIcon: true },
53           ]
54         }
55         buttonText="Sign In"
56         footerText="Don't have an account? Sign Up"
57         footerLinkText="Sign Up"
58         onFooterLinkClick={handleFooterLinkClick}
59         onSubmit={handleSubmit}
60         secondFooterText="Forgot your password? Reset Password"
61         secondFooterLinkText="Reset Password"
62         onSecondFooterLinkClick={handleResetPasswordLinkClick}
63       />
64     </form>
65     <ToastContainer />
66   );
67 };
68
69 export default SigninPage;
```

**Welcome Back** 🐼

Welcome to **NewsBridge**, where diverse perspectives converge to deliver balanced, transparent news for an informed citizenry.

Email

Password

Sign In

Don't have an account? [Sign Up](#)

Forgot your password? [Reset Password](#)



# Challenges and Insights - Said Mazaheri

## Challenges:

- **Consistent UI/UX Design:**
  - Ensuring a unified design across all components was challenging. Without a solid design system in place, inconsistencies arose, requiring additional styling and revisions.
- **Team Coordination & Work Distribution:**
  - Balancing individual workloads and making sure everyone had clearly defined tasks took time. Miscommunication sometimes led to redundant work or delays.
- **Considering Optimization:**
  - We realized some early design choices could impact performance, such as excessive re-renders in certain components or unnecessary API requests. There times where we wanted to charge forward, but realized that our UI/UX choices now would have runtime consequences down the line.

## Insights:

- **Wireframes Were Key:**
  - While I had used wireframes in the past for my classes, I had never used them in earnest for personal projects. The importance of wireframes and help that they provided for me convinced me of their use going forward.
- **Effective Communication:**
  - Frequent team check-ins, async updates, and well-documented PRs helped keep everyone aligned and reduced misunderstandings. This was really underscored to me by how we were able to take each other's issues from each other to help one another out—our documentation was so good that the details were all on the GitHub.
- **Looking Ahead**
  - We had length discussions about these tradeoffs, especially when it came to the creation of view models from DTOs, as well as searching techniques using filters vs queries.

# Future Improvements & Next Steps - Said Mazaheri

## **Searching By Category/Tag:**

One of the ideas that we came up with early in the project, but ruled out of the MVP, was the tagging of articles based off of their subject matter. This could be genre—sports, politics, finance, etc—geography—USA, international, etc.—or others. We ruled it out because it was too complicated for such an automated application for the time being, but we could add it down the line.

## **Bias Analysis:**

One of the ideas I proposed early on in this project was the usage of NLP for an automated, “official” bias rating of our content to provide in addition to the audience’s bias rating. Like searching by category/tags, the volume of effort required for such a task made us remove it from our MVP. It could be added down the line.