DATE 2025.05.01

WebDev FINAL

ScreenDiary

Ting Yang

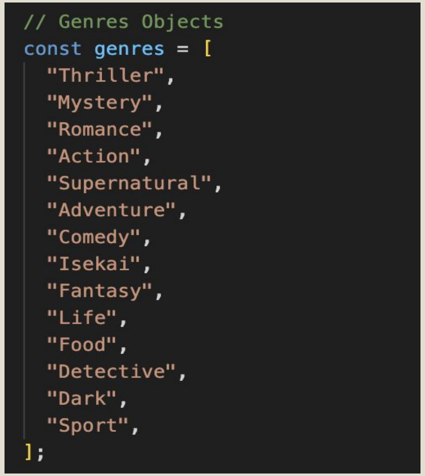
https://tinggg-yyy.github.io/Screen_Diary/

PROCESS

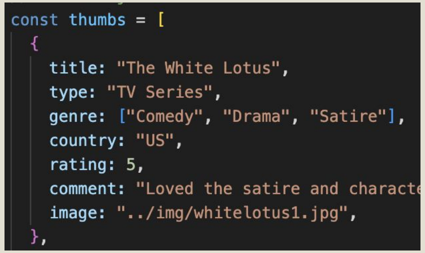
- **Watchlist Page**
  - Objects
  - Search Function
  - Filter Function
  - Lazy Loading Images
  - Page Number
  - Bootstrap (Dropdown)
  - Flip Animation

Filter Function

Data Setup

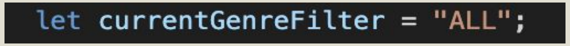


```
// Genres Objects
const genres = [
  "Thriller",
  "Mystery",
  "Romance",
  "Action",
  "Supernatural",
  "Adventure",
  "Comedy",
  "Isekai",
  "Fantasy",
  "Life",
  "Food",
  "Detective",
  "Dark",
  "Sport",
];
```



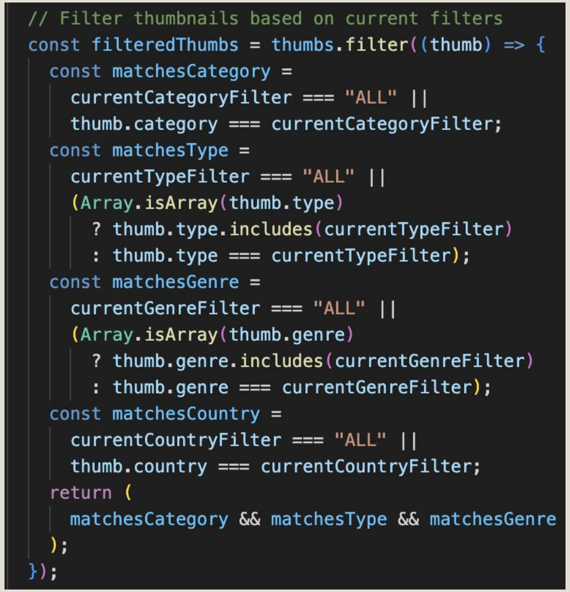
```
const thumbs = [
  {
    title: "The White Lotus",
    type: "TV Series",
    genre: ["Comedy", "Drama", "Satire"],
    country: "US",
    rating: 5,
    comment: "Loved the satire and characte
    image: "../img/whitelotus1.jpg",
  },
```

Default Unfiltered Status



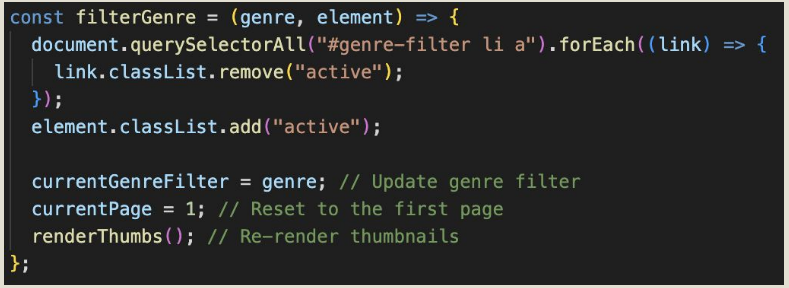
```
let currentGenreFilter = "ALL";
```

Check Array Values to Filter Thumbs



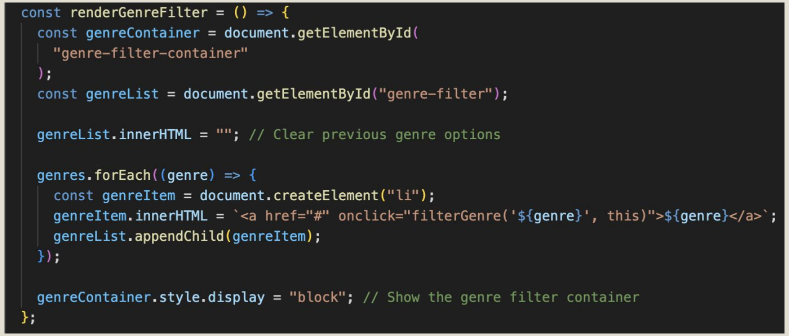
```
// Filter thumbnails based on current filters
const filteredThumbs = thumbs.filter((thumb) => {
  const matchesCategory =
    currentCategoryFilter === "ALL" ||
    thumb.category === currentCategoryFilter;
  const matchesType =
    currentTypeFilter === "ALL" ||
    (Array.isArray(thumb.type)
      ? thumb.type.includes(currentTypeFilter)
      : thumb.type === currentTypeFilter);
  const matchesGenre =
    currentGenreFilter === "ALL" ||
    (Array.isArray(thumb.genre)
      ? thumb.genre.includes(currentGenreFilter)
      : thumb.genre === currentGenreFilter);
  const matchesCountry =
    currentCountryFilter === "ALL" ||
    thumb.country === currentCountryFilter;
  return (
    matchesCategory && matchesType && matchesGenre
  );
});
```

Update the active filter
Re-render the thumbnails from page 1



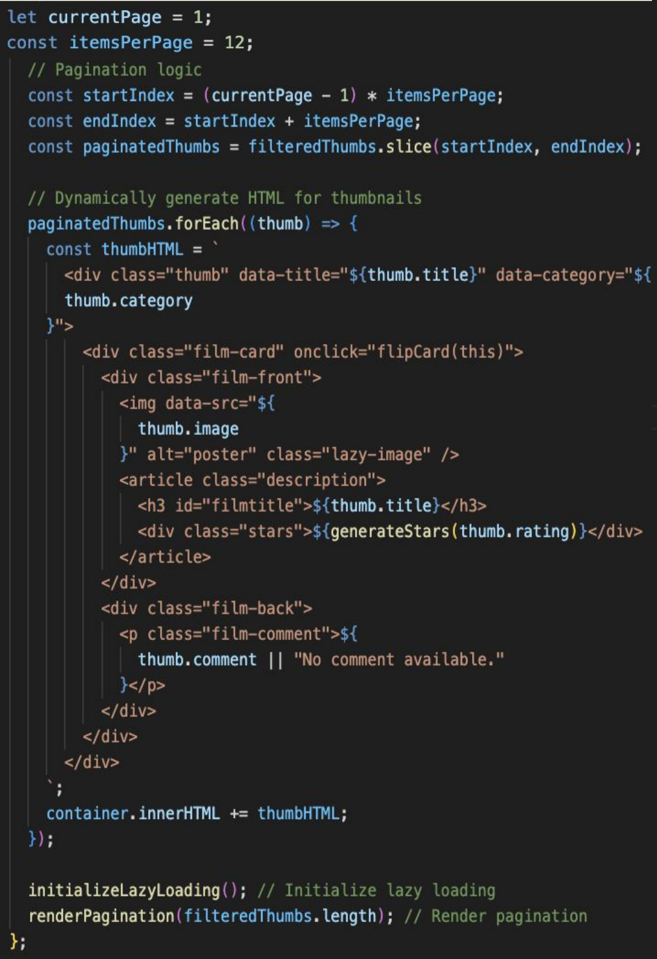
```
const filterGenre = (genre, element) => {
  document.querySelectorAll("#genre-filter li a").forEach((link) => {
    link.classList.remove("active");
  });
  element.classList.add("active");

  currentGenreFilter = genre; // Update genre filter
  currentPage = 1; // Reset to the first page
  renderThumbs(); // Re-render thumbnails
};
```

Generate Inner HTML

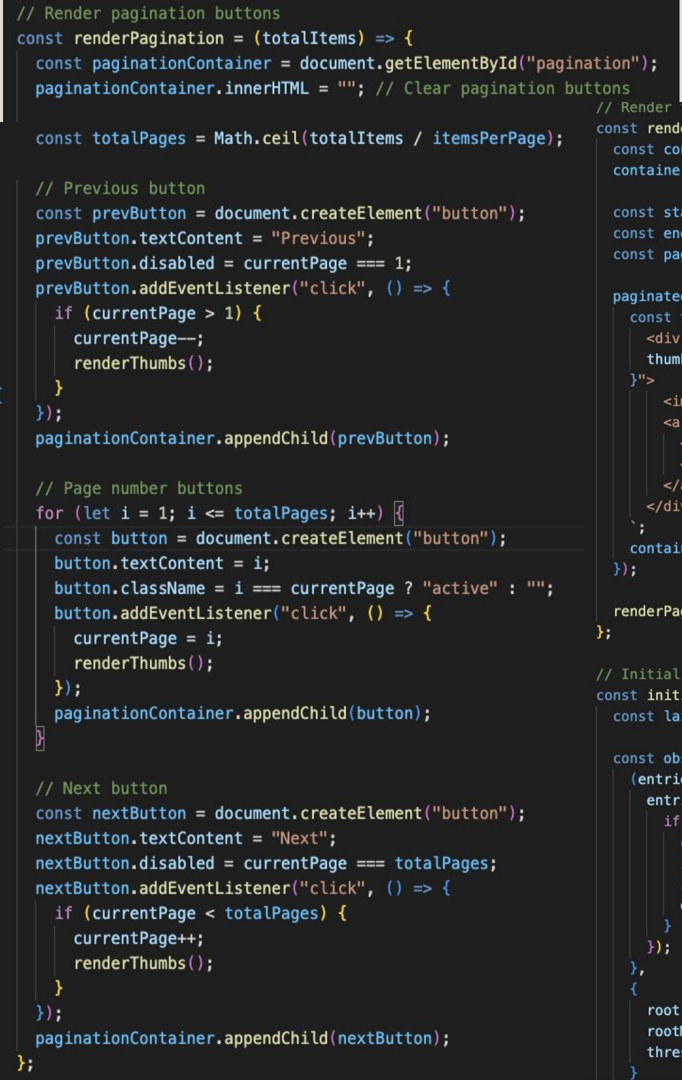


```
const renderGenreFilter = () => {
  const genreContainer = document.getElementById(
    "genre-filter-container"
  );
  const genreList = document.getElementById("genre-filter");

  genreList.innerHTML = ""; // Clear previous genre options

  genres.forEach((genre) => {
    const genreItem = document.createElement("li");
    genreItem.innerHTML = `<a href="#" onclick="filterGenre('${genre}', this)">${genre}</a>`;
    genreList.appendChild(genreItem);
  });

  genreContainer.style.display = "block"; // Show the genre filter container
};
```

Pagination

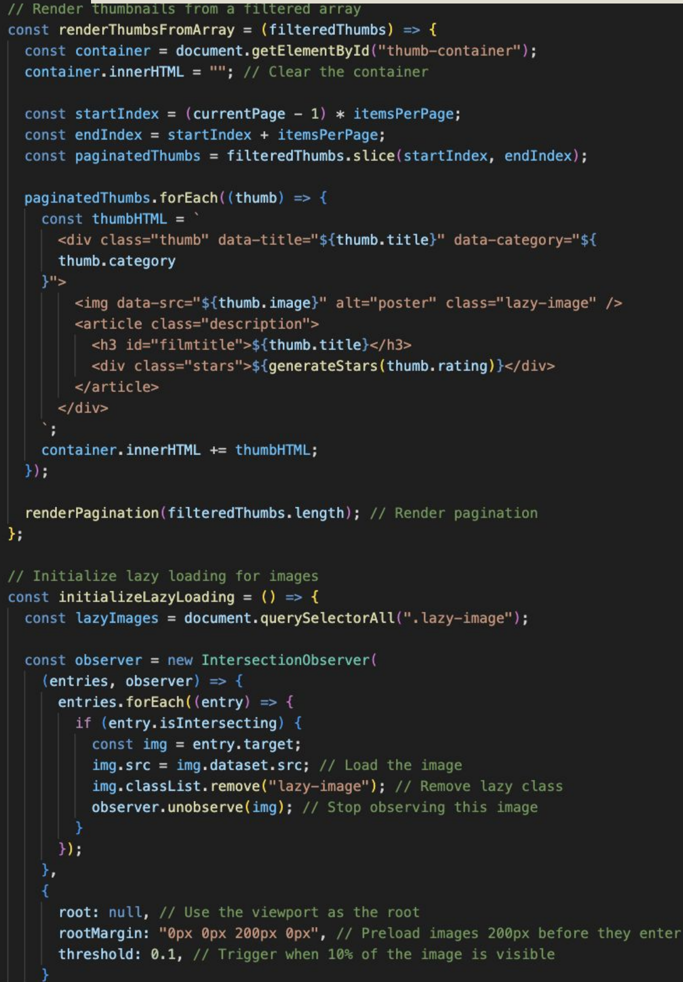


```javascript
let currentPage = 1;
const itemsPerPage = 12;
  // Pagination logic
  const startIndex = (currentPage - 1) * itemsPerPage;
  const endIndex = startIndex + itemsPerPage;
  const paginatedThumbs = filteredThumbs.slice(startIndex, endIndex);

  // Dynamically generate HTML for thumbnails
  paginatedThumbs.forEach((thumb) => {
    const thumbHTML = `
      <div class="thumb" data-title="${thumb.title}" data-category="${
      thumb.category
    }">
        <div class="film-card" onclick="flipCard(this)">
          <div class="film-front">
            <img data-src="${
              thumb.image
            }" alt="poster" class="lazy-image" />
            <article class="description">
              <h3 id="filmtitle">${thumb.title}</h3>
              <div class="stars">${generateStars(thumb.rating)}</div>
            </article>
          </div>
          <div class="film-back">
            <p class="film-comment">${
              thumb.comment || "No comment available."
            }</p>
          </div>
        </div>
      </div>
    `;
    container.innerHTML += thumbHTML;
  });

  initializeLazyLoading(); // Initialize lazy loading
  renderPagination(filteredThumbs.length); // Render pagination
};
```



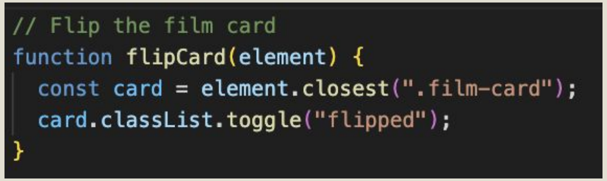
```javascript
// Render pagination buttons
const renderPagination = (totalItems) => {
  const paginationContainer = document.getElementById("pagination");
  paginationContainer.innerHTML = ""; // Clear pagination buttons

  const totalPages = Math.ceil(totalItems / itemsPerPage);

  // Previous button
  const prevButton = document.createElement("button");
  prevButton.textContent = "Previous";
  prevButton.disabled = currentPage === 1;
  prevButton.addEventListener("click", () => {
    if (currentPage > 1) {
      currentPage--;
      renderThumbs();
    }
  });
  paginationContainer.appendChild(prevButton);

  // Page number buttons
  for (let i = 1; i <= totalPages; i++) {
    const button = document.createElement("button");
    button.textContent = i;
    button.className = i === currentPage ? "active" : "";
    button.addEventListener("click", () => {
      currentPage = i;
      renderThumbs();
    });
    paginationContainer.appendChild(button);
  }

  // Next button
  const nextButton = document.createElement("button");
  nextButton.textContent = "Next";
  nextButton.disabled = currentPage === totalPages;
  nextButton.addEventListener("click", () => {
    if (currentPage < totalPages) {
      currentPage++;
      renderThumbs();
    }
  });
  paginationContainer.appendChild(nextButton);
};
```

Lazy Loading

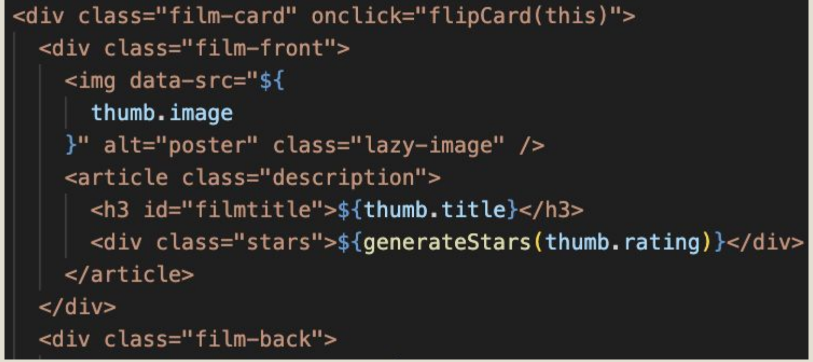


```javascript
// Render thumbnails from a filtered array
const renderThumbsFromArray = (filteredThumbs) => {
  const container = document.getElementById("thumb-container");
  container.innerHTML = ""; // Clear the container

  const startIndex = (currentPage - 1) * itemsPerPage;
  const endIndex = startIndex + itemsPerPage;
  const paginatedThumbs = filteredThumbs.slice(startIndex, endIndex);

  paginatedThumbs.forEach((thumb) => {
    const thumbHTML = `
      <div class="thumb" data-title="${thumb.title}" data-category="${
      thumb.category
    }">
        <img data-src="${thumb.image}" alt="poster" class="lazy-image" />
        <article class="description">
          <h3 id="filmtitle">${thumb.title}</h3>
          <div class="stars">${generateStars(thumb.rating)}</div>
        </article>
      </div>
    `;
    container.innerHTML += thumbHTML;
  });

  renderPagination(filteredThumbs.length); // Render pagination
};

// Initialize lazy loading for images
const initializeLazyLoading = () => {
  const lazyImages = document.querySelectorAll(".lazy-image");

  const observer = new IntersectionObserver(
    (entries, observer) => {
      entries.forEach((entry) => {
        if (entry.isIntersecting) {
          const img = entry.target;
          img.src = img.dataset.src; // Load the image
          img.classList.remove("lazy-image"); // Remove lazy class
          observer.unobserve(img); // Stop observing this image
        }
      });
    },
    {
      root: null, // Use the viewport as the root
      rootMargin: "0px 0px 200px 0px", // Preload images 200px before they enter
      threshold: 0.1, // Trigger when 10% of the image is visible
    }
```

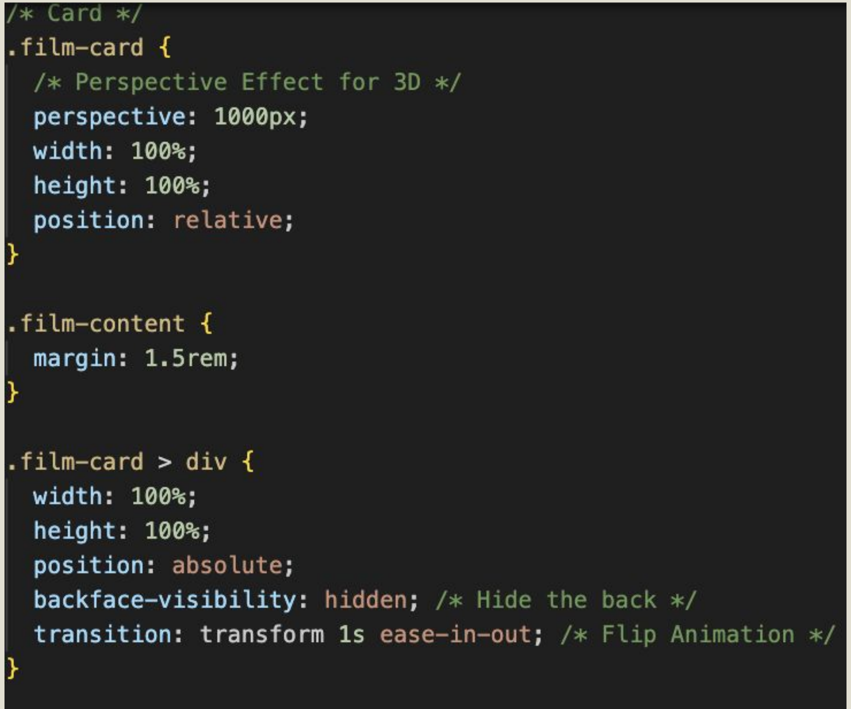
Flip Animation



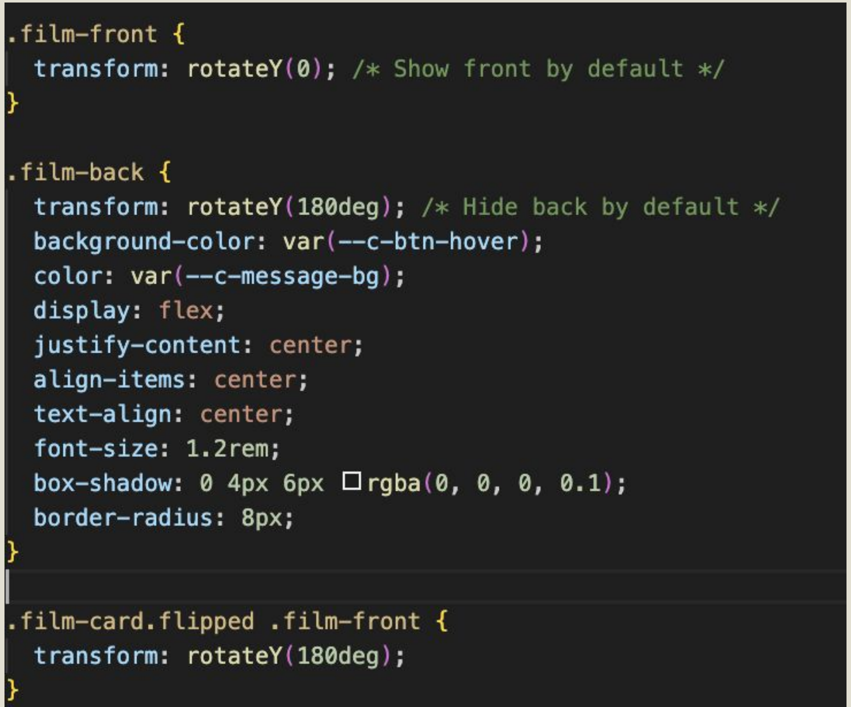
```javascript
// Flip the film card
function flipCard(element) {
  const card = element.closest(".film-card");
  card.classList.toggle("flipped");
}
```



```html
<div class="film-card" onclick="flipCard(this)">
  <div class="film-front">
    <img data-src="${
      thumb.image
    }" alt="poster" class="lazy-image" />
    <article class="description">
      <h3 id="filmtitle">${thumb.title}</h3>
      <div class="stars">${generateStars(thumb.rating)}</div>
    </article>
  </div>
  <div class="film-back">
```



```css
/* Card */
.film-card {
  /* Perspective Effect for 3D */
  perspective: 1000px;
  width: 100%;
  height: 100%;
  position: relative;
}

.film-content {
  margin: 1.5rem;
}

.film-card > div {
  width: 100%;
  height: 100%;
  position: absolute;
  backface-visibility: hidden; /* Hide the back */
  transition: transform 1s ease-in-out; /* Flip Animation */
}
```



```css
.film-front {
  transform: rotateY(0); /* Show front by default */
}

.film-back {
  transform: rotateY(180deg); /* Hide back by default */
  background-color: var(--c-btn-hover);
  color: var(--c-message-bg);
  display: flex;
  justify-content: center;
  align-items: center;
  text-align: center;
  font-size: 1.2rem;
  box-shadow: 0 4px 6px □rgba(0, 0, 0, 0.1);
  border-radius: 8px;
}

.film-card.flipped .film-front {
  transform: rotateY(180deg);
}
```

Issues

- **Style conflicts with Bootstrap**: Some of my custom styles were overridden by Bootstrap defaults, leading to unexpected formatting. I spent a lot of time debugging and ended up using !important to force certain styles to apply.

- **Too many images caused performance lag**: I implemented pagination and lazy loading to improve loading speed.

- **Combined filters required pagination reset:** Whenever any filter is changed, I needed to re-filter the dataset and reset the pagination—starting from page 1 and re-generating the correct number of pages based on the filtered results.

Learned

- DOMContentLoaded event
- Event handling (addEventListener & inline onclick)
- DOM manipulation (getElementById, querySelectorAll, etc.)
- Dynamically generating HTML
- Array operations (filter, forEach)
- Intersection Observer
- Pagination logic
- Flipping Animation

NEXT STEPS

- **Improve Pagination Display**
  Show fewer page numbers when total pages are too many (e.g., 1 2 3 ... 9 10), and add ellipsis for better UX.
- **Enhance Filter Layout**
  If there are too many filter options, switch to a **scrollable horizontal layout** instead of wrapping or stacking.
- **Expand Color Palette**
  Add more color variations to enrich the site's visual theme and improve contrast/accessibility.
- **Add More Animations & Interactions**
  For example, a **popcorn popping effect** when clicking on a thumbnail with a 5-star rating to increase user delight.

THANK YOU